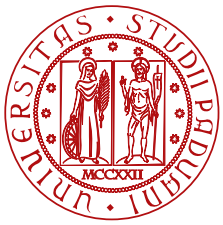

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DEPARTMENT OF INFORMATION ENGINEERING

BACHELOR'S DEGREE IN COMPUTER ENGINEERING

Medical Image Segmentation with Selective State Space Models

Thesis Supervisor

Prof. Carlo Fantozzi

Graduating Student

Alberto Levorato

ACADEMIC YEAR 2024-2025

Graduation Date 24/07/2025

To my family and the friends that I met along the way

Abstract

Medical image segmentation is a critical task in clinical diagnostics, requiring high precision and reliability. Traditional manual segmentation is labor intensive, prompting the need for automated approaches. While Transformer-based architectures have dominated recent advances, their limitations in scalability and computational efficiency have motivated exploration of alternative models. This paper investigates Mamba, a novel class of Selective State Space Models, as a promising alternative for medical image segmentation. Two Mamba-based architectures U-Mamba and Swin-UMamba have been evaluated across multiple polyp segmentation datasets, with custom data augmentation techniques applied to improve generalization and mitigate overfitting. Results demonstrate that Mamba models offer competitive performance, with Swin-UMamba approaching state-of-the-art accuracy. Thanks to their efficiency, adaptability, and compact size, these models are well-suited for emerging medical applications. Overall, the findings suggest that Mamba architectures represent a promising direction for future research in medical computer vision.

Sommario

La segmentazione delle immagini mediche è un compito fondamentale nella diagnostica clinica, che richiede alta precisione e affidabilità. La segmentazione manuale è un processo lavorativamente intenso, rendendo necessarie soluzioni automatizzate. Sebbene i recenti progressi nel campo siano stati dominati da architetture basate su Transformer, le loro limitazioni in termini di scalabilità ed efficienza computazionale hanno spinto alla ricerca di modelli alternativi. Questo studio analizza Mamba, una nuova classe di modelli Selective State Space, come alternativa promettente per la segmentazione di immagini mediche. Sono state valutate U-Mamba e Swin-UMamba, due architetture Mamba, su diversi dataset di segmentazione di polipi, applicando tecniche dedicate di data augmentation per migliorare la generalizzazione e ridurre l'overfitting. I risultati dimostrano che i modelli Mamba offrono prestazioni competitive, con Swin-UMamba che si avvicina allo stato dell'arte. Grazie alla loro efficienza, adattabilità e dimensioni compatte, questi modelli risultano adatti a nuove applicazioni in ambito medico. I risultati suggeriscono che le architetture Mamba rappresentano una direzione promettente per la ricerca futura nella computer vision applicata alla medicina.

Contents

1 Introduction **1**

1.1 State Space model 2

1.2 Structured State Space method 4

1.3 Selective State Space model 4

2 Mamba Models **7**

2.1 U-Mamba 8

2.2 Swin-UMamba 8

3 Datasets **11**

3.1 AbdomenMR 11

3.2 Kvasir-SEG 12

3.3 HarDNet-MSEG 13

4 Experiments **14**

4.1 Installation and validation 14

4.2 Training and evaluation 15

4.2.1 Kvasir-SEG 15

4.2.2 HarDNet-MSEG 16

4.3 Data augmentation 18

4.3.1 Data Augmentation 1 (DA1) 18

4.3.2 Data Augmentation 2 (DA2) 18

Conclusions **20**

Chapter 1

Introduction

Image segmentation is a Computer Vision task that divides an image into groups of pixels to detect objects. Medical image segmentation plays a crucial role in clinical applications, such as X-rays, Magnetic Resonance Imaging (MRI), and endoscopy, helping with disease diagnosis, treatment planning support, and monitoring of disease progression. A typical segmentation process is a manual task that requires an experienced doctor and it is both labor-intensive and time-consuming.

Deep learning research has recently made significant advances with medical image segmentation, modern Foundation Models (FMs) architectures are predominantly based on a single type of sequence model, the Transformer [29] and its core attention layer [2]. Self-attention is efficient, thanks to its ability to route information densely within a context window, allowing it to model complex data. However, this property leads to a modeling limitation for a finite window and quadratic complexity on its size.

Structured State Space Sequence models (S4) [6], based on State space models (SSMs) [7], emerged as a promising architecture for sequence modeling, S4 models can be interpreted as a combination of Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs) and classical State space models from control theory [12]. These models achieve linear or near-linear scaling in sequence length, and they have principled mechanisms for modeling long-range dependencies. Mamba [5] class of Selective state space models improves on S4 and SSMs on several axes to achieve the modeling power of Transformers while scaling linearly in the sequence length.

This thesis aims to investigate the architecture behind Selective state space models and experiment with two Mamba implementations in the domain of medical image segmentation, using datasets (not originally tested for the models) and self-designed data augmentations to improve accuracy. All the scripts mentioned in the rest of the document can be found in [13].

1.1 State Space model

A State space model is defined by the following system of continuous equations:

$$x'(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

It maps $u(t)$, a one-dimensional input signal, to $x(t)$, a N-dimensional latent state, and then projects to $y(t)$, a one-dimensional output signal. The SSM is used as a representation for the deep sequence model, where A, B, C, D are parameters learned by gradient descent. $Du(t) = 0$ is easy to compute as it can be viewed as a skip connection. The structure of the three significant matrices is as follows: $A \in \mathbb{R}^{N\times N}$, $B \in \mathbb{R}^{N\times 1}$, and $C \in \mathbb{R}^{1\times N}$.

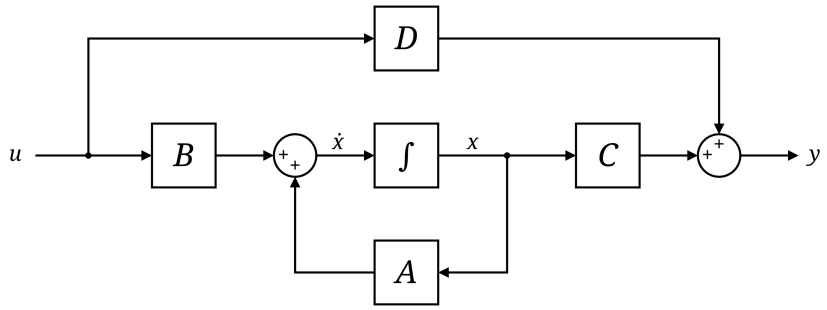

Figure 1.1: Block diagram representation of the State space model. Source: State-space representation (wikipedia)

Two important properties of SSMs are linear transformation of the input and time invariance, that can be resumed in Linear Time Invariance (LTI), which means that A, B, C, D are constant matrices for all time steps. LTI systems are described by linear recurrence and convolution.

Addressing Long-Range Dependencies: HiPPO

The basic state space model performs very poorly in practice. One intuitive explanation that was given is that linear first-order ordinary differential equations solve to an exponential function, and thus may suffer from gradients scaling exponentially in the sequence length. HiPPO theory [20] of continuous-time memorization has been developed to address the problem, HiPPO class matrices $A \in \mathbb{R}^{N\times N}$, incorporated with SSM, allow the state $x(t)$ to memorize the history of the input $u(t)$. The HiPPO matrix is defined as:

$$A_{nk} = \begin{cases} (2n+1)^{1/2}(2k+1)^{1/2} & \text{if } n > k \\ n+1 & \text{if } n = k \\ 0 & \text{if } n < k \end{cases}$$

Discretization: The Recurrent Representation

As we work with discrete data input sequences, the input signal is conceptually discretized with a step of size Δ: the sampling formula looks like $u_k(t) = u(k\Delta)$.

