information

MDPI

Article

SAM-Based Input Augmentations and Ensemble Strategies for Image Segmentation

Lorenzo Carisi †, Francesco Chiereghin †, Carlo Fantozzi * and Loris Nanni

Department of Information Engineering, University of Padova, 35122 Padua, Italy; loris.nanni@unipd.it (L.N.)
* Correspondence: carlo.fantozzi@unipd.it
† These authors contributed equally to this work.

Abstract

Despite the remarkable progress of deep learning in image segmentation, models often struggle with generalization across diverse datasets. This study explores novel input augmentation techniques and ensemble strategies to improve image segmentation performance. We investigate how the Segment Anything Model (SAM) can produce relevant information for model training. We believe that SAM offers a promising source of prior information that can be exploited to improve robustness and accuracy. Building on this, we propose input augmentation techniques that integrate SAM information directly into the images, enhancing the learning process of segmentation models. Each proposed augmentation method comes with its unique advantages; therefore, to leverage the strengths of each approach, we introduce AuxMix, a model trained with a combination of SAM-based augmentation methods. We conduct experiments on different state-of-the-art segmentation models, evaluating the effects of each method independently and within an ensemble framework. The results show that our ensemble strategy, combining complementary information from each augmentation, leads to robust and improved segmentation performance in a large set of datasets. We use only publicly available datasets in our experiments, and all the code developed to reproduce our results is available online on GitHub.

Keywords: image segmentation; input augmentation; ensembles; deep neural networks

check for updates

Academic Editor: Muhammad Kabir

Received: 6 August 2025
Revised: 25 September 2025
Accepted: 29 September 2025
Published: 30 September 2025

Citation: Carisi, L.; Chiereghin, F.; Fantozzi, C.; Nanni, L. SAM-Based Input Augmentations and Ensemble Strategies for Image Segmentation. *Information* **2025**, *16*, 848. https://doi.org/10.3390/info16100848

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Current research in big data, cloud computing, and computer vision is driving a revolution in fields such as autonomous driving, medical imaging, robotics, augmented reality (AR), remote sensing, and fashion e-commerce. These technologies have the potential to reshape our daily lives, from enabling safer and more efficient transportation to advancing healthcare, transforming industries with automation, and creating immersive user experiences in virtual environments.

To fully realize their potential, these fields rely heavily on computer vision to process and interpret complex visual data. Among the various tasks within computer vision, semantic segmentation stands out, allowing systems to operate effectively in dynamic and complex environments.

Semantic segmentation [1] refers to the task of classifying each pixel in an image into predefined categories, enabling the precise identification and delineation of objects and regions of interest. This capability is essential for systems that need to understand the content and structure of visual input at the pixel level. In autonomous driving [2], semantic segmentation plays a key role in identifying roads, vehicles, pedestrians, and other

obstacles, ensuring that self-driving cars can navigate safely and make decisions in real time. In medical imaging [3], it helps accurately identify anatomical structures and abnormalities, such as tumors, which enhances diagnostic precision and treatment planning. Robotics [4] relies on semantic segmentation for object recognition, navigation, and manipulation, boosting automation across various industries. In augmented reality (AR), semantic segmentation enables the seamless integration of virtual objects into real-world scenes, improving user interaction and overall experience. Remote sensing benefits from semantic segmentation in tasks such as land cover classification, vegetation monitoring, and environmental change detection, providing valuable insights for environmental and resource management. Lastly, in fashion e-commerce, semantic segmentation aids in the accurate categorization of clothing items, enhancing recommendations and enabling virtual try-on features for consumers [5].

However, these concepts could not have materialized earlier when semantic segmentation was viewed as a skill exclusive to humans. In the early stages, researchers introduced methods such as pixel-based, edge-detection-based, and region-based algorithms, which are now considered traditional approaches. However, although these methods laid the foundation for the field, they faced significant challenges: pixel-based methods were often sensitive to noise, edge-detection techniques struggled to form closed regions, and region-based methods had difficulty accurately delineating edges or fine details [6]. It was only with the advent of deep learning that semantic segmentation truly evolved, transforming the field and achieving levels of accuracy and efficiency previously considered unattainable.

The revolution started with convolutional neural networks (CNNs): one of the earliest and most influential approaches in this area was the fully convolutional network (FCN) [7], which replaced traditional fully connected layers with convolutional layers, providing a solid foundation for CNN-based segmentation methods.

Another important and widely adopted architecture is U-Net [8], which features an encoder–decoder structure complemented by skip connections. This design is particularly valuable in medical image segmentation, where access to large training datasets is often limited, as it helps to preserve important spatial information.

The DeepLab family of networks [9] has also been fundamental in pushing the boundaries of semantic segmentation. By incorporating atrous (dilated) convolutions, DeepLab can capture multiscale contextual information more effectively. Similarly, SegNet [10], another encoder–decoder-style architecture, uses pooling indices during the encoding phase, allowing for efficient upsampling of feature maps during decoding, all while maintaining computational efficiency.

These CNN-based methods [11] have had remarkable success in semantic segmentation tasks. However, CNNs do have certain limitations, especially when it comes to capturing global dependencies within images due to their localized convolution operations. To overcome these shortcomings, new models such as the Vision Transformer (ViT) [12] and the Pyramid Vision Transformer (PVT) [13] have been introduced, offering promising alternatives.

ViT has revolutionized the field of computer vision, achieving state-of-the-art (SOTA) performance in various visual recognition tasks. ViT relies on self-attention mechanisms within the transformer architecture to process images by dividing them into fixed-size patches. This approach enables the model to capture long-range relationships and global dependencies between image patches. On the other hand, PVT combines the benefits of CNNs and ViT by utilizing a hierarchical approach with multiscale feature pyramids. This method allows PVT to model both local details and global context effectively using sophisticated attention modules. PVT is trained with a combination of supervised and self-supervised learning techniques, which enhances its robustness and generalization ability.

Although the mentioned methods are significant, there is potential to improve their segmentation performance by combining them into an ensemble. Ensemble learning is a machine learning approach that integrates multiple models, known as base learners, to produce predictions or decisions that are more accurate than those of any single model [14]. The core idea of ensemble learning is to harness the collective strength of diverse models to enhance overall effectiveness. In this approach, base learners can be trained on the same dataset but with variations in algorithms, parameters, or training subsets. Each base learner independently generates predictions that are then combined to form the final output. This strategy provides numerous benefits, such as higher prediction accuracy, better resilience to overfitting, and enhanced robustness against noisy data. Ensemble learning is especially powerful when the base models are diverse and their errors are uncorrelated.

The META Segmentation Anything Model (SAM) [15] and the 2024 version SAM 2 [16] adopt a prompt-driven paradigm, integrating advanced transformer-based architectures with multimodal learning to generalize across a wide range of segmentation tasks, from fine-grained object delineation to contextual region identification, regardless of domain or dataset variability. The method proposed in ref. [17] introduces an innovative strategy in which SAM is utilized to generate high-quality segmentation masks for medical images. These masks are then used as additional input channels, alongside the original images, to augment the input data for the downstream segmentation models. By incorporating these SAM-generated masks, the approach provides complementary spatial and contextual information that helps improve the accuracy, robustness, and generalizability of segmentation models, particularly in complex and diverse medical imaging scenarios.

Given the significance of the aforementioned methods and the potential enhancement in segmentation accuracy through their combination, we propose an investigation of approaches for building ensembles of segmentation models. We introduce an ensemble approach that combines diverse segmentation methods and incorporates SAM information directly into the images in different ways. Building on the work of ref. [17], we developed alternative methods to their approach, with the goal of directly integrating additional output generated by SAM into the images. We explored various techniques to enrich the data, including alternative combination methods such as Principal Component Analysis (PCA) and variance-based techniques, as well as experimenting with different channel representations beyond simply adding information to the RGB channels. Different SAM-augmentation techniques create different training images, leading to different performance outcomes during testing. Each trained model may excel at segmenting certain images while underperforming on others: their fusion allows for state-of-the-art performance among CNN and transformer-based segmentation approaches.

The key contributions of this paper are summarized below.

1. We demonstrate how SAM and SAM 2 can provide valuable information for training segmentation models. Five different input augmentation methods (see Section 2.3) are proposed, each integrating SAM-derived information into images to improve the learning process of segmentation models. These methods consistently outperform the baseline across most datasets; each dataset tends to favor a different method, which indicates that new, meaningful information is being extracted, tailored to the specific characteristics of each dataset.
2. We introduce AuxMix, an ensemble model trained with a combination of SAM-based augmentation techniques, designed to harness the complementary strengths of each method. The rationale behind AuxMix is to integrate diverse and complementary sources of information into a unified ensemble. These include different color representations, the principal directions of variance in the data (via PCA), segmentation logits

produced by the SAM model, and stability scores of various segmentation masks; details are provided in Section 2.4.

3. We propose a new protocol to evaluate methods trained on the Kvasir-SEG and CVC-ClinicDB datasets by also incorporating the public datasets from Polyp-Gen, which are divided across six clinical centers; for details, see Section 2.5.1. Since the training set remains unchanged, this approach does not increase computational demands while expanding the range of unseen datasets used for testing.

Comprehensive experiments are conducted using various state-of-the-art segmentation models to evaluate the individual and ensemble effects of the proposed methods across multiple datasets. The study emphasizes transparency by using only publicly available datasets and making all the code for reproducing the results openly accessible on GitHub.

The paper is structured as follows: Sections 2 and 3 introduce and experimentally evaluate the input augmentation methods and ensembles with diverse topologies that achieve state-of-the-art performance. Section 3 also contains a discussion on the outcome of the experiments. Finally, Section 4 offers concluding remarks.

2. Materials and Methods

2.1. Base Models: PVT, HSNet

In this section, we describe the base models used in our ensembles: PVT and HSNet.

The Pyramid Vision Transformer (PVT) [18] is a transformer-based network that eliminates the use of convolutions. Its primary focus is capturing high-resolution representations from detailed input. The network architecture combines depth with a progressively narrowing pyramid structure to minimize computational demands. Moreover, a spatial-reduction attention (SRA) layer is employed to further reduce computational overhead.

The Hybrid Semantic Network (HSNet) [19] is a deep learning architecture built upon a PVT encoder, which enables it to capture multiscale features and long-range dependencies within the input images. This encoder is particularly adept at preserving both global context and fine-grained details, crucial for accurate segmentation tasks. The architecture incorporates a dual-branch structure to enhance its ability to distinguish between long-range dependencies and local appearance details. One branch focuses on modeling semantic spatial relationships and channel dependencies within lower-layer features, effectively suppressing noise and ensuring cleaner outputs. The other branch is responsible for bridging feature disparities through a semantic interaction mechanism, allowing the network to better integrate information across different layers of the feature map. The model was trained following the procedure in ref. [19], using the code available in the corresponding GitHub repository. This involved training with the structure loss and using the AdamW optimizer with an initial learning rate of 5×10^{-5}, which was halved at iterations 15 and 30. The training spanned 100 epochs, with input images resized to 352×352.

2.2. Foundation Models: SAM and SAM 2

The work of ref. [15] introduced a novel deep learning architecture known as the Segment Anything Model (SAM, also referred to as SAM1 in this paper when it is necessary to distinguish it from Version 2). This transformer-based model is designed for semantic segmentation; however, unlike traditional segmentation models, SAM can utilize prompts as inputs alongside image samples.

SAM was trained on the SA-1B dataset, also introduced by ref. [15]. SA-1B includes over 11 million high-resolution images and more than 1.1 billion segmentation masks, averaging approximately 100 masks per image. The dataset encompasses a wide variety of content, including objects, landscapes, people, and animals. This diversity enables SAM to demonstrate strong zero-shot performance across many segmentation tasks.

The SAM architecture comprises three main components: the image encoder, the prompt encoder, and the mask decoder. The image encoder generates a low-dimensional embedding for each input image while retaining its most significant features. As described by the authors, the image encoder is based on a pre-trained ViT [12], adapted to handle larger image sizes, following the approach in ref. [20]. The prompt encoder is one of the key contributions in the paper, enabling SAM to process a variety of visual prompts that guide the segmentation process by identifying target objects and refining segmentation masks. SAM supports two types of prompt: sparse (e.g., points, boxes, and text) and dense (e.g., masks). Sparse prompts such as points and boxes are represented by the sum of positional encodings [21] and learned embeddings, while text prompts are encoded using a pre-trained CLIP model [22]. Dense prompts, such as masks, are embedded using convolutional layers and combined element-wise with the image embedding. The mask decoder, the final component, incorporates a modified transformer block [23] and a dynamic mask prediction head. This block uses both prompt self-attention and cross-attention mechanisms to update embeddings with prompt-specific information, enabling effective prompt-to-image and image-to-prompt interactions. The decoder employs two transformer blocks to refine embeddings, which are then processed through two methods to produce output masks.