The SSM is discretized with the bilinear method [28], that converts the state matrices into the following approximations:

$$\begin{aligned}\overline{A} &= (I - \Delta/2 \cdot A)^{-1} (I + \Delta/2 \cdot A) \\ \overline{B} &= (I - \Delta/2 \cdot A)^{-1} \Delta B \\ \overline{C} &= C\end{aligned}$$

The discrete SSM is then represented by a sequence-to-sequence mapping from u_k to y_k:

$$\begin{aligned}x_k &= \overline{A} x_{k-1} + \overline{B} u_k \\ y_k &= \overline{C} x_k\end{aligned}$$

the state equation is now expressed as a recurrence in x_k, allowing the discrete SSM to be computed like an RNN. In practice, $x_k \in \mathbb{R}^N$ can be viewed as a hidden state with transition matrix $\overline{A}$.

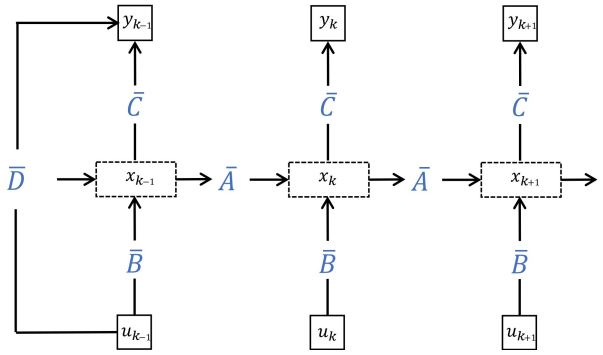

Figure 1.2: from [7]. Recurrent representation, inference can be performed efficiently by computing the layer timewise (i.e., one vertical slice at a time: (u_t, x_t, y_t), $(u_{t+1}, x_{t+1}, y_{t+1})$, ...), by unrolling the linear recurrence.

Training: The Convolutional Representation

The recurrent representation is sequential, hence it is not practical for training on modern hardware. It can be written as a discrete convolution, that can be vectorized into $y = \overline{K} * u$ for which

$$y_k = \overline{CA^kB}u_0 + \overline{CA^{k-1}B}u_1 + \cdots + \overline{CAB}u_{k-1} + \overline{CB}u_k$$

where $\overline{K} \in \mathbb{R}^L = (\overline{CB}, \overline{CAB}, ..., \overline{CA^{L-1}B})$ is the *convolution kernel*, a filter that is the size of the entire input sequence.

Given that $\overline{K}$ is known, computing the non-circular convolutional formula is very efficient with FFTs as the calculations can be parallelized on GPUs.

1.2 Structured State Space method

The Structured state spaces method proposes a parametrization on the SSM to efficiently compute the continuous (A, B, C), recurrent $(\overline{A}, \overline{B}, \overline{C})$ and convolutional $\overline{K}$ representations.

Parametrization

The fundamental computing bottleneck for discrete SSM $(\overline{A}, \overline{B}, \overline{C})$ is the repeated multiplication of $\overline{A}$ matrices. Firstly, S4 [6] proposes a diagonalization technique that fails due to numerical problems, but suggests that the conjugation is an equivalence relation on SSMs, with just a change of basis by V: it is equivalent to compute the SSM for matrices $(V^{-1}AV, V^{-1}B, CV)$. The HiPPO matrix is not defined as a *normal matrix*, so it is not diagonalizable by a perfectly conditioned matrix. Although HiPPO is not normal, it can be decomposed as a *normal plus low-rank* (NPLR) matrix. Given A, initialized as a HiPPO matrix, it can be conjugated into *diagonal plus low-rank* (DPLR) matrix $A = \Lambda - PQ^*$ for some diagonal Δ and vectors $P, Q, B, C \in \mathbb{C}^{N\times 1}$ (note that it has been switched from $\mathbb{R}$ to $\mathbb{C}$).

Finally, using S4 [6] proposed techniques, computing one step of the recurrence for any step size Δ can be done in $O(N)$ time, where N is the state size. Computing the SSM convolution kernel $\overline{K}$ requires approximately $O(N + L)$ operations and $O(N + L)$ space, where L is the length of the sequence.

1.3 Selective State Space model

Starting with the observation that a fundamental problem of sequence modeling is compressing context into a smaller state, Selective SSM proposes to use Selection as a means of compression. Effective models such as Transformers [29], which infer accurate and detailed responses, suffer from inefficiency because all information must be stored without compression. However, finite-state models sacrifice accuracy to achieve higher efficiency, meaning that data compression is closely related to effectiveness. The context-aware ability to focus on or filter out inputs into a sequential state, also known as selectivity, controls how information propagates or interacts along the sequence dimension.

Selective Copy

The Selective copy is based on the standard copying task [19], which involves constant spacing between the input and output elements and is solved by LTI models. Instead, selective copy has random spacing between inputs and requires time-varying models that can selectively remember or ignore inputs depending on their content (1.3).

Induction Heads

The Induction Heads task is an associative recall that retrieves an answer based on context, a key feature of the in-context learning abilities of LLMs. It requires context-aware reasoning to know when to produce the correct output in the appropriate context (1.3).

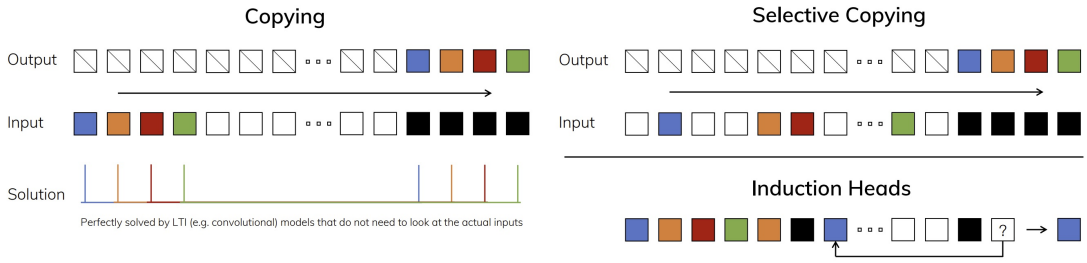

Figure 1.3: from [5]. Relevant tokens - colored / Irrelevant tokens - white / Appropriate context - black. (Left) The standard version of the Copying task. (Right Top) The Selective Copying task. (Right Bottom) The Induction Heads task.

The proposed method to incorporate a selection mechanism into S4s is to make the parameters that influence sequential interactions (e.g., the recurrent dynamics of an RNN or the convolution kernel of a CNN) dependent on the input. Algorithm 2 shows the implementation of selection, where parameters Δ, B, C are changed to functions of the input: they have a length dimension L, so now the model is time-varying and is no longer equivalent to convolutions.