1. Two transposed convolutional layers.
2. A token-to-image attention block that updates embeddings and feeds them into a three-layer perceptron.

The final masks are generated through the dot product of the upsampled embeddings.

To address ambiguous prompt inputs, the decoder is designed to predict multiple output masks (typically three), which can be ranked based on their predicted Intersection over Union (IoU) confidence scores, also provided by the decoder.

The results presented in ref. [15] highlight SAM's remarkable zero-shot performance, achieving competitive results compared to fully supervised task-specific models in certain scenarios. SAM has also shown strong performance on tasks for which it was not explicitly trained, such as edge prediction. However, its performance may decline when applied to image samples that differ significantly from training data, such as medical images.

A survey conducted by ref. [24] provides an in-depth analysis of SAM's applications and derivative architectures. This study includes a dedicated section on medical imaging, highlighting SAM's significant potential in this domain. Medical image processing poses unique challenges, requiring expert knowledge to annotate data for training deep learning models. According to the authors, existing deep networks for medical imaging are typically designed for specific tasks and lack the ability to generalize across different applications. To address this limitation, recent studies have tailored SAM for medical image segmentation, enhancing its performance in this context.

Focusing on this specific application, the work presented in ref. [25] introduces SAM-Med2D, an adaptation of SAM for medical imaging. The authors demonstrate the effectiveness of incorporating learnable adapter layers into the image encoder, allowing the model to acquire domain-specific knowledge. In addition to SAM-Med2D, ref. [26] proposed a highly adaptable architecture for medical domain tasks, known as SAMUS. This architecture integrates trainable adapters into the ViT image encoder and introduces a lightweight parallel CNN encoder. The CNN encoder processes the same input as the ViT encoder and mitigates overfitting during training. The outputs of both encoders are combined point-wise to produce the final image embeddings. Unlike SAM-Med2D, the SAMUS training procedure leaves the prompt encoder and mask decoder frozen. The authors report excellent performance with SAMUS, outperforming similar SAM adaptations (e.g., MedSAM [27],

SAMed [28], and MSA [29]) while achieving reduced inference time due to its smaller input size and the use of the ViT base model.

SAM's latest version, SAM 2 [16], introduced notable advancements, expanding its capabilities and efficiency. One major improvement was the introduction of new memory components into the architecture, enabling robust support for video segmentation. While the original SAM uses ViT as its image encoder, SAM 2 transitions to a more compact Hiera encoder [30]. This change not only streamlines the model but also significantly reduces latency for both image and video processing, representing a marked enhancement in performance and usability.

Fine-Tuning SAM

By fine-tuning SAM, it is possible to leverage its broad generalization ability from large-scale datasets while improving it for specific applications.

Our fine-tuning methodology follows established approaches. To prepare the dataset for training, we generate bounding box prompts from the ground truth masks. These bounding boxes serve as simplified representations of regions of interest (ROI), guiding SAM during the segmentation process. Given both the image and bounding box prompts, SAM is fine-tuned to predict segmentation masks accordingly. In this study, we explored fine-tuning exclusively with SAM1 in Polyp datasets: the specific configuration and hyper-parameters used are summarized in Table 1. Results were not encouraging, so for all the experiments described in Section 3 we used the original pretrained SAM models.

Table 1. Summary of SAM1 fine-tuning protocol for Polyp segmentation.

Parameter	Value
Model	SAM1, ViT-H backbone
Trainable layers	Mask decoder only
Frozen layers	Vision encoder, Prompt encoder
Input resolution	Resized to `img_size` = 1024 (longest side), aspect ratio preserved
Batch size	1 (single image)
Number of epochs	20
Optimizer	Adam
Learning rate	1×10^{-4}
Weight decay	0
Loss function	`DiceCELoss` (`sigmoid` = True, `squared_pred` = True)
Data augmentation	Bounding box perturbation $\pm$ 20 px, no additional augmentations
Prompt type	Bounding boxes from masks
Early stopping	Not implemented

The training process is guided by a custom loss function, derived from the MONAI framework, which integrates the Dice similarity coefficient with cross-entropy loss. The function is defined as

$$\mathcal{L} = \lambda_{dice} \cdot \mathcal{L}_{dice} + \lambda_{ce} \cdot \mathcal{L}_{ce}, \quad (1)$$

where $\mathcal{L}_{dice}$ is the Dice loss, $\mathcal{L}_{ce}$ is the cross-entropy loss, and λ_{dice}, λ_{ce} are weight parameters. This combination is particularly effective for segmentation tasks as it balances accurate pixel-wise classification with the ability to measure overlap between predicted and actual segmentation masks.

2.3. Input Augmentation

As stated in Section 1, our input augmentation approaches are based on SAMAug [17]. In what follows, we first summarize the SAMAug approach; then, we provide a detailed explanation of our five variations of the approach (RG-segPrior, RG-logits, SV-segPrior,

PCA-segPrior, OurSAMAug), highlighting their individual contributions to improving image segmentation (all images are first rescaled to $[0, 1]$; any values outside this range due to augmentations are min–max rescaled to stay within $[0, 1]$, with no additional clipping.).

2.3.1. SAMAug

Given an input image, SAM produces multiple segmentation masks at different potential positions, each representing a plausible region of interest. These masks are stored in a list, each paired with a corresponding stability score that measures the confidence in its accuracy. A higher stability score indicates a mask that more reliably captures an actual object in the image. To obtain a comprehensive representation of the segmented regions, two key outputs are generated: the segmentation prior map and the boundary prior map (Figure 1).

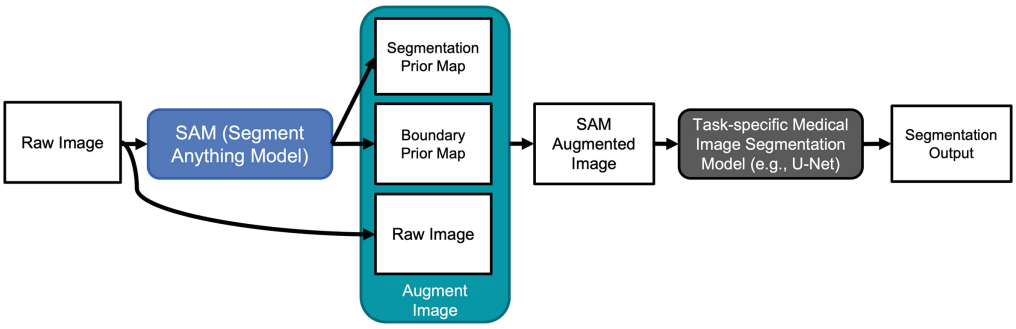


Figure 1. Scheme of the SAMAug method. Source: [17].

The segmentation prior map aggregates all individual segmentation masks, weighted by their stability scores. This process results in a unified representation of the regions where objects are most likely located within the image. In contrast, the boundary prior map serves a different purpose. Instead of considering full segmentation masks, it focuses solely on the outer boundaries of detected regions. By combining these boundaries, the map highlights object edges, providing additional spatial context to the overall scene.

To further clarify this process, Figure 2 presents visual examples of both prior maps. The second column displays the segmentation prior map, where each mask contributes based on its stability score, forming a comprehensive object representation. The third column presents the boundary prior map, emphasizing only the edges of the segmented regions and enhancing the spatial structure of the image.

After constructing the segmentation and boundary prior maps, the function enhances the input image x by integrating these maps. The augmentation process involves expanding the input image channels to include the prior maps. Specifically:

- the first channel contains the grayscale version of x;
- the second channel is populated with the segmentation prior map, providing additional segmentation-related information;
- the third channel accommodates the boundary prior map, enriching the image with boundary details.

The details of the input augmentation procedure are provided by the pseudo-code in Algorithm 1.

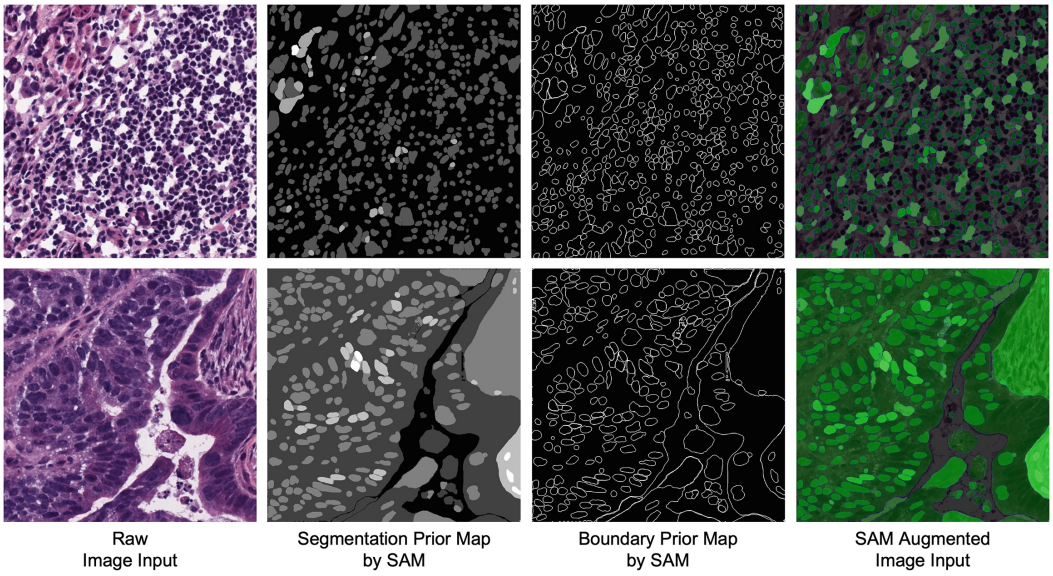


Figure 2. Examples of raw and augmented images used in the segmentation task, from the MoNuSeg dataset. Source: [17].

Algorithm 1 SAMAug

Input: tI (input image), *mask_generator* (SAM model mask generator)
Output: tI (augmented image with segmentation and boundary priors)
$masks \leftarrow mask_generator.generate(tI)$
$SegPrior \leftarrow \text{np.zeros}(tI.\text{shape}[0], tI.\text{shape}[1])$
$BoundaryPrior \leftarrow \text{np.zeros}(tI.\text{shape}[0], tI.\text{shape}[1])$
for $maskindex = 0$ **to** $len(masks) - 1$ **do**
    $thismask \leftarrow masks[maskindex].\text{'segmentation'}$
    $stability_score \leftarrow masks[maskindex].\text{'stability_score'}$
    $thismask_binary \leftarrow \text{np.zeros}(thismask.\text{shape})$
    $thismask_binary[\text{np.where}(thismask == \texttt{True})] \leftarrow 1$
    $indices \leftarrow \text{np.where}(thismask_binary == 1)$
    $SegPrior[indices] \leftarrow SegPrior[indices] + stability_score$
    $BoundaryPrior \leftarrow BoundaryPrior + \text{find_boundaries}(thismask_binary, \text{mode='thick'})$
    $BoundaryPrior[\text{np.where}(BoundaryPrior > 0)] \leftarrow 1$
end for
$tI[:,:,1] \leftarrow tI[:,:,1] + SegPrior$
$tI[:,:,2] \leftarrow tI[:,:,2] + BoundaryPrior$
Return tI

Although initial training may rely solely on SAM-augmented images, a more robust approach can be adopted when SAM fails to produce reliable prior maps. In such cases, training can incorporate both the original images x_i and their augmented counterparts $x_{\text{aug},i}$, leading to the following loss function:

$$\mathcal{L}_{\text{combined}} = \sum_{i=1}^{n} \beta \, \text{loss}(M(x_i), y_i) + \lambda \, \text{loss}(M(x_{\text{aug},i}), y_i), \tag{2}$$

where β and λ are hyperparameters that control the relative contributions of the raw and augmented images. The values of β and λ are crucial to optimize the performance of the model. Typically, both are set to 1 by default, though fine-tuning these parameters based on the dataset and task may be necessary.

When segmentation models trained on raw and SAM-augmented images are deployed, additional strategies can enhance the inference phase. A possible approach involves

performing inference twice for each test image—once using the raw image and once using its augmented version—then combining the results via an averaging ensemble strategy:

$$\hat{y} = \tau\big(M(x) + M(x_{\text{aug}})\big), \tag{3}$$

where $M(x)$ is the segmentation output of the raw image, $M(x_{\text{aug}})$ is the output of the augmented image, and τ represents a transformation function (e.g., softmax or sigmoid). In other words, the two predicted masks are summed pixel by pixel, then the function τ is applied per pixel to the sum to generate the final prediction $\hat{y}$.

Another possible strategy selects the most reliable segmentation output between $M(x)$ and $M(x_{\text{aug}})$. Reliability is estimated through the entropy of the predicted segmentation maps, favoring the output with the least uncertainty:

$$\hat{y} = \tau(M(x^*)), \tag{4}$$

where x^* is chosen from $\{x, x_{\text{aug}}\}$ by minimizing the entropy of the segmentation prediction

$$x^* = \arg \min_{x' \in \{x, x_{\text{aug}}\}} \text{Entropy}\big(\tau\big(M(x')\big)\big). \tag{5}$$

Entropy in the context of information theory quantifies uncertainty in a system. In the domain of image processing, the entropy of a L-level image I is defined as

$$\text{Entropy}(I) = -\sum_{i=0}^{L-1} p_i \log p_i, \tag{6}$$

where p_i represents the probability of pixels of value i. In practice, the probability is estimated with the histogram count, that is, with the number of pixels that assume value i in the image. This definition of entropy provides a statistical measure of randomness in the distribution across pixels, with higher entropy indicating greater unpredictability and lower entropy signifying greater uniformity. By selecting x^* based on entropy minimization, the model can prioritize the segmentation output with higher uniformity, potentially improving overall segmentation accuracy.

In the binary case ($L = 2$), entropy simplifies to

$$H(p) = -p \log p - (1 - p) \log(1 - p). \tag{7}$$

In our work, following the SAMAug implementation, we use an alternative binary entropy indicator defined as

$$EI = \text{mean}(|\tau - 0.5|), \tag{8}$$

where τ represents the transformed model logit for each pixel, mapping logits to probabilities. In our binary setting, we specifically use the sigmoid function. The entropy indicator measures the distance from maximum uncertainty (0.5) and, while not equivalent to entropy, serves as an effective proxy to select the most uncertain predictions. Aggregation is performed as the mean over all pixels in the image.

2.3.2. RG-segPrior

The RG-segPrior algorithm we propose computes a segmentation prior matrix that aggregates the stability scores of different segmentation masks; the matrix is then inserted into the blue channel of the image. This approach is particularly useful when enhancing images with spatial information from segmentation, while preserving the color structure of the original image. The choice of the blue channel is motivated by perceptual and empirical considerations. First, cones sensitive to blue are scarce in the fovea [31], so the human

visual system is less sensitive to variations in the blue channel. In turn, this makes the blue channel a good candidate for embedding additional information (see, e.g., ref. [32]) with minimal disruption to both human perception and, we believe, to segmentation models that are trained on data intended for human perception. Second, there is precedent in the literature for applying thresholding and other preprocessing techniques specifically to the blue channel [33], suggesting its utility in image analysis tasks. The pseudo-code of the algorithm is reported in Algorithm 2. In what follows, we describe the steps of the algorithm.

1. **Segmentation Mask Generation**: The algorithm begins by generating segmentation masks for the input image using a mask generator (for all experiments, we used the default parameters of `SamAutomaticMaskGenerator`; a complete list of the settings is provided in Appendix A.1.), a class inside the SAM library that automatically builds a prompt for the given image and outputs a list of binary masks. Each mask represents a region of interest within the image and is associated with a stability score that indicates the confidence of the segmentation.
2. **Segmentation Prior Calculation**: The *SegPrior* matrix is initialized as an empty matrix. The algorithm iterates over each segmentation mask, modifying *SegPrior* by adding the stability score for regions where the mask indicates the presence of a feature or object. This results in a prior that reflects the confidence in various regions of the image.
3. **Channel Separation**: The image is separated into its red, green, and blue channels. The red and green channels are kept unchanged.
4. **Segmentation Prior Integration**: The blue channel is replaced by *SegPrior*, which is calculated from the segmentation masks. This modification injects segmentation information into the image, highlighting areas of interest as determined by the segmentation model.
5. **Image Reconstruction**: The image is reconstructed by combining the unchanged red and green channels with the modified blue channel. The resulting image now embeds both the original image content and additional segmentation information.

Algorithm 2 Segmentation Prior Modification

procedure RG-SEGPRIOR(tI, *mask_generator*)
  $masks \leftarrow$ mask_generator.generate(tI)
  $SegPrior \leftarrow$ np.zeros($tI.shape[0], tI.shape[1]$)
  for each mask in masks **do**
    $SegPrior \leftarrow SegPrior +$ mask.stability_score $\cdot$ mask.segmentation
  end for
  $r, g \leftarrow tI[:, :, 0], tI[:, :, 1]$
  $b \leftarrow SegPrior$
  $modded_image \leftarrow$ np.dstack(r, g, b)
  return *modded_image*
end procedure

2.3.3. RG-logits

The RG-logits algorithm we propose modifies the blue channel of an image based on segmentation logits produced by the SAM model (Figure 3). The pseudo-code of the algorithm is reported in Algorithm 3. In the following, we describe the steps of the algorithm.

1. **Input Image Preparation**: The algorithm starts by setting the input image to the SAM model, preparing it for segmentation prediction. This allows the model to process

the image and generate relevant outputs, such as the segmentation mask and the corresponding logits.

2. **Segmentation Logit Prediction**: The SAM model returns the predicted mask along with its associated logits. These logits represent the model's confidence in different regions of the image, highlighting areas where specific features or objects are present.
3. **Channel Separation**: The image is separated into its red, green, and blue channels. The red and green channels are retained as is; the blue channel is modified based on the segmentation logits (Step 5).
4. **Logit Normalization**: The logits are normalized to fit within the standard image channel range of $[0, 255]$. This ensures that the logits can be represented as valid pixel values and applied to the blue channel of the image.
5. **Blue Channel Replacement**: The normalized logits replace the original blue channel of the image, resulting in a new image in which the blue channel now reflects the segmentation information. This modification highlights the areas of interest determined by the segmentation model.
6. **Image Reconstruction**: The image is reconstructed by combining the unchanged red and green channels with the modified blue channel. The resulting image now displays the segmentation information while maintaining the original color integrity.

Algorithm 3 Logit-based Modification

procedure RG-LOGITS($tI, mask_generator$)
  $mask_generator.predictor.set_image(tI)$
  $mask, logits \leftarrow$ mask_generator.predict($return_logits = True$)
  $r, g \leftarrow tI[:, :, 0], tI[:, :, 1]$
  $b \leftarrow \frac{(mask[0] - \min(mask[0])) \times 255}{\max(mask[0]) - \min(mask[0])}$
  $modded_image \leftarrow$ np.dstack(r, g, b)
  return $modded_image$
end procedure

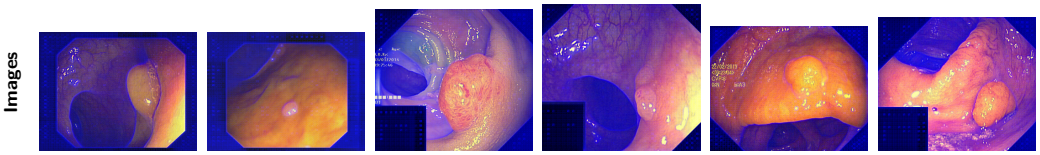


Figure 3. Examples of images augmented with RG-logits.

2.3.4. SV-segPrior

The SV-segPrior algorithm we propose is designed to integrate the same information as the other techniques proposed but by changing the color representation of the image. Specifically, the image is converted to the HSV color space (see Appendix A.2 for conversion details), and the additional information is embedded into the hue channel. This approach leverages SAM's masks and stability scores to inject its knowledge of areas of interest while preserving the original saturation and brightness channels in the transformed image. The pseudo-code of the algorithm is reported in Algorithm 4. In what follows, we describe the steps of the algorithm.

1. **Input Image Preparation**: The input image is converted to floating-point format to prepare it for processing. This ensures the compatibility of the pixel values with subsequent operations.
2. **Segmentation Prior Generation**: Using the segmentation masks generated by the SAM model, a semantic prior map is created. For each mask, the segmentation region

is weighted by its stability score, and these weights are accumulated across all masks to produce the segmentation prior.

3. **Color Space Conversion**: The image is converted from RGB to HSV color space. This transformation separates the image into hue (H), saturation (S), and brightness (V) channels, allowing for isolated modifications to the hue channel.
4. **Channel Separation and Modification**: The segmentation prior is assigned to the H channel, effectively encoding the semantic information in the hue component of the HSV image. The S and V channels remain unchanged.
5. **Image Reconstruction**: The modified H channel is recombined with the original S and V channels to reconstruct the modified HSV image. The resulting image integrates semantic information directly into its color representation.

Algorithm 4 HSV-based Segmentation Prior Modification

procedure SV-SEGPRIOR(tI, *mask_generator*)
  $masks \leftarrow$ mask_generator.generate(tI)
  $SegPrior \leftarrow$ zeros($tI.shape[0], tI.shape[1]$)
  for all $mask \in masks$ **do**
    $SegPrior \leftarrow SegPrior +$ mask['segmentation'] $\times$ mask['stability_score']
  end for
  $hsv_image \leftarrow$ color.rgb2hsv(tI)
  $h, s, v \leftarrow SegPrior, hsv_image[:, :, 1], hsv_image[:, :, 2]$
  $modded_image \leftarrow$ np.dstack(s, v, h)
  return *modded_image*
end procedure

This algorithm is particularly effective in scenarios where the hue channel can serve as a visual representation of semantic information. By embedding segmentation priors into the hue channel, the algorithm provides an intuitive way to analyze and visualize areas of interest.

2.3.5. PCA-segPrior

The following algorithm we propose aims to introduce SAM's information without completely losing any image channel. All prior calculations are performed as in the original SAMAug method, but the three channels of the image are then mapped into two channels using Principal Component Analysis (PCA) (see Appendix A.2 for PCA implementation details). This process is carried out to leave an empty channel in order to incorporate SAM's output (Figure 4). PCA is a statistical technique that finds the most significant directions of variance in the data and projects the image data onto those directions. The pseudo-code of the algorithm is reported in Algorithm 5. In the following, we describe the steps of the algorithm.

1. **Segmentation Mask Generation**: The algorithm begins by generating segmentation masks using a mask generator. These masks represent regions of interest within the image, and each mask is associated with a stability score, which indicates the confidence in the mask's correctness.
2. **Segmentation Prior Calculation**: The *SegPrior* matrix is initialized as a blank matrix (zeros). Then, for each segmentation mask, the algorithm updates *SegPrior*. This update involves multiplying each mask by its corresponding stability score and accumulating the results. The segmentation prior thus reflects the confidence in various regions of the image based on the available segmentation masks.
3. **Dimensionality Reduction via PCA**: The image is subject to PCA to reduce its dimensionality from three channels (RGB) to two channels. The reduced representation

captures the key features of the image in fewer dimensions, making it more efficient for further processing.

4. **Scaling of PCA and Segmentation Data**: The PCA output and the segmentation prior are normalized to the range $[0, 255]$ to ensure that the data are properly adjusted for visualization and processing.
5. **Image Reconstruction**: The image is reconstructed by combining the scaled PCA results and the segmentation prior into the red, green, and blue channels of the image. The red and green channels come from the PCA output, while the blue channel is influenced by the segmentation prior. This modified image now encodes both the reduced-dimensional representation of the image and additional information from the segmentation.

Algorithm 5 PCA and Segmentation Prior Modification

procedure PCA-SEGPRIOR(tI, *mask_generator*)
  $masks \leftarrow$ mask_generator.generate(tI)
  $SegPrior \leftarrow$ np.zeros($tI.shape[0], tI.shape[1]$)
  for each mask in masks **do**
    $thismask \leftarrow$ mask.segmentation
    $s_score \leftarrow$ mask.stability_score
    $SegPrior \leftarrow SegPrior +$ thismask $\times\ s_score$
  end for
  $pca \leftarrow$ PCA($n_components = 2$)
  $image_pca \leftarrow pca.fit_transform(tI)$
  $image_pca_scaled \leftarrow$ scale($image_pca$)
  $r, g \leftarrow image_pca_scaled[:, 0], image_pca_scaled[:, 1]$
  $b \leftarrow$ scale($SegPrior$)
  $modded_image \leftarrow$ np.dstack(r, g, b)
  return *modded_image*
end procedure

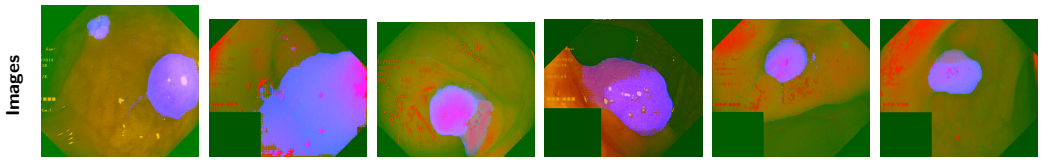


Figure 4. Examples of images augmented with PCA-segPrior.