Algorithm 1 SSM (S4)

Input: $x : (\mathsf{B}, \mathsf{L}, \mathsf{D})$
Output: $y : (\mathsf{B}, \mathsf{L}, \mathsf{D})$
$\boldsymbol{A} : (\mathsf{D}, \mathsf{N}) \leftarrow$ Parameter
▷ Represents structured $N \times N$ matrix
$\boldsymbol{B} : (\mathsf{D}, \mathsf{N}) \leftarrow$ Parameter
$\boldsymbol{C} : (\mathsf{D}, \mathsf{N}) \leftarrow$ Parameter
$\Delta : (\mathsf{D}) \leftarrow \tau_\Delta(\text{Parameter})$
$\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathsf{D}, \mathsf{N}) \leftarrow \text{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$
$y \leftarrow \text{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-invariant: recurrence or convolution
return y

Algorithm 2 SSM + Selection (S6)

Input: $x : (\mathsf{B}, \mathsf{L}, \mathsf{D})$
Output: $y : (\mathsf{B}, \mathsf{L}, \mathsf{D})$
$\boldsymbol{A} : (\mathsf{D}, \mathsf{N}) \leftarrow$ Parameter
▷ Represents structured $N \times N$ matrix
$\boldsymbol{B} : (\mathsf{B}, \mathsf{L}, \mathsf{N}) \leftarrow s_B(x)$
$\boldsymbol{C} : (\mathsf{B}, \mathsf{L}, \mathsf{N}) \leftarrow s_C(x)$
$\Delta : (\mathsf{B}, \mathsf{L}, \mathsf{D}) \leftarrow \tau_\Delta(\text{Parameter} + s_\Delta(x))$
$\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathsf{B}, \mathsf{L}, \mathsf{D}, \mathsf{N}) \leftarrow \text{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$
$y \leftarrow \text{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-varying: recurrence (*scan*) only
return y

Figure 1.4: from [5]. (Algorithm 1) procedure previously used on S4, (Algorithm 2) proposed selection mechanism

The mentioned functions are defined as follows, where $Linear_d$ is a parameterized projection to dimension d:

$$s_B(x) = \text{Linear}_N(x) \qquad s_C(x) = \text{Linear}_N(x)$$
$$s_\Delta(x) = \text{Broadcast}_D(\text{Linear}_1(x)) \qquad \tau_\Delta = \text{softplus}$$

In [5] it is proved that when $N = 1$, $A = -1$, $B = 1$, the selective SSM recurrence takes the form

$$h_t = (1 - \sigma(Linear(x_t)))h_{t-1} + \sigma(Linear(x_t))x_t$$

In particular, if a given input x_t should be completely ignored (as necessary in the selection tasks), all D channels should ignore it. The input is projected down to 1 dimension before repeating/broadcasting with Δ, which controls the balance between how much to focus on or ignore the current input, generalizing the RNN gates.

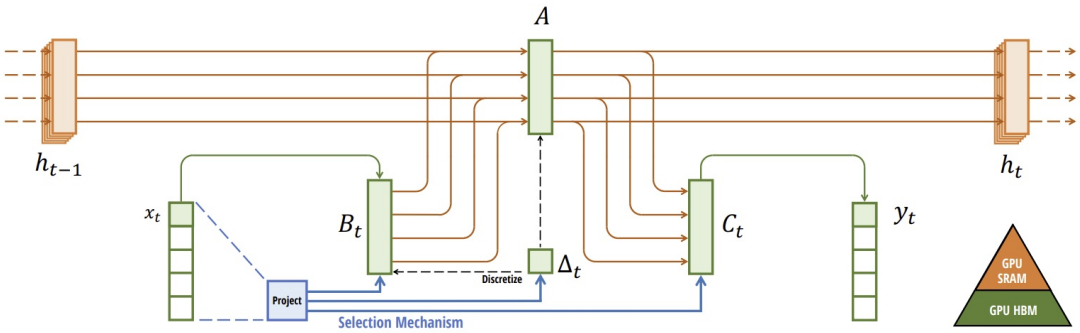

Figure 1.5: from [5]. S4s independently map each channel (e.g. D = 5) of an input x to output y through a higher dimensional latent state h (e.g. N = 4). The selection mechanism adds back input-dependent dynamics, which also requires a careful hardware-aware algorithm to only compute the expanded states in more efficient levels of the GPU memory hierarchy.

In terms of computation, the core idea is to take advantage of the strengths of modern accelerators (GPUs) by storing the state h only in the most efficient levels of the memory hierarchy. The selective scan operation is constrained by memory bandwidth; kernel fusion is used to minimize memory I/O. Specifically, instead of preparing the scan input $(\overline{A}, \overline{B})$ with shape (B, L, D, N) in the GPU's high-bandwidth memory (HBM), the SSM parameters (Δ, A, B, C) are directly loaded into faster SRAM, where discretization and recurrence computations are then performed, and only the final outputs of shape (B, L, D) are written back to HBM (1.5).
Additionally, to reduce memory usage, the storage of intermediate states needed for backpropagation is avoided. Instead, they are recomputed during the backward pass by reloading input from HBM to SRAM. This approach allows the selective scan layer to match the memory efficiency of an optimized Transformer implementation using FlashAttention.

Chapter 2

Mamba Models

This chapter presents the selection and preliminary evaluation of Mamba-based models for the segmentation of medical images. The selection process began with a review of recent survey papers on image segmentation [35] [34] [16] [23], from which models with publicly available code were identified. Then, the models were ranked on the basis of their citation counts. Given the intention to apply the models to a medical imaging dataset, preference was given to architectures that had previously been used in medical contexts, with the expectation that they would be accurate with polyp segmentation. The models in the table below emerged as great implementations for the medical image segmentation task [36] [17] [18] [24] [33] [15] [31], the models are shown with their respective tested datasets:

Dataset	Vision Mamba	VMamba	**U-Mamba**	VM-UNet	SegMamba	**Swin-Umamba**	Mamba-UNet
ADE20K	✓	✓					
COCO 2017	✓	✓					
Abdomen CT			✓				
Abdomen MRI			✓			✓	
Endoscopy images			✓			✓	
Microscopy images			✓			✓	
ISIC17, ISIC18				✓			
Synapse				✓			✓
CRC-50					✓		
BraTS2023					✓		
AIIB202					✓		
ACDC MRI							✓

Both U-Mamba (2d - 3d fullres) and Swin-UMamba (2d) are tested with Abdomen MRI, Endoscopy images, and Microscopy images datasets, so they are comparable. These two implementations have shown promising results modeling medical image datasets, and so have been selected in this paper for a deeper investigation with polyp segmentation datasets. Based on the analysis of the results reported in the articles, Swin-UMamba appears to outperform U-Mamba, achieving consistently higher Dice scores. In particular, U-Mamba's scores were respectively 2%, 3%, and 6% lower across the three datasets considered.

2.1 U-Mamba

U-Mamba [18] is designed to acquire the best of U-Net [9] and Mamba [5] to understand the global context in long-range dependencies for medical image segmentation. It follows the encoder-decoder network structure that features the U shape shown in figure (2.1). Each building block first contains two successive Residual blocks, where convolution and Leaky ReLU are applied, then image features with a shape of (B, C, H, W, D) are flattened and transposed to $(B, L = H \times W \times D, C)$ in the Normalization Layer, and subsequently they enter the Mamba block. The features of the two branches are merged with the Hadamard product; finally, they are projected back and transposed to the original (B, C, H, W, D) shape. The encoder is built with the mentioned blocks to capture both local features and long-range dependencies, while the decoder, composed of Residual blocks and transposed convolutions, focuses on detailed local information and resolution recovery. Moreover, the U-Net skip connection is inherited to connect the hierarchical features from the encoder to the decoder. The final decoder feature is passed to a $1 \times 1 \times 1$ convolutional and Softmax layer to produce the final segmentation probability map in the *U-Mamba_Enc* version, instead in the *U-Mamba_Bot* variant the bottleneck uses the U-Mamba block.