We can identify three principles behind this algorithm.

- **Image Segmentation**: the segmentation prior improves the analysis by injecting spatially relevant data that reflect the structure and boundaries of objects in the image. By modifying the color channels based on segmentation, the image highlights areas of interest, making further analysis or feature extraction more accurate.
- **Dimensionality Reduction**: reducing the dimensionality of the image using PCA can significantly simplify computational tasks, especially when processing large datasets or when the main features of the image can be captured in fewer dimensions. It also helps to reduce noise by focusing on the principal components of the image.
- **Visual Enhancement**: the combination of PCA and segmentation can also aid in visual enhancement, where segmentation-prioritized areas are highlighted in the image, making it easier to interpret or present visually.

2.3.6. OurSAMAug

This algorithm we propose is a simple modification of the original SAMAug algorithm, consisting of the integration of an additional prior matrix to the R channel. In this case, we build the additional *AreaPrior* matrix that contains values between 0 and 255, proportional to the corresponding mask generated by SAM. The rationale behind this choice is to simply add the information about the predicted mask size to the original image. The following describes the steps of the algorithm that differ from the general description provided in Section 2.3.1. The full pseudo-code of the algorithm is reported in Algorithm 6.

1. **Prior matrix calculation**: In this stage, both the *SegPrior* matrix and the *BoundaryPrior* matrix are calculated as described for SAMAug. The *AreaPrior* matrix is initialized to a zero matrix; then, for each mask provided by SAM, the corresponding area in the matrix is colored in proportion to the area of the mask.
2. **Prior Integration**: The *SegPrior* matrix is summed to the G channel and the *BoundaryPrior* matrix is summed to the B channel, as in the SAMAug method. The *AreaPrior* matrix is added to the R channel.
3. **Image Reconstruction**: The image is reconstructed by merging the blue and green channels containing the same information as SAMAug (*SegPrior* and *BoundaryPrior*), plus the red channel with the additional *AreaPrior*. The final image then contains the same information as SAMAug, plus some additional area information regarding the size of the detected masks.

Algorithm 6 Our simple modification of the SAMAug algorithm

procedure OURSAMAUG(tI, *mask_generator*)
  $masks \leftarrow$ mask_generator.generate(tI)
  $SegPrior \leftarrow$ np.zeros($tI.shape[0], tI.shape[1]$)
  $BoundaryPrior \leftarrow$ np.zeros($tI.shape[0], tI.shape[1]$)
  $Area_prior \leftarrow$ np.zeros($tI.shape[0], tI.shape[1]$)
  for each mask in masks **do**
    $SegPrior \leftarrow SegPrior +$ mask.stability_score $\cdot$ mask.segmentation
    $BoundaryPrior \leftarrow BoundaryPrior +$ find_boundaries($mask$)
    $BoundaryPrior[BoundaryPrior > 0] \leftarrow 1$
    $Area \leftarrow \frac{mask['area']}{tI.shape[0] \times tI.shape[1]}$
    $Area_prior[mask == 1] \leftarrow area_prior[mask == 1] + area$
  end for
  $tI[:, :, 0] \leftarrow tI[:, :, 0] + Area_prior$
  $tI[:, :, 1] \leftarrow tI[:, :, 1] + SegPrior$
  $tI[:, :, 2] \leftarrow tI[:, :, 2] + BoundaryPrior$
  return tI
end procedure

For further details, a visual comparison of each newly added layer/prior map is presented in Appendix B, making it easy to see how each one influences the generation of the final mask.

In addition to all the above augmentations, we use in the baseline tests the data augmentation technique presented in ref. [1] under the name of DA3. This data augmentation involves applying multiscale strategies (e.g., 1.25, 1, 0.75) to reduce the network's sensitivity to scale variations. Lastly, random perspective transformations are applied to the input image with a 50% probability, and random color adjustments are performed with a 20% probability as part of the augmentation process.

2.4. Ensembles

As well known in the literature, an effective ensemble requires the combination of methods that extract complementary information from the input pattern. To this end, we designed our approach by integrating diverse strategies for generating input images used by the segmentation model. These include alternative color space representations, principal components capturing the dominant variance in the dataset (via PCA), segmentation logits produced by the SAM model, and stability scores derived from various segmentation masks. To enhance the differences between the models, we employed both Version 1 and Version 2 of SAM. All models within each ensemble share the same architecture and hyperparameters; the only variation is the random seed, which differs for each snapshot. To mitigate the risk of overfitting and maintain simplicity in the ensemble, we combined the segmentation approaches forming our proposed ensemble using the mean rule. For all datasets and tests, the averaging was performed after the activation function, which is always a sigmoid, and the resulting outputs were thresholded in the range [0,1] with a threshold of 0.5.

- **Baseline($\mathbf{K}_X$)**: K HSNet/PVT models (X = 'H' for HSNet, X = 'P' for PVT) are combined using the mean rule, employing standard data augmentation as the augmenting strategy.
- **AuxMix($_X$)**: fusion by the mean rule among nine HSNet/PVT models (X = 'H' for HSNet, X = 'P' for PVT): three HSNet/PVT models for Baseline$_X$ and one HSNet/PVT model for each of the following strategies: SAM1_RG-logits, SAM2_RG-logits, SAM1_SAMAug, SAM2_SAMAug, SAM1_PCA-segPrior and SAM2_PCA-segPrior (see Figure 5).
- **AuxMix($_{H+P}$)**: fusion by the mean rule of AuxMix($_H$) and AuxMix($_P$).
- **HUGE($_X$)**: fusion by the mean rule among 27 HSNet/PVT models (X = 'H' for HSNet, X = 'P' for PVT): nine HSNet/PVT models for Baseline$_X$; three HSNet/PVT models for each of the following strategies: SAM1_RG-logits, SAM2_RG-logits, SAM1_SAMAug, SAM2_SAMAug, SAM1_PCA-segPrior and SAM2_PCA-segPrior.
- **HUGE**: fusion by the mean rule of HUGE($_H$) and HUGE($_P$).

Figure 5 shows the scheme of the proposed ensemble AuxMix.

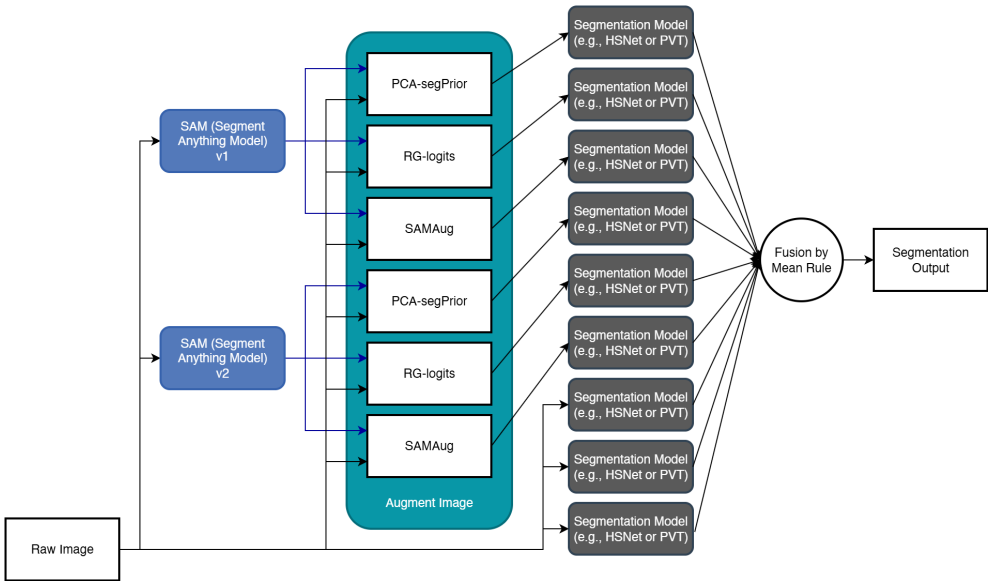


Figure 5. Scheme of our proposed ensemble: AuxMix.

2.5. Datasets

To evaluate the performance of our techniques, we conducted experiments on several datasets, commonly used in the field, to ensure the generalizability of the proposed approach. We considered different segmentation domains, as detailed below.

2.5.1. Polyp Segmentation

We ran tests with the following datasets.

- **CVC-T** [34] contains 300 images. It is a test set derived from the larger CVC-EndoSceneStill dataset, which includes colonoscopy images with different polyp presentations.
- **CVC-ClinicDB** [35] (ClinDB) includes 612 images extracted from 31 videos of colonoscopy procedures. Expert annotations identify polyp regions, and ground-truth data are also available for light reflections. The images in this dataset are uniformly sized at 576 $\times$ 768 pixels.
- **Kvasir-SEG** [36] (Kvasir) contains 1000 images meticulously labeled and verified by medical professionals. This dataset features various segments of the digestive system, including both healthy and diseased tissue. The images have resolutions ranging from 720 $\times$ 576 pixels to 1920 $\times$ 1072 pixels and are organized into folders based on content. Some images also include a small picture-in-picture display indicating the position of the endoscope within the body.
- **CVC-ColonDB** [37] (ColDB) contains 380 images and provides a diverse range of polyp appearances to maximize dataset variability and encompass various polyp types and scenarios.
- **ETIS-LaribPolypDB** [38] (ETIS) consists of 196 colonoscopy images, which are valuable for evaluating segmentation performance due to their variety and quality.
- **PolypGen** [39] is an open-access resource containing 1537 polyp images. It was collected from six centers across Europe and Africa, offering a total of 3762 positive frames and 4275 negative frames. These images represent diverse populations, endoscopic systems, and surveillance expertise from Norway, France, the United Kingdom, Egypt, and Italy.

We used the same testing protocol proposed in the literature. The training set consisted of 1450 images drawn from the largest datasets, including 900 from Kvasir and 550 from ClinDB. The test set for our experiments contained the remaining images, specifically 100 from Kvasir, 62 from ClinDB, and all images from ColDB, CVC-T, and ETIS. These polyp datasets are available at https://github.com/james128333/HarDNet-MSEG (accessed on 4 August 2025). In addition, we conducted experiments using the same training set of 1450 images and PolypGen as the test set. PolypGen is available at https://github.com/DebeshJha/PolypGen (accessed on 4 August 2025).

2.5.2. Segmentation of X-Ray Images

VinDr-RibCXR [40] (Ribs) is a dataset focused on rib segmentation in medical imaging; it contains 245 images. The images are taken from chest radiographs, and the dataset is annotated with precise rib region boundaries, providing a valuable resource to evaluate segmentation algorithms in the context of skeletal structures. We use the split training/test set suggested by the original authors of this dataset [40]. Instructions to obtain the dataset are available at https://github.com/vinbigdata-medical/MIDL2021-VinDr-RibCXR (accessed on 5 August 2025).

2.5.3. Camouflaged Object Segmentation

CAMO [41] is a camouflaged object segmentation dataset that provides a challenging set of 1250 images containing various objects camouflaged in natural scenes. The dataset is

designed to evaluate segmentation algorithms in detecting objects that blend with their background, offering a significant challenge for image analysis techniques. We divide CAMO into training and test set as suggested by the original creators of the dataset [41]. The dataset can be downloaded at https://sites.google.com/view/ltnghia/research/camo (accessed on 5 August 2025).

2.5.4. Locust Segmentation

Locust [42] includes 994 images for the segmentation of locusts in natural environments. It offers a rich set of images with high variability in lighting, background, and occlusions, which are typical in real-world scenarios, making it useful for testing the robustness of segmentation techniques. We split Locust for training and testing as suggested by the original authors of the dataset [42]. The link to the dataset is https://github.com/Chloe-Liu33/Locust-mini (accessed on 5 August 2025).

3. Results and Discussion

We performed the experiments in three phases.

1. First, we tested the effectiveness of our proposed input augmentation strategies without resorting to ensembling, that is, using base models.
2. Then, we tested the performance of our proposed ensembles.
3. Lastly, we carried out ablation studies to quantify the impact of the different implementation choices in our strategies. Some studies could be performed with the experimental data collected in the second phase, while others required new experiments.

During training, all networks processed images resized to a uniform input size (352×352). During testing, input images were resized to match the network's input dimensions and the output masks were resized back to their original image dimensions to compute the performance metrics. In all tests, when referring to a SAM-augmented model, the following circumstances always hold.

1. The model is trained with the combined loss function $\mathcal{L}_{\text{combined}}$ discussed in Section 2.3.1, meaning it is trained with both raw and augmented images along with the corresponding ground truth.
2. During test time, two inferences of the same model are performed and the mask with lower entropy is chosen.
3. No other classical augmentation techniques are used alongside SAM-augmentation: each model is trained using exactly one input augmentation algorithm.

We employed two standard metrics as performance indicators: the Dice score and the Intersection over Union (IoU). These metrics facilitate comparison with other studies, offer valuable insight into segmentation accuracy, and are well-suited for diverse datasets. In the formulas below, true positives (TPs), true negatives (TNs), false positives (FPs), and false negatives (FNs) are used to represent the respective components; A is the predicted mask and B is the ground truth mask. The Dice score is defined as

$$F1Score = Dice = \frac{|A \cap B|}{|A| + |B|} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}. \tag{9}$$

The IoU is defined as

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN}. \tag{10}$$

3.1. Input Augmentation

To avoid an excessive number of large tables, we report only a subset of all possible tests. Specifically, for each SAM-based strategy, we include only the results obtained using HSNet, as this base model provides the best performance. Each reported score is the average of five different runs. Results are summarized in Tables 2 and 3, showing the Dice scores for all the proposed augmentation techniques. These tables provide valuable insight into how different segmentation prior augmentations impact performance, offering a clear view of the trade-offs involved in selecting the most effective method for specific datasets or tasks. In these tables, Baseline(1_X) indicates the performance of model X when trained only using standard data augmentations.

Table 2. Input augmentation: performance (Dice score, arithmetic mean over 5 runs) obtained by HSNet for different augmentation methods in Polyp datasets. The last column provides the average score over the datasets. The best scores in each column are marked in bold.

Method	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
Baseline(1_H)	0.903	0.917	0.910	0.819	**0.816**	0.873
SAM1_RG-segPrior	0.894	0.938	**0.925**	0.823	0.800	0.876
SAM1_RG-logits	0.898	0.944	0.917	0.816	0.808	**0.877**
SAM1_SV-segPrior	0.899	0.943	0.910	0.811	0.756	0.864
SAM1_PCA-segPrior	0.903	0.945	0.905	0.820	0.777	0.870
SAM1_SAMAug	**0.907**	0.932	0.918	0.811	0.768	0.867
SAM1_OurSAMAug	0.880	0.943	0.913	0.814	0.769	0.861
SAM2_RG-segPrior	0.878	0.936	0.915	0.824	0.798	0.870
SAM2_RG-logits	0.890	0.934	0.910	0.814	0.781	0.866
SAM2_SV-segPrior	0.890	0.927	0.901	0.807	0.729	0.851
SAM2_PCA-segPrior	0.896	0.945	0.914	0.808	0.778	0.868
SAM2_SAMAug	0.899	0.930	0.917	**0.826**	0.766	0.868
SAM2_OurSAMAug	0.905	**0.946**	0.920	0.820	0.781	0.875

Table 3. Input augmentation: performance (Dice score, arithmetic mean over 5 runs) obtained by HSNet for different augmentation methods in non-Polyp datasets. The best scores in each column are marked in bold.

Method	Ribs	CAMO	Locust
Baseline(1_H)	**0.863**	0.811	0.882
SAM1_RG-segPrior	0.854	0.807	**0.885**
SAM1_RG-logits	0.850	0.796	0.873
SAM1_SV-segPrior	0.855	0.805	0.867
SAM1_PCA-segPrior	0.858	0.804	0.863
SAM1_SAMAug	0.857	0.814	0.863
SAM1_OurSAMAug	0.857	**0.815**	0.872
SAM2_RG-segPrior	0.856	0.794	0.881
SAM2_RG-logits	0.849	0.795	0.871
SAM2_SV-segPrior	0.855	0.796	0.866
SAM2_PCA-segPrior	0.857	0.793	0.871
SAM2_SAMAug	0.856	0.793	0.873
SAM2_OurSAMAug	0.857	0.786	0.869

Taking into account the results detailed in Tables 2 and 3, we can draw the following conclusions.

1. Baseline(1_H) obtains the best performance only in two of the five datasets: ETIS and Ribs. For the other datasets, the best results are obtained by an approach proposed in this paper.

2. The performance of each method is not stable across the datasets. Baseline(1_H) performs very well for ETIS but poorly for ClinDB. SAM1_RG-logits achieves the highest mean performance in the Polyp datasets but in CVC-T, Ribs, and CAMO it performs worse than many other approaches.

3.2. Ensembles

In this section, we focus on evaluating the proposed ensembles, listed in Section 2.4. The numbers in brackets indicate the total numbers of models; the final output is obtained by the mean rule. We average the network outputs after the sigmoid function but before the threshold, so we average values in the range [0–1]. Using logits directly appears to slightly decrease performance; further tests are necessary to determine whether this behavior is dataset-dependent.

The results of our experiments are provided in Tables 4–6. Again, we report only a subset of the experiments we performed. Specifically, for HUGE($_X$), we include only the results obtained using HSNet, as it achieves the best performance. Each reported score is the average of five different runs.

Table 4. Ensembles: performance (Dice score, arithmetic mean over 5 runs) obtained in non-Polyp datasets. *H* refers to HSNet, *P* to PolypPVT; in cases where both P and H are indicated, both models are used in the same ensemble. The best scores in each column are marked in bold.

Method	Ribs	CAMO	Locust
Baseline(1_P)	0.843	0.772	0.876
Baseline(1_H)	0.863	0.811	0.882
Baseline(9_P)	0.846	0.783	0.882
Baseline(9_H)	0.863	0.812	0.876
Baseline($_{H\,+\,P}$)	**0.865**	0.811	0.886
AuxMix($_P$)	0.843	0.793	0.888
AuxMix($_H$)	0.864	**0.821**	0.885
AuxMix($_{H\,+\,P}$)	0.862	0.816	0.897
HUGE($_H$)	0.864	**0.821**	0.890
HUGE	0.862	0.818	**0.898**

Table 5. Ensembles: performance (Dice score, arithmetic mean over 5 runs) obtained in Polyp datasets. *H* refers to HSNet, *P* to PolypPVT; in cases where both P and H are indicated, both models are used in the same ensemble. The last column provides the average score over the datasets. The best scores in each column are marked in bold.

Method	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
Baseline(1_P)	0.906	0.923	0.899	0.783	0.770	0.856
Baseline(1_H)	0.903	0.917	0.910	0.819	0.816	0.873
Baseline(9_P)	0.907	0.933	0.908	0.829	0.805	0.876
Baseline(9_H)	0.904	0.927	0.915	0.820	0.828	0.879
Baseline($9_H + 9_P$)	0.907	0.927	0.911	0.832	0.838	0.883
AuxMix($_P$)	0.908	0.929	0.908	0.834	0.802	0.876
AuxMix($_H$)	0.908	0.938	0.919	0.843	0.832	0.888
AuxMix($_{H\,+\,P}$)	**0.909**	0.947	0.918	0.838	0.836	0.889
HUGE($_H$)	0.905	**0.949**	0.920	**0.846**	0.829	0.890
HUGE	0.907	**0.949**	**0.921**	0.843	**0.840**	**0.892**

Table 6. Ensembles: performance (Dice score) in Polyp-Gen dataset, to evaluate out-of-distribution generalization. The best scores in each column are marked in bold.

Method	C1	C2	C3	C4	C5	C6	Average
Baseline(1_P)	0.857	0.745	0.894	0.423	0.642	0.802	0.727
Baseline(1_H)	0.860	**0.765**	0.900	0.456	0.695	0.804	0.747
Baseline(9_P)	0.863	0.748	0.901	0.440	0.666	0.790	0.735
Baseline(9_H)	**0.873**	0.763	0.903	0.457	0.709	0.799	0.751
Baseline(9_P + 9_H)	0.871	0.760	0.905	0.454	0.708	0.804	0.750
AuxMix($_P$)	0.861	0.751	0.900	0.437	0.672	0.801	0.737
AuxMix($_H$)	0.870	0.757	0.905	**0.472**	0.700	**0.805**	**0.752**
AuxMix($_P$ + $_H$)	0.871	0.756	0.906	0.456	0.699	**0.805**	0.749
HUGE($_H$)	0.869	0.760	**0.908**	0.469	0.702	0.804	**0.752**
HUGE	0.871	0.759	0.907	0.458	**0.713**	0.804	**0.752**

Considering the results detailed in the tables, the following conclusions can be drawn.

1. Ensembling consistently improves the performance of standalone base networks, for both HSNet and PVT topologies.
2. While the ensemble approach proposed here yields a statistically significant (Wilcoxon signed-rank test, p-value = 0.05) improvement over the baseline for HSNet, this is not observed for PVT. Based on our results, we hypothesize that the method may have reached a performance plateau. For instance, in polyp datasets, an ensemble combining eight SAM-based methods (PCA-segPrior, RG-logits, RG-segPrior, OurSAMAug for both SAM and SAM 2) achieves the same mean performance as Baseline(9_X). Additionally, merging the two ensembles does not result in any performance gain, suggesting saturation.
3. AuxMix($_X$) offers the best trade-off between performance and computation time.
4. The results in Polyp-Gen are also noteworthy: the ensemble method appears to offer limited benefits, even for HSNet, and performance varies significantly across centers.
5. The best ensemble models proposed in this work achieve state-of-the-art performance among CNN- and Transformer-based methods. Even some very recent results based on the fusion of CNNs and Transformers [43] are not competitive with those of our ensemble-based models. Table 7 provides an extensive comparison with the literature: the only approach that reports better results than ours is based on Mamba [44] and its implementation has not yet been made public. Nevertheless, the ensemble strategy we introduce in this paper is also compatible with Mamba-based models.

Table 7. Ensembles: performance of our best models for Polyp datasets (HUGE, HUGE($_H$), and AuxMix($_H$): see Table 5) compared with the best models in the open literature. The models are sorted by average Dice score. The best scores in each column are marked in bold. Except for Lines 2 to 4, figures are taken from the literature. Where different papers provide different figures (example: FCBFormer), the highest figure is reported.

Model	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
SAM-Mamba [44]	**0.920**	0.942	0.924	**0.853**	**0.848**	**0.897**
HUGE	0.907	0.949	0.921	0.843	0.840	0.892
HUGE($_H$)	0.905	0.949	0.920	0.846	0.829	0.890
AuxMix($_H$)	0.908	0.938	0.919	0.843	0.832	0.888
Ens2 [1]	0.899	0.935	0.927	0.840	0.833	0.887
HiFiSeg [45]	0.905	0.942	0.933	0.826	0.822	0.886

Table 7. *Cont.*

Model	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
MGCBFormer [46]	0.913	**0.955**	0.931	0.807	0.819	0.885
PVT-CASCADE [47]	0.905	0.943	0.926	0.825	0.801	0.880
HSNet [19]	0.903	0.948	0.926	0.810	0.808	0.879
PFENet [48]	0.896	0.940	0.931	0.821	0.809	0.879
FCBFormer [49], from ref. [46]	0.911	0.949	0.922	0.809	0.799	0.878
CTNet [50]	0.908	0.936	0.917	0.813	0.810	0.877
MIA-Net [51]	0.900	0.942	0.926	0.816	0.800	0.877
ColonFormer-L [52]	0.906	0.932	0.924	0.811	0.801	0.875
DBMF [53]	0.919	0.933	0.932	0.803	0.790	0.875
Fu-TransHNet [43]	0.907	0.938	0.912	0.810	0.793	0.872
Polyp-PVT [54]	0.900	0.937	0.917	0.808	0.787	0.870
DS-TransUNet-L [55]	0.911	0.936	**0.935**	0.798	0.761	0.868
DFINet [56]	0.886	0.937	0.924	0.799	0.791	0.867
PSNet [57]	0.877	0.928	0.929	0.795	0.787	0.863
MEGANet (ResNet-34) [58]	0.887	0.930	0.911	0.781	0.789	0.860
BRNet [59]	0.898	0.921	0.918	0.795	0.760	0.858
SSformer [60], from ref. [45]	0.887	0.927	0.926	0.772	0.767	0.856
UACANet-L [61], from ref. [53]	0.913	0.929	0.915	0.753	0.769	0.856
CaraNet [62]	0.903	0.936	0.918	0.773	0.747	0.855
TransFuse-L* [63]	0.894	0.942	0.920	0.781	0.737	0.855
UACANet-L [61]	0.910	0.926	0.912	0.751	0.766	0.853
TransUNet, from ref. [63]	0.893	0.935	0.913	0.781	0.731	0.851
SANet [64], from ref. [53]	0.899	0.922	0.909	0.759	0.763	0.850
M^2SNet [65]	0.903	0.922	0.912	0.758	0.749	0.849
CFA-Net [66]	0.893	0.933	0.915	0.743	0.732	0.843
MSNet [67], from ref. [53]	0.873	0.926	0.915	0.756	0.733	0.841

In the CAMO dataset, many results are reported in https://paperswithcode.com/sota/camouflaged-object-segmentation-on-camo (accessed on 5 August 2025); however, not all follow the same protocol we used—specifically, training only on the CAMO training set. Methods that outperform AuxMix($_H$) are trained using a combination of COD10K and CAMO, making their results not directly comparable to ours. In our case, we achieve an MAE of 0.0546 and a Weighted F-Measure of 0.8345. Additionally, our previous ensemble Ens2 [1] achieved a lower Dice score on the CAMO dataset (0.812) than AuxMix($_H$).