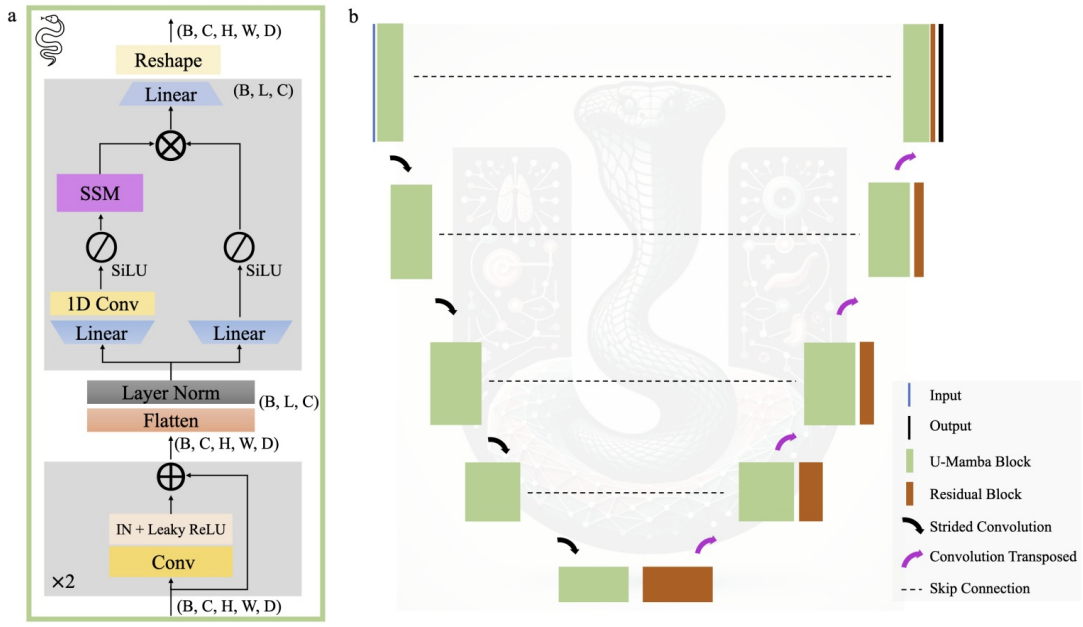

Figure 2.1: from [18]. Overview of the *U-Mamba_Enc* architecture. (a) U-Mamba building block. (b) encoder-decoder framework.

2.2 Swin-UMamba

Swin-UMamba [15] is mainly composed of a Mamba-based encoder that was pretrained on the large-scale ImageNet dataset [25] to extract features at different scales, a decoder with several up-sample blocks for predicting segmentation results, and skip connections to bridge the gap between low-level details and high-level semantics. The visual state space (VSS) block is the basic unit: it employs 2D-selective-scan to unfold image patches along four directions, creating

four distinct sequences. Subsequently, each feature sequence is processed through the SSM. Finally, the output features are merged to form the complete 2D feature map. The selective scan space state sequential model from [17] is the core SSM operator of the VSS block, enabling each element in a 1D sequence to interact with any of the previously scanned samples through a compressed hidden state.

Pretraining on ImageNET

The encoder has been pretrained on the extensive ImageNet dataset with multi-scale features, to integrate the power of the generic vision model to extract information with long-range modeling capability, mimic the risk of overfitting, and establish a robust initialization for Swin-UMamba. The encoder features five stages, each stage down-samples by a factor of 2, aiming to retain low-level details. The first stage contains a convolution layer, and then a 2d instance normalization layer. The second stage uses a patch embedding layer, maintaining the feature resolution at $\frac{1}{4}\times$ of the original image. Subsequent stages are composed of a patch merging layer for down-sampling and several VSS blocks for high-level feature extraction. There are $2, 2, 9, 2$ VSS blocks from stage 2 to 5, respectively; the feature dimensions after each stage are quadratically increased. All VSS blocks are initialized using the ImageNet pretrained weights from VMamba-Tiny.

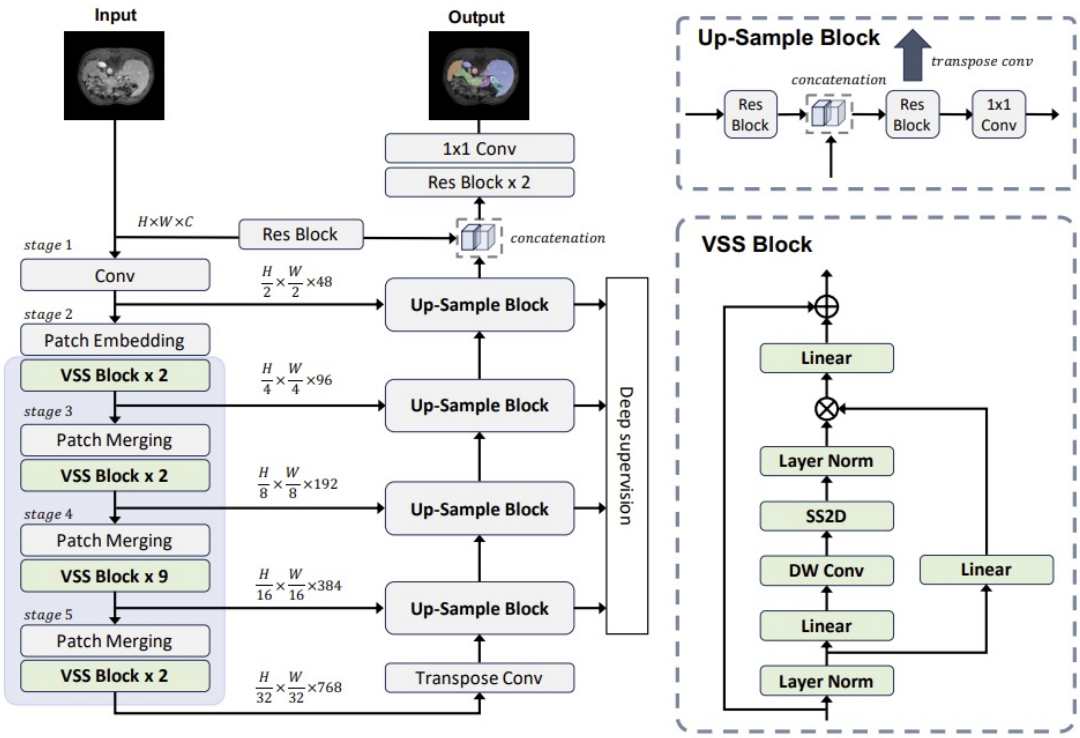

Figure 2.2: from [15]. The overall architecture of Swin-UMamba. Each block within the blue box was initialized with the ImageNet pretrained weights.

Decoder

The U-Net architecture leverages skip connections for the recovery of low-level details and employs an encoder-decoder structure for high-level information extraction. To enhance the native up-sample block, an extra convolution block with a residual connection to process skip connec-

tion features, and an additional segmentation head at each scale for deep supervision, have been added. Swin-UMamba† is a proposed variant that features a Mamba-based decoder. It obtains lower but competitive results compared to the original, utilizing fewer network parameters to reduce the computational burden.

Chapter 3

Datasets

This chapter provides a brief description of the datasets that have been involved in the research. AbdomenMR (3.1) was used to validate and confirm the results achieved in the articles [18][15]. In this study, we were not interested in abdominal datasets, but rather in polyp detection, which represents a different medical application. Kvasir-SEG (3.2) and HardNet-MSEG (3.3) are two popular datasets for the medical image segmentation of colorectal polyps; they were not previously tested in the literature with U-Mamba and Swin-UMamba. Polyps are abnormal tissue growths that typically form on mucous membranes inside the body. They are tumors that usually develop in various parts of the body, such as the gastrointestinal tract, which includes the esophagus, stomach, small intestine, colon, and rectum. World Health Organization states that Colorectal cancer (CRC) is the third most commonly diagnosed cancer worldwide and represents the second leading cause of cancer-related mortality. It predominantly affects people aged 50 years and older. The etiology of CRC is closely associated with modifiable lifestyle factors, including high consumption of processed meats, insufficient intake of fruits and vegetables, physical inactivity, obesity, tobacco use, and excessive alcohol consumption. CRC is frequently diagnosed at advanced stages, thereby reducing the efficacy of therapeutic interventions. Nonetheless, its incidence and associated mortality can be substantially mitigated through the adoption of primary prevention strategies, including the promotion of healthy behaviors, avoidance of established risk factors, and the implementation of systematic screening programs for early detection [21]. The 95% of CRC cases is due to colorectal adenomatous polyps, early polyp resection surgery can greatly improve the prognosis of CRC.

3.1 AbdomenMR

AbdomenMR [11] is the dataset that has been used to test both U-Mamba and Swin-UMamba by their authors, it is known as Dataset702_AbdomenMR. The training set contains 60 $(image, mask)$ samples, while there are 50 samples in the test set. Each mask can have the following set of labels: background, liver, right kidney, spleen, pancreas, aorta, inferior vena cava, right adrenal gland, left adrenal gland, gallbladder, esophagus, stomach, duodenum, and left kidney. The labels are respectively associated to a certain shade of black of the mask image.

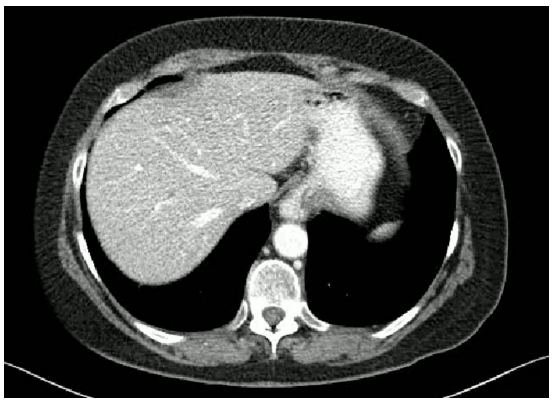

Figure 3.1: image from training dataset

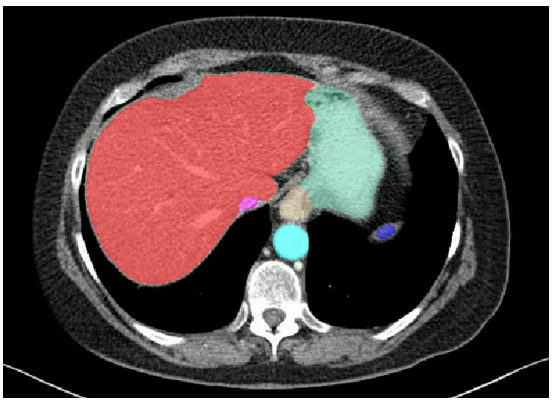

Figure 3.2: colored label representation

3.2 Kvasir-SEG

The Kvasir-SEG dataset [10] (size 46.2 MB) contains 1000 $(image, mask)$ samples of polyp images and their corresponding ground truth from the Kvasir Dataset v2[22]. The resolution of the images contained in Kvasir-SEG varies from 332x487 to 1920x1072 pixels. The image files are encoded using JPEG compression.

The dataset has been adapted with *convert_kvasir-seg.py* [13]: the script picks 900 samples for training and 100 for testing, and stores them in the required nnU-Net folder structure [9]. The raw images are converted to PNG format and masks in binary PNG matrices, so the background label is associated with bit 0 and the polyp with bit 1, as described in the dataset.json file that is generated. The *convert_kvasir-seg.sh* [13] bash file is designed to be placed inside the model/data path with the aforementioned Python script; when run in a virtual environment, it automatically installs the required dependencies, downloads the dataset, converts it to the required nnU-Net format [9], and finally removes the downloaded files.

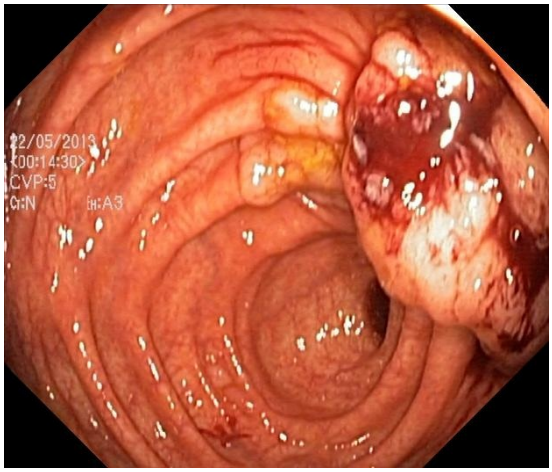

Figure 3.3: image from training dataset

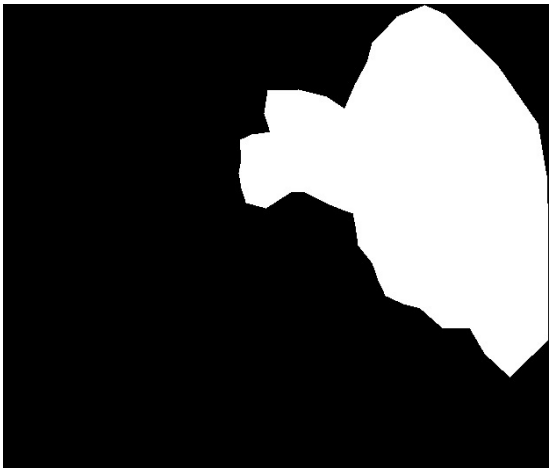

Figure 3.4: black-white label mask

3.3 HarDNet-MSEG

HarDNet-MSEG [8] is a group of popular datasets regarding polyp segmentation. The training set has 1450 $(image, mask)$ samples from CVC-ClinicDB [3] and Kvasir-SEG [10], while the test set counts 798 $(image, mask)$ samples from: CVC-300 [30] (60), CVC-ClinicDB [3] (62), CVC-ColonDB [27] (380), ETIS-LaribPolypDB [26] (196), and Kvasir-SEG [10] (100). Hiding three datasets from the training set allows us to measure the generalization and overfitting features of the network.

The dataset was processed using the *convert_HarDNet-MSEG.py* script [13], which extracts samples from their respective subfolders and reorganizes them into the nnU-Net folder structure as specified in [9]. Original images are converted to PNG format, while masks are transformed into binary PNG matrices where the background corresponds to bit 0 and the polyp to bit 1, as detailed in the generated dataset.json file. Additionally, the *convert_HarDNet-MSEG.sh* [13] bash script is intended to be placed within the model/data directory alongside the Python script. When executed in a virtual environment, it automatically installs all necessary dependencies, downloads the dataset, converts it to the nnU-Net compatible format [9], and subsequently deletes the original downloaded files.