Finally, in Figure 6, we present some examples where the proposed ensemble AuxMix($_H$) significantly improves upon the baseline performance. In many images, the results are comparable; however, in others, such as those shown in this figure, the difference between AuxMix($_H$) and the output of Baseline(1_H) is clearly noticeable. This is a promising indication for future work, where we aim to extend our ensemble construction methodology to other mainstream segmentators. Moreover, we plan to investigate how the AuxMix approach can be adapted to models like PVTv2, which currently do not exhibit performance gains over the baseline ensemble.

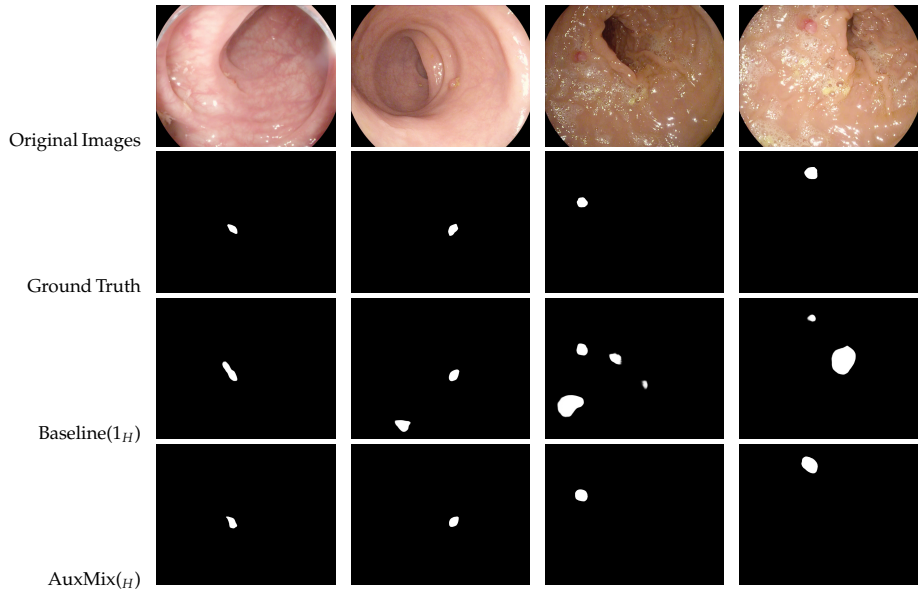


Figure 6. Examples of segmentation masks: Baseline(1_H) vs. ensemble AuxMix($_H$).

3.3. Ablation Studies

3.3.1. Contribution of Augmented Images

In this section, we analyze the behavior of the models during the inference phase. Each model was trained using both standard images (baseline) and images obtained through a specific augmentation technique (e.g., RG-logits, PCA-segPrior, etc.). As described in Section 2.3.1, the entropy-based strategy performs inference with both versions (baseline and augmented) of the same input image, produces two segmentation masks, and selects the one with lower entropy, which is considered more reliable. However, we do not explicitly assess how the models perform without this strategy. In principle, two separate evaluation scenarios can be compared.

1. The model receives only baseline images.
2. The model receives only augmented images.

It is important to note that adopting the entropy criterion was not our original proposal: we adopted the approach followed by SAMAug [17]. Our contribution focuses on reusing and extending this mechanism, rather than validating its baseline effectiveness.

We acknowledge that it would be interesting to investigate whether the model tends to rely predominantly on baseline or augmented images. While this analysis was not the primary goal of our study, we conducted exploratory checks to measure the frequency with which each image type is selected. Results show an approximately balanced distribution (about 50% baseline and 50% augmented), suggesting that both input types effectively contribute to the final prediction.

3.3.2. Effect of SAM vs. SAM 2

The proposed augmentation methods were applied using two different versions of the Segment Anything Model: SAM and SAM 2. In this section, we comment on how a specific version of the model affects the overall performance of the augmentation techniques we propose.

Since input augmentation experiments were carried out in parallel with both models, we have directly comparable results in Tables 2 and 3 and can assess whether, and to what extent, the increased capacity of the newer model (SAM 2) translates into tangible improvements. The numbers in Table 2 show that, in Polyp datasets, none of the two models is better than the other in all—or even the majority of—the datasets: the best

score is obtained with SAM in two cases, with SAM 2 in other two. The best average score is obtained with a data augmentation based on SAM. Instead, in non-Polyp datasets, the figures in Table 3 show that the best results are always obtained with SAM, albeit with different data augmentation strategies depending on the dataset.

3.3.3. Effects of Ensembling

The results presented in Tables 4–6 allow us to conduct another ablation study. Given our goal of proposing an effective ensemble method, the study follows a straightforward structure. First, we compare a single baseline model, namely Baseline($1_H/1_P$), against an ensemble of nine baseline models, i.e, Baseline($9_H/9_P$). Next, we evaluated the effectiveness of our proposed ensemble models AuxMix and HUGE by comparing them with Baseline($9_H/9_P$). This comparison highlights the advantages of combining multiple networks.

We also examined how the ensemble size in AuxMix influences performance. Starting from the base AuxMix($_H$) setup, we gradually increased the number of models in proportional steps. For the 36-model configuration, since 12 baseline-trained models were not available, two of them were randomly selected and duplicated. Each experiment was repeated three times, and the results reported in Table 8 correspond to the average. Note that the figures for AuxMix($_H$) differ slightly from those in Tables 5 and 7 because, for consistency, we performed a new batch of experiments and averaged three runs instead of five. The figures shows that average performance increases very slightly up to 27 models, at the expense of a massive increase in the complexity of the ensemble, then starts to decline. All in all, we believe that the choice of nine models is the most reasonable compromise between performance and complexity.

Table 8. Ablation studies: effects (arithmetic mean over 3 runs) on the Dice score of the number of models in AuxMix, expressed as the multiplier of models included. The base segmentation model used is HSNet, i.e., the baseline (×1) is AuxMix($_H$). The best scores in each column are marked in bold.

Multiplier	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
×1 (9 models)	**0.907**	**0.948**	0.920	0.841	0.825	0.888
×2 (18 models)	**0.907**	0.946	**0.921**	0.845	0.830	**0.890**
×3 (27 models)	**0.907**	0.945	0.920	**0.846**	**0.831**	**0.890**
×4 (36 models)	0.905	0.945	**0.921**	0.844	**0.831**	0.889

3.3.4. Evaluating Input Strategies

The training experiments were conducted using models trained on both types of images, namely the original images (baseline) and those generated through the different augmentation techniques. However, it is also interesting to compare these models with scenarios where training is restricted to a single type of input, i.e., only baseline images or only augmented images. To this end, we consider three main configurations.

1. **Baseline**: we use the HSNet model trained as described in the original paper, with the training set and the standard data augmentation proposed by the authors, and evaluate it on a test set composed exclusively of original images.
2. **Baseline (aug-only)**: models trained exclusively on augmented images and evaluated on the same test set. In this case, the entropy-based method is not applied; instead, evaluation is performed directly on the predictions produced when the model receives only augmented images as input.
3. **Combined Loss**: the approach proposed in the SAMAug paper, which trains with both baseline and augmented images, integrating them within a single loss function.

At test time, the entropy-based method is applied to combine the predictions from the two input types.

The comparison is presented in Table 9.

Table 9. Ablation studies: performance comparison of different training strategies, namely Baseline (original images), SAM1_SAMAug (only augmented images), and SAM1_SAMAug (combined original + augmented). Results are averaged over 3 runs using the same settings.

Method	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
Baseline	0.903	0.917	0.910	0.819	0.816	0.873
SAM1_SAMAug (aug-only)	0.828	0.903	0.896	0.769	0.726	0.824
SAM1_SAMAug (combined)	0.907	0.932	0.918	0.811	0.768	0.867

Beyond the specific results reported for SAM1_SAMAug, similar experiments were conducted with other methods in an earlier phase of the research, yielding consistent trends. In particular, training with only augmented images generally leads to lower performance compared to using only original images. Conversely, leveraging both original and augmented images during training, combined with the entropy-based selection at test time, results in performance improvements that surpass the baseline in several datasets. These findings motivated us to abandon the use of single-image training and focus on the combined approach, as it consistently provides superior and more robust performance.

3.3.5. Effect of Excluding Each Augmentation from AuxMix

AuxMix was defined as an ensemble composed of the models that, during the preliminary analysis, achieved the best performance and provided complementary contributions through different training strategies. To better understand the impact of each model within AuxMix($_H$), we performed a leave-one-out analysis, removing one model at a time from the ensemble and observing how the overall performance changed. The results are contained in Table 10. This experiment allows us to measure the unique contribution of each augmentation technique to the final accuracy and to assess whether certain augmentations play a more decisive role than others in improving global performance. It is worth noting that in this ablation, we did not include the case of removing the models trained on non-augmented images (previously referred to as the baseline model), since three such instances are present in AuxMix($_H$). Removing them would have led to an unfair comparison (ensemble with six instead of eight components).

Table 10. Ablation studies: performance comparison in Polyp datasets when removing one method from the AuxMix($_H$) ensemble. Each row indicates the excluded component, except for the first row presenting the original performance of the ensemble.

Method	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
AuxMix($_H$)	0.908	0.938	0.919	0.843	0.832	0.888
w/o SAM1_RG-logits	0.905	0.937	0.922	0.838	0.819	0.884
w/o SAM1_PCA-segPrior	0.903	0.934	0.922	0.842	0.821	0.884
w/o SAM1_SAMAug	0.901	0.940	0.925	0.840	0.821	0.885
w/o SAM2_RG-logits	0.904	0.937	0.921	0.841	0.822	0.885
w/o SAM2_PCA-segPrior	0.902	0.938	0.924	0.838	0.827	0.886
w/o SAM2_SAMAug	0.905	0.937	0.922	0.840	0.824	0.886

3.3.6. Evaluating Network Stability

The standard deviation of the Dice score for different augmentation strategies and models, reported in Table 11, is relatively low, confirming the stability of transformer-

based segmentation models. While this stability is desirable, for stand-alone approaches, it also reduces the diversity typically required for ensemble methods. It can be observed that ensemble AuxMix($_H$) exhibits a lower standard deviation compared to stand-alone approaches. Precisely because of this stability, we explored preprocessing variants to modify the input image provided to the segmentator in order to increase diversity and thus build a more effective ensemble.

Table 11. Standard deviation of the Dice performance metric in Polyp datasets computed over 5 networks for each method.

Method	CVC-T	ClinDB	Kvasir	ColDB	ETIS
Baseline(1_H)	±0.008	±0.009	±0.007	±0.006	±0.005
SAM1_RG-segPrior	±0.006	±0.010	±0.007	±0.005	±0.008
SAM1_RG-logits	±0.009	±0.008	±0.005	±0.007	±0.009
SAM1_SV-segPrior	±0.005	±0.007	±0.009	±0.004	±0.008
SAM1_PCA-segPrior	±0.007	±0.009	±0.008	±0.006	±0.004
SAM1_SAMAug	±0.010	±0.008	±0.005	±0.007	±0.009
SAM1_OurSAMAug	±0.004	±0.009	±0.006	±0.010	±0.007
SAM2_RG-segPrior	±0.009	±0.006	±0.010	±0.005	±0.007
SAM2_RG-logits	±0.008	±0.005	±0.009	±0.006	±0.010
SAM2_SV-segPrior	±0.007	±0.009	±0.006	±0.004	±0.008
SAM2_PCA-segPrior	±0.009	±0.006	±0.008	±0.007	±0.005
SAM2_SAMAug	±0.008	±0.007	±0.009	±0.006	±0.010
SAM2_OurSAMAug	±0.010	±0.008	±0.006	±0.004	±0.009
AuxMix($_H$)	±0.004	±0.003	±0.004	±0.004	±0.005

4. Conclusions

This work produced a number of interesting results, particularly through the incorporation of SAM-based augmentation techniques, which improve the process of learning segmentation patterns by incorporating SAM information directly into images. To exploit the strengths of these approaches, we introduced AuxMix, an ensemble trained with distinct SAM-based augmentation strategies. We evaluated the performance of different fusion strategies that combine different models to achieve state-of-the-art segmentation (SOTA) results. To ensure robust validation, we conducted tests with datasets from different application domains, ranging from polyp segmentation to the segmentation of locusts in natural environments. The main disadvantage of our approach is that it is an ensemble, so it is not suitable for applications where low-power computing devices are used. We do not report execution times because different components of our segmentation pipeline were executed on different machines with varying hardware. Consequently, reporting a total runtime would not be meaningful and could be misleading. However, our experiments show that the primary bottleneck of the pipeline lies in the execution of SAM models. A SAM1 inference required approximately 6 s per image on a consumer-grade NVIDIA GTX 1080 Ti GPU (2017, no tensor cores, about 10 TFLOPS of single-precision performance or 1/10th of the performance of a current-gen RTX 5080 GPU) and consumed around 1.4 GB of GPU memory. Compared to this step, the computational cost of all other components in the pipeline was negligible. For instance, HSNet inference required only 5 s for 100 images while using 1.3 GB of VRAM. Future work will explore pruning, quantization, low-rank factorization, and distillation to reduce the computing power required for our approach. We used only publicly available datasets, and all the computer code we developed is available on GitHub.