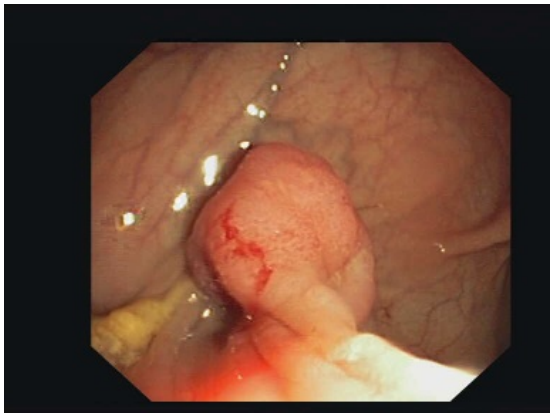

Figure 3.5: image from training dataset

Figure 3.6: black-white label mask

Chapter 4

Experiments

This section describes the experiments that have been performed on U-Mamba and Swin-UMamba: section (4.1) shows the installation and validation of the two models: they were trained with the AbdomenMR dataset, and the results were compared with those reported in the original articles. In (4.2), there are the results of model training on Kvasir-SEG and HarDNet-MSEG polyp datasets with various model configurations. Finally, in (4.3) two different data augmentations have been designed and tested with Swin-UMamba, the results are shown in comparison with current state-of-the-art (SOTA) models.
All trainings, inferences, and augmentations of this work have been carried out on the 'Blade' Cluster of the Department of Information Engineering at the University of Padua. Blade is a computing cluster that features many nodes with different hardware, it is managed with Simple Linux Utility for Resource Management (SLURM), which lets you allocate resources using a batch or job file. Nodes 'gpu5' and 'gpu6' feature 64 CPU cores each (2x Intel Xeon Platinum 8358 CPU @ 2.60/3.40GHz), 1TB RAM, 10 GPUs Nvidia A40. The slurm batch files are documented and all the jobs were executed on a single Nvidia A40 GPU.
To evaluate the performance of the segmentation models, two metrics were employed: the Dice Similarity Coefficient (DSC) and the Normalized Surface Dice (NSD). The DSC measures the volumetric overlap between the predicted segmentation and the ground truth, providing a robust indication of the general accuracy of the segmentation. The NSD assesses the alignment of the predicted and actual surfaces by calculating the proportion of surface points within a distance threshold.

4.1 Installation and validation

The first step of the experimentation is the deployment of both U-Mamba and Swin-UMamba in the Blade cluster. The cluster has been accessed via the Secure Shell (SSH) connection. Miniconda3 [1] has been installed in the local partition and two virtual environments (one for each model) have been created, activated and filled with the required dependencies. All the steps required to install, prepare the datasets, train, and test in the cluster are fully documented in the repository [13].

The validation of the installation and model performance was conducted using the AbdomenMR dataset, following the training setup described in the original paper. Models were trained for 200 and 1000 epochs, as reported by the authors [18] [15]. The term *validation* refers to taking the available model code on Github, verifying its functionality, and testing it on the datasets originally declared by the authors. The goal is to assess whether the implementation works correctly and yields reliable results that are consistent with those reported in the original studies. It was carried out using the same data and methodology referenced in the paper to ensure consistency and comparability of results.
Here a breakdown of the results performed on Blade and the results mentioned in the papers:

AbdomenMR		200 EPOCHS				1000 EPOCHS			
		computed		paper		computed		paper	
		DSC	NSD	DSC	NSD	DSC	NSD	DSC	NSD
Umamba	nnUNetTrainerUMambaBot	0.7278	0.7955	0.7588	0.8285	0.7532	0.8237	0.7588	0.8285
	nnUNetTrainerUMambaEnc	0.7319	0.8012	0.7625	0.8327	0.7541	0.8234	0.7625	0.8327
Swin-Umamba	nnUNetTrainerSwinUMamba	0.7723	0.8408	0.7760	0.8421	0.7642	0.8366	-	-

The computed results show an average deviation of approximately 2% from the values reported in the original paper, indicating a high level of consistency and successful replication of the model performance. The results provide evidence that Swin-UMamba performances are higher than U-Mamba, the model parameters are set for a best 200-epoch training, which outweighs even the 1000-epoch training. Swin-UMamba appears to overfit when trained for 1000 epochs, as its performance on the test set decreases compared to the 200-epoch results. Training Swin-UMamba for 200 epochs takes approximately 21 hours, whereas both U-Mamba trainers require around 15.5 hours. For 1000 epochs, Swin-UMamba takes about 110 hours, U-MambaBot approximately 121 hours, and U-MambaEnc significantly less at around 80 hours.

4.2 Training and evaluation

Since the validation phase showed strong alignment with the declared capabilities of the models, with less than a 2% difference from the expected results, the models were trained on the Kvasir-SEG and HarDNet-MSEG polyp datasets to further demonstrate their effectiveness in a different domain of medical images.

4.2.1 Kvasir-SEG

Having adapted the Kvasir-SEG dataset for training, it was subsequently used to test both models. The results of this evaluation are presented below.

Kvasir-SEG		200 EPOCHS		500 EPOCHS	
		DSC	NSD	DSC	NSD
Umamba	nnUNetTrainerUMambaBot	0.9036	0.9191	-	-
	nnUNetTrainerUMambaEnc	0.8870	0.9022	-	-
Swin-Umamba	nnUNetTrainerSwinUMamba	0.9113	0.9266	0.9164	0.9324

U-Mamba achieved the best Dice results with BOT on the Kvasir-SEG dataset, reflecting strong performance in both segmentation accuracy and boundary adherence.
The Swin UMamba model improved with a +1% increase on *U-Mamba_Bot* and a +3% gain on *U-Mamba_Enc* metrics, reflecting excellent accuracy in both region overlap and contour precision. These results highlight the model's capability to capture fine-grained polyp details, which is essential for reliable medical image analysis. This strong performance establishes it as a promising candidate for real-world applications in colonoscopy-based decision support systems.

4.2.2 HarDNet-MSEG

Training becomes more challenging when incorporating the HarDNet-MSEG dataset, as it is a well-established benchmark for evaluating model performance across diverse conditions. This dataset includes five distinct subsets, of which only two are exposed during training, making it a rigorous test of generalization. Consequently, performance on the unseen subsets often declines, highlighting the difficulties models face in generalizing beyond the training distribution. It is evident that performance deteriorates, making polyp detection more challenging.

HarDNet-MSEG		200 EPOCHS		500 EPOCHS	
		DSC	NSD	DSC	NSD
Umamba	nnUNetTrainerUMambaBot	0.7143	0.7378	-	-
	nnUNetTrainerUMambaEnc	0.7114	0.7370	-	-
Swin-Umamba	nnUNetTrainerSwinUMamba	0.7633	0.7894	0.7669	0.7926

The table below provides a more detailed analysis to investigate the factors contributing to the loss in performance, the datasets labeled with (u) were unseen during the training phase. Analyzing performance on the seen datasets, it is observed that on Kvasir-SEG the results remain unchanged for Swin-UMamba, while only slight variations are present between the two U-Mamba versions. Similarly, on CVC-ClinicDB, the results are very good and closely comparable to those on Kvasir-SEG. Looking at the results on unseen datasets, CVC-300 shows very good performance, only slightly lower than on the seen datasets. CVC-ColonDB, on the other hand, yields results around the previously reported average for the model, indicating mediocre performance. The situation is much worse with ETIS-LaribPolypDB, which obtains very low DSC scores around 50–60%, resulting in poor segmentation precision. This is likely due to the distinct characteristics of the dataset: some images are extracted from video sequences

that are quite different from the other datasets, and they also present variations in lighting and overall image quality.

HarDNet-MSEG			200 EPOCHS		500 EPOCHS	
			DSC	NSD	DSC	NSD
Umamba	nnUNetTrainerUMambaBot	CVC-300 (u)	0.8452	0.8782	-	-
		CVC-ClinicDB	0.8965	0.9287	-	-
		CVC-ColonDB (u)	0.7162	0.7440	-	-
		ETIS-LaribPolypDB (u)	0.5253	0.5387	-	-
		Kvasir-SEG	0.8855	0.9019	-	-
	nnUNetTrainerUMambaEnc	CVC-300 (u)	0.8668	0.9024	-	-
		CVC-ClinicDB	0.8813	0.9135	-	-
		CVC-ColonDB (u)	0.7424	0.7734	-	-
		ETIS-LaribPolypDB (u)	0.4556	0.4695	-	-
		Kvasir-SEG	0.8966	0.9143	-	-
Swin-Umamba	nnUNetTrainerSwinUMamba	CVC-300 (u)	0.8740	0.9065	0.8679	0.9005
		CVC-ClinicDB	0.9147	0.9498	0.9177	0.9491
		CVC-ColonDB (u)	0.7742	0.8041	0.7816	0.8115
		ETIS-LaribPolypDB (u)	0.5850	0.6029	0.5894	0.6076
		Kvasir-SEG	0.9111	0.9290	0.9049	0.9219

Below a table with performance of SOTA models on HarDNet-MSEG, sorted by average Dice score, the best scores in each column are marked in bold.
Swin-UMamba results compared to the best Dice score for each subdataset is only -3% (CVC-300), -2% (ClinicDB), and -1% (Kvasir-SEG) away from SOTA. The low scores indicate a notable performance gap when compared with the SOTA for the last two data sets -8.5% (ColonDB) and -30% (ETIS-LaribPolyp), lowering the average distance from SAM-Mamba by -8%. The results demonstrate strong performance on seen datasets but substantially lower performance on unseen datasets, indicating limited generalization capability.
U-Mamba, particularly the *U-Mamba_Bot* training variant, exhibits behavior comparable to Swin-UMamba. However, due to consistently lower Dice scores, only Swin-UMamba is selected for further investigation.

Model	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
SAM-Mamba [4]	**0.920**	0.942	0.924	**0.853**	**0.848**	**0.897**
MGCBFormer [32]	0.913	**0.955**	0.931	0.807	0.819	0.885
DS-TransUNet-L [14]	0.911	0.936	**0.935**	0.798	0.761	0.868
Swin-UMamba	0.890	0.932	0.920	0.789	0.594	0.825
U-Mamba (bot)	0.862	0.9126	0.894	0.730	0.532	0.786
U-Mamba (enc)	0.885	0.897	0.905	0.758	0.463	0.782

4.3 Data augmentation

The performances of a segmentation model are influenced by the size of the dataset, thus data augmentation is performed to double the size of the training set. The proposed techniques modify brightness, contrast, color, and sharpness levels of the images; they have been implemented with the Pillow Python Imaging Library and the code is available in the repository [13]. The dedicated scripts are *da1.py* and *da2.py*: they both implement augmentation in the defined *augment_sample* function.

Data augmentation has been performed on HarDNet-MSEG, because the training set includes just two out of the five subdatasets contained in the test set, and previous results show a clear loss in segmentation quality on the unseen datasets. The test set remains unchanged. Since Swin-UMamba has shown to be the most promising architecture, it has been selected to test with the proposed data augmentations.

4.3.1 Data Augmentation 1 (DA1)

The first proposed augmentation consists of five predefined sets of scalar values, each randomly selected to adjust brightness, contrast, color, and sharpness. These values were chosen based on preliminary experimentation, as they yielded the greatest visual diversity among the augmented images. The sets are cyclically applied to the training images, the results are not designed to be visually realistic, but to enhance training generalization and support the learning of relevant features. In the table below, the obtained results are provided for the five testing sets.

HarDNet-MSEG DA1			200 EPOCHS		500 EPOCHS	
			DSC	NSD	DSC	NSD
Swin-Umamba	nnUNetTrainerSwinUMamba	CVC-300 (u)	0.8891	0.9229	0.8885	0.9219
		CVC-ClinicDB	0.9209	0.9536	0.9179	0.9498
		CVC-ColonDB (u)	0.7745	0.8055	0.7868	0.8166
		ETIS-LaribPolypDB (u)	0.6845	0.7057	0.6345	0.6537
		Kvasir-SEG	0.9106	0.9277	0.9107	0.9282

The results provide evidence of overfitting at 500 epochs, as performance drops significantly, particularly on unseen datasets, indicating reduced generalization capability.

At 200 epochs, all results show slight improvements, notably, ETIS-LaribPolypDB, which previously had the lowest performance, shows a substantial gain of 17% in both DSC and NSD.

4.3.2 Data Augmentation 2 (DA2)

The second proposed augmentation consists of nine predefined sets of scalar values accurately selected to adjust brightness, contrast, and color. These values were chosen based on preliminary experimentation, the adjustments excluded sharpness to achieve realistic results. The sets are cyclically applied to the training images, this augmentation main achievement is to improve

image quality, while doubling the size of the samples. In the table below, the obtained results are provided for the five testing sets.

HarDNet-MSEG DA2			200 EPOCHS		500 EPOCHS	
			DSC	NSD	DSC	NSD
Swin-Umamba	nnUNetTrainerSwinUMamba	CVC-300 (u)	0.8870	0.9212	0.8911	0.9250
		CVC-ClinicDB	0.9173	0.9497	0.9224	0.9536
		CVC-ColonDB (u)	0.7720	0.8024	0.7876	0.8184
		ETIS-LaribPolypDB (u)	0.6382	0.6587	0.5706	0.5878
		Kvasir-SEG	0.9104	0.9273	0.9120	0.9296

The results provide, again, evidence of overfitting at 500 epochs, as performance drops significantly. At 200 epochs, all results show slight improvements: ETIS-LaribPolypDB shows a gain of 9% in both DSC and NSD, with respect to original dataset training.

HarDNet-MSEG overall			200 EPOCHS		500 EPOCHS	
			DSC	NSD	DSC	NSD
Swin-Umamba	nnUNetTrainerSwinUMamba	ORIGINAL	0.7633	0.7894	0.7669	0.7926
		DA1	0.7894	0.8166	0.7828	0.8088
		DA2	0.7764	0.8031	0.7682	0.7942

Overall, DA1 leads to a 3.5% improvement, while DA2 results in a 2% gain in performance. These findings suggest that generating realistic images is not essential; instead, synthetic images with randomized values contribute to making the training set more diverse and general. This enhanced diversity improves the model's generalization capabilities, especially on datasets that were not seen during training but only used for testing. Additionally, the best performance is achieved with 200 epochs, highlighting the importance of early stopping to prevent overfitting.

Model	CVC-T	ClinDB	Kvasir	ColDB	ETIS	Average
SAM-Mamba [4]	**0.920**	0.942	0.924	**0.853**	**0.848**	**0.897**
MGCBFormer [32]	0.913	**0.955**	0.931	0.807	0.819	0.885
DS-TransUNet-L [14]	0.911	0.936	**0.935**	0.798	0.761	0.868
Swin-UMamba	0.908	0.938	0.921	0.803	0.695	0.853

The best performances achieved by Swin-UMamba are on average just 5% below the SOTA on the HarDNet-MSEG dataset. The Dice scores on ColonDB and ETIS-LaribPolyp are respectively down by 6% and 18%, while the performance drop on CVC-300, ClinicDB, and Kvasir-SEG remains below 2%.

Conclusions

In conclusion, Mamba-based models demonstrate strong potential for application in computer vision tasks, particularly in medical image segmentation, where high reliability, precision, and accuracy are essential. Their integration could significantly accelerate diagnosis and streamline monitoring processes in the treatment of tumors and polyps. Moreover, their use could extend to a wide range of medical domains. One of their key advantages lies in their ability to train on datasets without theoretical size limitations, offering a significant edge over transformer-based architectures in terms of scalability and training efficiency. Their fast inference speed and compact model size also make them ideal for embedded solutions, where rapid diagnoses and low computational requirements are crucial.

Their performance is highly promising, showing both robustness and adaptability across diverse datasets. Of course, there is still room for improvement to make these models optimal and generalizable. The research suggests that polyp detection is approaching the state-of-the-art with Swin-UMamba. Future work could focus on designing more sophisticated and diverse data augmentation strategies to further enhance model generalization.