As future work, we aim to integrate the ensemble method proposed in this study with other network topologies to assess whether HSNet-like performance can be achieved across

other state-of-the-art segmentation approaches. Another aim is to integrate into AuxMix other ensemble strategies already proposed in the literature; for instance, approaches that vary the learning rate across networks, methods that apply different data augmentation schemes to each network, or techniques that employ distinct optimization strategies for each model. The common goal of these approaches is to encourage each network to converge to a different minimum, thereby increasing the diversity among the ensemble members. Such diversity is expected to make their combination more effective, ultimately improving the performance of the ensemble. Additionally, we plan to expand the evaluation of our augmentations—including the blue-channel heuristic—and ensemble strategies to a broader set of datasets to investigate the generalizability of the observed results.

Author Contributions: Conceptualization, L.N.; methodology, L.C., F.C., C.F. and L.N.; software, F.C., L.C., C.F. and L.N.; validation, C.F. and L.N.; formal analysis, L.C., F.C., C.F. and L.N.; writing—original draft preparation, L.C., F.C., C.F. and L.N.; writing—review and editing, L.C., F.C., C.F. and L.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: We used only publicly available datasets: links to such datasets are provided in Section 2.5. All the computer code we developed is available on GitHub (https://github.com/LorisNanni/Exploring-SAM-Augmented-Ensembles, accessed on 5 August 2025) (the repo will be populated upon publication of this paper).

Acknowledgments: This paper contains a substantially expanded version of the work carried out by Lorenzo Carisi [68] and Francesco Chiereghin [69] for their Master's theses. We acknowledge the support that NVIDIA provided us through the GPU Grant Program.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A. Experimental Details and Augmentations

Appendix A.1. SAM Mask Generation Parameters

For all experiments, we used the following parameters for the SamAutomaticMaskGenerator, without performing any dataset-specific tuning. The first three parameters for thresholds deviate from the default settings, while the remaining parameters retain their default values. The generator was configured as follows:

- points per side = 32;
- points per batch = 64;
- pred iou thresh = 0.5;
- stability score thresh = 0.95;
- stability score offset = 1.0;
- box nms thresh = 0.5;
- crop n layers = 0;
- crop nms thresh = 0.5;
- crop overlap ratio = 512/1500;
- crop n points downscale factor = 1;
- point grids = none;
- min mask region area = 0;
- output mode = "binary mask";

as specified in https://github.com/facebookresearch/segment-anything/blob/main/segment_anything/automatic_mask_generator.py (accessed on 5 August 2025).

Appendix A.2. Image Augmentations

For HSV augmentations, conversion between RGB and HSV is carried out using the `skimage.color.rgb2hsv` function, with hue, saturation, and value remaining in the $[0, 1]$ range. No inverse transformation is applied after the augmentation.

PCA augmentations are implemented using `sklearn.decomposition.PCA`, fitted on each individual image (i.e., per-image fit). Whitening is not applied, no random seed is set, and no global fit across the training set is performed.

Appendix B. Channel Visual Comparison

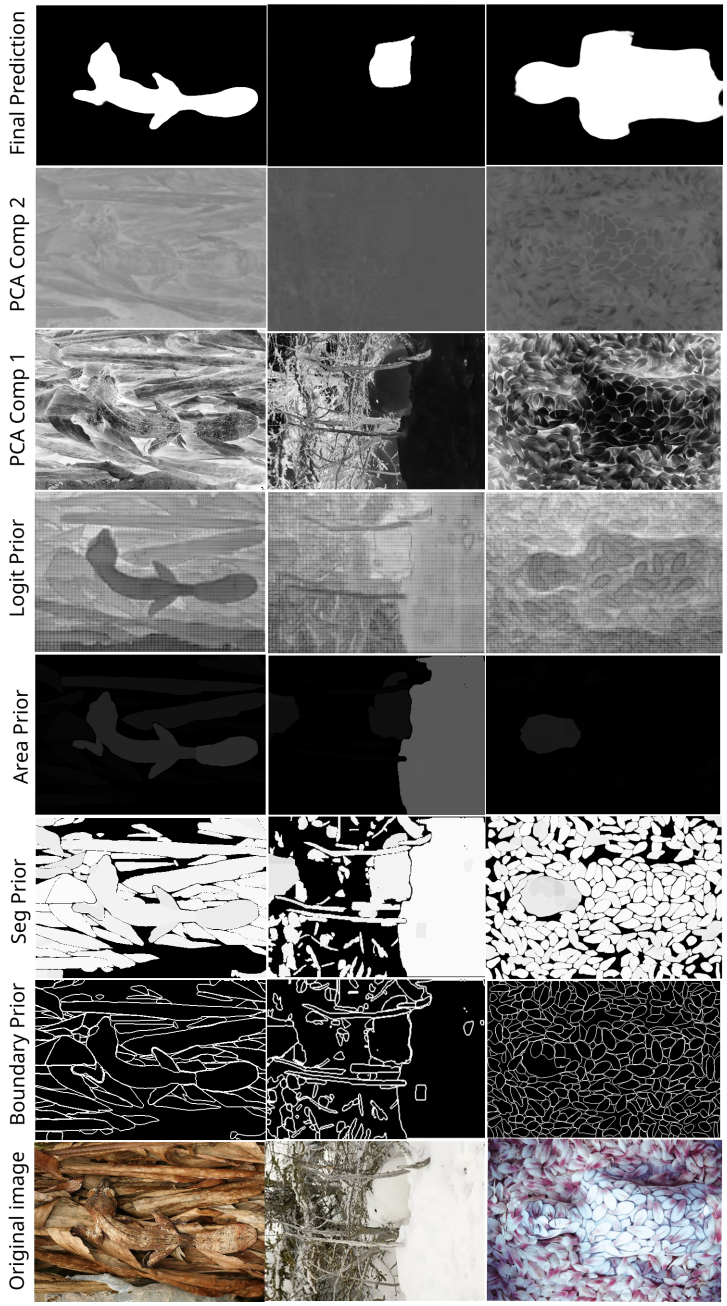


Figure A1. Visual comparison of model's output alongside different SAM's priors, illustrating how each added channel affects segmentation predictions.

References

1. Nanni, L.; Lumini, A.; Fantozzi, C. Exploring the Potential of Ensembles of Deep Learning Networks for Image Segmentation. *Information* **2023**, *14*, 657. [CrossRef]
2. Elhassan, M.A.; Zhou, C.; Khan, A.; Benabid, A.; Adam, A.B.; Mehmood, A.; Wambugu, N. Real-time semantic segmentation for autonomous driving: A review of CNNs, Transformers, and Beyond. *J. King Saud Univ. Comput. Inf. Sci.* **2024**, *36*, 102226. [CrossRef]
3. Qureshi, I.; Yan, J.; Abbas, Q.; Shaheed, K.; Riaz, A.B.; Wahid, A.; Khan, M.W.J.; Szczuko, P. Medical image segmentation using deep semantic-based methods: A review of techniques, applications and emerging trends. *Inf. Fusion* **2023**, *90*, 316–352. [CrossRef]
4. Hurtado, J.V.; Valada, A. Semantic Scene Segmentation for Robotics. *arXiv* **2024**, arXiv:2401.07589.
5. Farhan Audianto, M.; Dwi Sulistiyo, M.; Rachmawati, E.; Hadiyoso, S. Fashion Parsing and Identification for E-Commerce Using Semantic Segmentation. In Proceedings of the 12th International Conference on Information and Communication Technology (ICoICT), Bandung, Indonesia, 7–8 August 2024; pp. 566–571. [CrossRef]
6. Wang, S.; Mu, X.; Yang, D.; He, H.; Zhao, P. Attention Guided Encoder-Decoder Network With Multi-Scale Context Aggregation for Land Cover Segmentation. *IEEE Access* **2020**, *8*, 215299–215309. [CrossRef]
7. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440. [CrossRef]
8. Siddique, N.; Paheding, S.; Elkin, C.P.; Devabhaktuni, V. U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications. *IEEE Access* **2021**, *9*, 82031–82057. [CrossRef]
9. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. [CrossRef]
10. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef]
11. Hao, S.; Zhou, Y.; Guo, Y. A Brief Survey on Semantic Segmentation with Deep Learning. *Neurocomputing* **2020**, *406*, 302–321. [CrossRef]
12. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale. In Proceedings of the 9th International Conference on Learning Representations (ICLR), Virtual, 3–7 May 2021.
13. Wang, W.; Xie, E.; Li, X.; Fan, D.P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; Shao, L. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 548–558. [CrossRef]
14. Mohammed, A.; Kora, R. A comprehensive review on ensemble deep learning: Opportunities and challenges. *J. King Saud Univ.-Comput. Inf. Sci.* **2023**, *35*, 757–774. [CrossRef]
15. Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.Y.; et al. Segment Anything. *arXiv* **2023**, arXiv:2304.02643.
16. Ravi, N.; Gabeur, V.; Hu, Y.T.; Hu, R.; Ryali, C.; Ma, T.; Khedr, H.; Rädle, R.; Rolland, C.; Gustafson, L.; et al. SAM 2: Segment Anything in Images and Videos. *arXiv* **2024**, arXiv:2408.00714
17. Zhang, Y.; Zhou, T.; Wang, S.; Liang, P.; Zhang, Y.; Chen, D.Z. Input Augmentation with SAM: Boosting Medical Image Segmentation with Segmentation Foundation Model. In *MICCAI 2023 Workshops, Proceedings of the Medical Image Computing and Computer Assisted Intervention, Vancouver, BC, Canada, 8–12 October 2023*; Celebi, M.E., Salekin, M.S., Kim, H., Albarqouni, S., Barata, C., Halpern, A., Tschandl, P., Combalia, M., Liu, Y., Zamzmi, G., et al., Eds.; Springer: Cham, Switzerland, 2023; pp. 129–139. [CrossRef]
18. Wang, W.; Xie, E.; Li, X.; Fan, D.P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; Shao, L. PVT v2: Improved baselines with Pyramid Vision Transformer. *Comput. Vis. Media* **2022**, *8*, 415–424. [CrossRef]
19. Zhang, W.; Fu, C.; Zheng, Y.; Zhang, F.; Zhao, Y.; Sham, C.W. HSNet: A hybrid semantic network for polyp segmentation. *Comput. Biol. Med.* **2022**, *150*, 106173. [CrossRef]
20. Li, Y.; Mao, H.; Girshick, R.; He, K. Exploring Plain Vision Transformer Backbones for Object Detection. In *Computer Vision—ECCV 2022, Proceedings of the 17th European Conference, Tel Aviv, Israel, 23–27 October 2022*; Proceedings, Part IX; Springer: Berlin/Heidelberg, Germany, 2022; pp. 280–296. [CrossRef]
21. Tancik, M.; Srinivasan, P.; Mildenhall, B.; Fridovich-Keil, S.; Raghavan, N.; Singhal, U.; Ramamoorthi, R.; Barron, J.; Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 7537–7547.
22. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning transferable visual models from natural language supervision. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 18–24 July 2021; pp. 8748–8763.

23. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.u.; Polosukhin, I. Attention is All you Need. In *Proceedings of the Advances in Neural Information Processing Systems*; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
24. Zhang, C.; Liu, L.; Cui, Y.; Huang, G.; Lin, W.; Yang, Y.; Hu, Y. A Comprehensive Survey on Segment Anything Model for Vision and Beyond. *arXiv* **2023**, arXiv:2305.08196.
25. Cheng, J.; Ye, J.; Deng, Z.; Chen, J.; Li, T.; Wang, H.; Su, Y.; Huang, Z.; Chen, J.; Sun, L.J.H.; et al. SAM-Med2D. *arXiv* **2023**, arXiv:2308.16184.
26. Lin, X.; Xiang, Y.; Zhang, L.; Yang, X.; Yan, Z.; Yu, L. SAMUS: Adapting Segment Anything Model for Clinically-Friendly and Generalizable Ultrasound Image Segmentation. *arXiv* **2023**, arXiv:2309.06824.
27. Ma, J.; He, Y.; Li, F.; Han, L.; You, C.; Wang, B. Segment Anything in Medical Images. *arXiv* **2023**, arXiv:2304.12306.
28. Zhang, K.; Liu, D. Customized Segment Anything Model for Medical Image Segmentation. *arXiv* **2023**, arXiv:2304.13785.
29. Wu, J.; Ji, W.; Liu, Y.; Fu, H.; Xu, M.; Xu, Y.; Jin, Y. Medical SAM Adapter: Adapting Segment Anything Model for Medical Image Segmentation. *arXiv* **2023**, arXiv:2304.12620.
30. Ryali, C.; Hu, Y.T.; Bolya, D.; Wei, C.; Fan, H.; Huang, P.Y.; Aggarwal, V.; Chowdhury, A.; Poursaeed, O.; Hoffman, J.; et al. Hiera: A hierarchical vision transformer without the bells-and-whistles. In Proceedings of the International Conference on Machine Learning, PMLR, Honolulu, HI, USA, 23–29 July 2023; pp. 29441–29454.
31. Curcio, C.A.; Allen, K.A.; Sloan, K.R.; Lerea, C.L.; Hurley, J.B.; Klock, I.B.; Milam, A.H. Distribution and morphology of human cone photoreceptors stained with anti-blue opsin. *J. Comp. Neurol.* **1991**, *312*, 610–624. [CrossRef]
32. Chen, C.; Huang, W.; Zhang, L.; Mow, W.H. Robust and Unobtrusive Display-to-Camera Communications via Blue Channel Embedding. *IEEE Trans. Image Process.* **2019**, *28*, 156–169. [CrossRef]
33. Emre Celebi, M.; Wen, Q.; Hwang, S.; Iyatomi, H.; Schaefer, G. Lesion border detection in dermoscopy images using ensembles of thresholding methods. *Skin Res. Technol.* **2013**, *19*, e252–e258. [CrossRef]
34. Vázquez, D.; Bernal, J.; Sánchez, F.J.; Fernández-Esparrach, G.; López, A.M.; Romero, A.; Drozdzal, M.; Courville, A. A Benchmark for Endoluminal Scene Segmentation of Colonoscopy Images. *arXiv* **2016**, arXiv:1612.00799.
35. Bernal, J.; Sánchez, F.J.; Fernández-Esparrach, G.; Gil, D.; Rodríguez, C.; Vilariño, F. WM-DOVA maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians. *Comput. Med. Imaging Graph.* **2015**, *43*, 99–111. [CrossRef]
36. Kvasir-SEG: A Segmented Polyp Dataset. In Proceedings of the International Conference on Multimedia Modeling, Daejeon, South Korea, 5–8 January 2020; pp. 30, 820. [CrossRef]
37. Bernal, J.; Sánchez, J.; Vilariño, F. Towards automatic polyp detection with a polyp appearance model. *Pattern Recognit.* **2012**, *45*, 3166–3182. [CrossRef].
38. Silva, J.; Histace, A.; Romain, O.; Dray, X.; Granado, B. Toward embedded detection of polyps in WCE images for early diagnosis of colorectal cancer. *Int. J. Comput. Assist. Radiol. Surg.* **2014**, *9*, 283–293. [CrossRef].
39. Ali, S.; Jha, D.; Ghatwary, N.; Realdon, S.; Cannizzaro, R.; Salem, O.E.; Lamarque, D.; Daul, C.; Riegler, M.A.; Anonsen, K.V.; et al. A multi-centre polyp detection and segmentation dataset for generalisability assessment. *Sci. Data* **2023**, *10*, 75.
40. Nguyen, H.C.; Le, T.T.; Pham, H.H.; Nguyen, H.Q. VinDr-RibCXR: A Benchmark Dataset for Automatic Segmentation and Labeling of Individual Ribs on Chest X-rays. *arXiv* **2021**, arXiv:2107.01327.
41. Le, T.N.; Nguyen, T.V.; Nie, Z.; Tran, M.T.; Sugimoto, A. Anabranch network for camouflaged object segmentation. *Comput. Vis. Image Underst.* **2019**, *184*, 45–56. [CrossRef].
42. Liu, L.; Liu, M.; Meng, K.; Yang, L.; Zhao, M.; Mei, S. Camouflaged locust segmentation based on PraNet. *Comput. Electron. Agric.* **2022**, *198*, 107061. [CrossRef].
43. Luo, C.; Wang, Y.; Deng, Z.; Lou, Q.; Zhao, Z.; Ge, Y.; Hu, S. Colonic polyp segmentation based on transformer-convolutional neural networks fusion. *Pattern Recognit.* **2026**, *170*, 112116. [CrossRef].
44. Dutta, T.K.; Majhi, S.; Nayak, D.R.; Jha, D. SAM-Mamba: Mamba Guided SAM Architecture for Generalized Zero-Shot Polyp Segmentation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Tucson, AZ, USA, 26 February–6 March 2025; pp. 4655–4664. [CrossRef].
45. Ren, J.; Zhang, X.; Zhang, L. HiFiSeg: High-Frequency Information Enhanced Polyp Segmentation With Global-Local Vision Transformer. *IEEE Access* **2025**, *13*, 38704–38713. [CrossRef].
46. Xia, Y.; Yun, H.; Liu, Y.; Luan, J.; Li, M. MGCBFormer: The multiscale grid-prior and class-inter boundary-aware transformer for polyp segmentation. *Comput. Biol. Med.* **2023**, *167*, 107600. [CrossRef].
47. Rahman, M.M.; Marculescu, R. Medical Image Segmentation via Cascaded Attention Decoding. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 4–6 January 2023; pp. 6211–6220. [CrossRef].
48. Yue, G.; Xiao, H.; Zhou, T.; Tan, S.; Liu, Y.; Yan, W. Progressive Feature Enhancement Network for Automated Colorectal Polyp Segmentation. *IEEE Trans. Autom. Sci. Eng.* **2025**, *22*, 5792–5803. [CrossRef]..

49. Sanderson, E.; Matuszewski, B.J. FCN-transformer feature fusion for polyp segmentation. In *Medical Image Understanding and Analysis, Proceedings of the 26th Annual Conference, Cambridge, UK, 27–29 July 2022*; Springer: Cham, Switzerland, 2022; pp. 892–907. [CrossRef].
50. Xiao, B.; Hu, J.; Li, W.; Pun, C.M.; Bi, X. CTNet: Contrastive Transformer Network for Polyp Segmentation. *IEEE Trans. Cybern.* **2024**, *54*, 5040–5053. [CrossRef].
51. Li, W.; Zhao, Y.; Li, F.; Wang, L. MIA-Net: Multi-information aggregation network combining transformers and convolutional feature learning for polyp segmentation. *Knowl.-Based Syst.* **2022**, *247*, 108824. [CrossRef].
52. Duc, N.T.; Oanh, N.T.; Thuy, N.T.; Triet, T.M.; Dinh, V.S. ColonFormer: An Efficient Transformer Based Method for Colon Polyp Segmentation. *IEEE Access* **2022**, *10*, 80575–80586. [CrossRef].
53. Liu, F.; Hua, Z.; Li, J.; Fan, L. DBMF: Dual Branch Multiscale Feature Fusion Network for polyp segmentation. *Comput. Biol. Med.* **2022**, *151*, 106304. [CrossRef].
54. Dong, B.; Wang, W.; Fan, D.P.; Li, J.; Fu, H.; Shao, L. Polyp-PVT: Polyp Segmentation with Pyramid Vision Transformers. *CAAI Artif. Intell. Res.* **2023**, *2*, 9150015. [CrossRef].
55. Lin, A.; Chen, B.; Xu, J.; Zhang, Z.; Lu, G.; Zhang, D. DS-TransUNet: Dual Swin Transformer U-Net for Medical Image Segmentation. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 4005615. [CrossRef].
56. Yue, G.; Li, Y.; Wu, S.; Jiang, B.; Zhou, T.; Yan, W.; Lin, H.; Wang, T. Dual-Domain Feature Interaction Network for Automatic Colorectal Polyp Segmentation. *IEEE Trans. Instrum. Meas.* **2024**, *73*, 5034012. [CrossRef].
57. Lewis, J.; Cha, Y.J.; Kim, J. Dual encoder–decoder-based deep polyp segmentation network for colonoscopy images. *Sci. Rep.* **2023**, *13*, 1183.
58. Bui, N.T.; Hoang, D.H.; Nguyen, Q.T.; Tran, M.T.; Le, N. MEGANet: Multi-Scale Edge-Guided Attention Network for Weak Boundary Polyp Segmentation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 1–6 January 2024; pp. 7970–7979. [CrossRef].
59. Yue, G.; Li, Y.; Jiang, W.; Zhou, W.; Zhou, T. Boundary Refinement Network for Colorectal Polyp Segmentation in Colonoscopy Images. *IEEE Signal Process. Lett.* **2024**, *31*, 954–958. [CrossRef].
60. Shi, W.; Xu, J.; Gao, P. SSformer: A lightweight transformer for semantic segmentation. In Proceedings of the IEEE 24th International Workshop on Multimedia Signal Processing (MMSP), Shanghai, China, 26–28 September 2022; IEEE: New York, NY, USA, 2022; pp. 1–5. [CrossRef].
61. Kim, T.; Lee, H.; Kim, D. UACANet: Uncertainty Augmented Context Attention for Polyp Segmentation. In Proceedings of the 29th ACM International Conference on Multimedia—MM '21, New York, NY, USA, 20–24 October 2021; pp. 2167–2175. [CrossRef].
62. Lou, A.; Guan, S.; Loew, M.H. CaraNet: Context axial reverse attention network for segmentation of small medical objects. *J. Med. Imaging* **2023**, *10*, 014005. [CrossRef].
63. Zhang, Y.; Liu, H.; Hu, Q. TransFuse: Fusing Transformers and CNNs for Medical Image Segmentation. In *Medical Image Computing and Computer Assisted Intervention—MICCAI 2021, 24th International Conference, Strasbourg, France, 27 September–1 October 2021*; de Bruijne, M., Cattin, P.C., Cotin, S., Padoy, N., Speidel, S., Zheng, Y., Essert, C., Eds.; Springer: Cham, Switzerland, 2021; pp. 14–24. [CrossRef].
64. Wei, J.; Hu, Y.; Zhang, R.; Li, Z.; Zhou, S.K.; Cui, S. Shallow Attention Network for Polyp Segmentation. In *Medical Image Computing and Computer Assisted Intervention—MICCAI 2021, 24th International Conference, Strasbourg, France, 27 September–1 October 2021*; Springer: Cham, Switzerland, 2021; Volume 12901, pp. 699–708. [CrossRef].
65. Zhao, X.; Jia, H.; Pang, Y.; Lv, L.; Tian, F.; Zhang, L.; Sun, W.; Lu, H. M^2SNet: Multi-scale in Multi-scale Subtraction Network for Medical Image Segmentation. *arXiv* **2023**, arXiv:2303.10894.
66. Zhou, T.; Zhou, Y.; He, K.; Gong, C.; Yang, J.; Fu, H.; Shen, D. Cross-level Feature Aggregation Network for Polyp Segmentation. *Pattern Recognit.* **2023**, *140*, 109555. [CrossRef].
67. Zhao, X.; Zhang, L.; Lu, H. Automatic Polyp Segmentation via Multi-scale Subtraction Network. In *Proceedings of the Medical Image Computing and Computer Assisted Intervention—MICCAI 2021, Strasbourg, France, 27 September–1 October 2021*; de Bruijne, M., Cattin, P.C., Cotin, S., Padoy, N., Speidel, S., Zheng, Y., Essert, C., Eds.; Springer: Cham, Switzerland, 2021; pp. 120–130. [CrossRef].

68. Carisi, L. Augmentation and Ensembles: Improving Medical Image Segmentation with SAM and Deep Networks. Master's Thesis, University of Padova, Padova, Italy, 2024. Available online : https://hdl.handle.net/20.500.12608/78065 (accessed on 5 August 2025).
69. Chiereghin, F. Exploring SAM-Augmented Ensembles For Image Segmentation Tasks. Master's Thesis, University of Padova, Padova, Italy, 2025. Available online: https://hdl.handle.net/20.500.12608/84251 (accessed on 5 August 2025).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.