Bibliography

[1] Anaconda. *Miniconda*. https://www.anaconda.com/docs/getting-started/miniconda/main. Accessed: 2025.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. «Neural Machine Translation by Jointly Learning to Align and translate». In: (2014). DOI: https://doi.org/10.48550/arXiv.1409.0473.

[3] Jorge Bernal et al. «WM-DOVA maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians». In: *Computerized Medical Imaging and Graphics* (2015). DOI: https://doi.org/10.1016/j.compmedimag.2015.02.007.

[4] Tapas Kumar Dutta et al. «SAM-Mamba: Mamba Guided SAM Architecture for Generalized Zero-Shot Polyp Segmentation». In: *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2025, pp. 4655–4664. DOI: 10.1109/WACV61041.2025.00457.

[5] Albert Gu and Tri Dao. «Mamba: Linear-Time Sequence Modeling with Selective State Spaces.» In: (2023). DOI: https://doi.org/10.48550/arXiv.2312.00752.

[6] Albert Gu, Karan Goel, and Christopher Ré. «Efficiently Modeling Long Sequences with Structured State Spaces». In: (2021). DOI: https://doi.org/10.48550/arXiv.2111.00396.

[7] Albert Gu et al. «Combining Recurrent, Convolutional, and Continuous-time Models with Linear State-Space Layers». In: (2021). DOI: https://doi.org/10.48550/arXiv.2110.13985.

[8] Chien-Hsiang Huang, Hung-Yu Wu, and Youn-Long Lin. «HarDNet-MSEG: A Simple Encoder-Decoder Polyp Segmentation Neural Network that Achieves over 0.9 Mean Dice and 86 FPS». In: (2021). DOI: https://doi.org/10.48550/arXiv.2101.07172.

[9] Fabian Isensee et al. «nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation». In: *Nature Methods* (2020). DOI: https://doi.org/10.1038/s41592-020-01008-z.

[10] Debesh Jha et al. «Kvasir-seg: A segmented polyp dataset». In: *International Conference on Multimedia Modeling*. Springer. 2020, pp. 451–462.

[11] Yuanfeng Ji et al. «AMOS: A Large-Scale Abdominal Multi-Organ Benchmark for Versatile Medical Image Segmentation». In: (2022). DOI: https://doi.org/10.48550/arXiv.2206.08023.

[12] Rudolph Emil Kalman. «A New Approach to Linear Filtering and Prediction Problems». In: (1960). DOI: http://dx.doi.org/10.1115/1.3662552.

[13] Alberto Levorato. *MISWSSSM: Medical Image Segmentation with Selective State Space Models*. https://github.com/levalby/MISWSSSM. 2025.

[14] Ailiang Lin et al. «DS-TransUNet: Dual Swin Transformer U-Net for Medical Image Segmentation». In: *IEEE Transactions on Instrumentation and Measurement* 71 (2022), pp. 1–15. DOI: 10.1109/TIM.2022.3178991.

[15] Jiarun Liu et al. «Swin-UMamba: Mamba-based UNet with ImageNet-based pretraining». In: (2024). DOI: https://doi.org/10.48550/arXiv.2402.03302.

[16] Xiao Liu, Chenxu Zhang, and Lei Zhang. «Vision Mamba: A Comprehensive Survey and Taxonomy». In: (2024). DOI: https://doi.org/10.48550/arXiv.2405.04404.

[17] Yue Liu et al. «VMamba: Visual State Space Model». In: (2024). DOI: https://doi.org/10.48550/arXiv.2401.10166.

[18] Jun Ma, Feifei Li, and Bo Wang. «U-Mamba: Enhancing Long-range Dependency for Biomedical Image Segmentation». In: (2024). DOI: https://doi.org/10.48550/arXiv.2401.04722.

[19] Yoshua Bengio Martin Arjovsky Amar Shah. «Unitary Evolution Recurrent Neural Networks». In: *The International Conference on Machine Learning (ICML)*. (2016). DOI: https://doi.org/10.48550/arXiv.1511.06464.

[20] HiPPO: Recurrent Memory with Optimal Polynomial Projections. «Albert Gu and Tri Dao and Stefano Ermon and Atri Rudra and Christopher Ré». In: (2020). DOI: https://doi.org/10.48550/arXiv.2008.07669.

[21] World Health Organization. *Colorectal cancer*. https://www.who.int/news-room/fact-sheets/detail/colorectal-cancer. Update: 2023-07-11. 2023.

[22] Konstantin Pogorelov et al. «KVASIR: A Multi-Class Image Dataset for Computer Aided Gastrointestinal Disease Detection». In: *Proceedings of the 8th ACM on Multimedia Systems Conference* (2017). DOI: https://doi.org/10.1145/3193289.

[23] Haohao Qu et al. «A Survey of Mamba». In: (2024). DOI: https://doi.org/10.48550/arXiv.2408.01129.

[24] Jiacheng Ruan, Jincheng Li, and Suncheng Xiang. «VM-UNet: Vision Mamba UNet for Medical Image Segmentation». In: (2024). DOI: https://doi.org/10.48550/arXiv.2402.02491.

[25] Olga Russakovsky et al. «ImageNet Large Scale Visual Recognition Challenge». In: (2014). DOI: https://doi.org/10.48550/arXiv.1409.0575.

[26] Juan Silva et al. ≪Toward embedded detection of polyps in wce images for early diagnosis of colorectal cancer≫. In: *International journal of computer assisted radiology and surgery* (2014). DOI: https://doi.org/10.1007/s11548-013-0926-3.

[27] Nima Tajbakhsh, Suryakanth R Gurudu, and Jianming Liang. ≪Automated polyp detection in colonoscopy videos using shape and context information≫. In: *IEEE transactions on medical imaging* (2015). DOI: https://doi.org/10.1109/TMI.2015.2487997.

[28] Arnold Tustin. ≪A method of analysing the behaviour of linear systems in terms of time series≫. In: *Journal of the Institution of Electrical Engineers-Part IIA: Automatic Regulators and Servo Mechanisms* (1947).

[29] Ashish Vaswani et al. ≪Attention Is All You Need≫. In: (2017). DOI: https://doi.org/10.48550/arXiv.1706.03762.

[30] David Vazquez et al. ≪A benchmark for endoluminal scene segmentation of colonoscopy images≫. In: *Journal of healthcare engineering* (2017). DOI: https://doi.org/10.1155/2017/4037190.

[31] Ziyang Wang et al. ≪Mamba-UNet: UNet-Like Pure Visual Mamba for Medical Image Segmentation≫. In: (2024). DOI: https://doi.org/10.48550/arXiv.2402.05079.

[32] Yang Xia et al. ≪MGCBFormer: The multiscale grid-prior and class-inter boundary-aware transformer for polyp segmentation≫. In: *Computers in Biology and Medicine* 167 (2023), p. 107600. ISSN: 0010-4825. DOI: 10.1016/j.compbiomed.2023.107600.

[33] Zhaohu Xing et al. ≪SegMamba: Long-range Sequential Modeling Mamba For 3D Medical Image Segmentation≫. In: (2024). DOI: https://doi.org/10.48550/arXiv.2401.13560.

[34] Rui Xu et al. ≪Visual Mamba: A Survey and New Outlooks≫. In: (2024). DOI: https://doi.org/10.48550/arXiv.2404.18861.

[35] Hanwei Zhang et al. ≪A Survey on Visual Mamba≫. In: (2024). DOI: https://doi.org/10.48550/arXiv.2404.15956.

[36] Lianghui Zhu et al. ≪Vision Mamba: Efficient Visual Representation Learning with Bidirectional State Space Model≫. In: (2024). DOI: https://doi.org/10.48550/arXiv.2401.09417.