UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Università degli Studi Di Padova

Department of Information Engineering

Bachelor's degree in Computer Engineering

DATA AUGMENTATION APPROACHES FOR POLYP SEGMENTATION

Supervisor
Prof. Loris Nanni

Candidate
Alberto Dorizza
1217616

Academic Year 2021/2022

Alla mia Famiglia

Non ho particolari talenti, sono solo appassionatamente curioso.
- Albert Einstein

Sometimes it is the very people who no one imagines anything of,
who do the things that no one can image.
- Alan Turing in “The Imitation Game”

Abstract

Image augmentation, and in general data augmentation techniques, can greatly improve the performances of deep neural networks through the creation of artificial patterns, in fact the presence of these new patterns helps the network to generalise thus avoiding overfitting, i.e. the excessive adaptation of the network to the training data. The application of these techniques has become increasingly important, especially in fields where data availability is scarce, such as the medical field.

This work reports various non-traditional image augmentation techniques based on filtering and mixup operations in the frequency domain, tone mapping, background change and others. To evaluate the performance and benefits, these techniques are applied to the Kvasir-SEG dataset, which is then used to train the DeepLabV3+ semantic segmentation model.

Sommario

L’image augmentation, e in generale, le tecniche di data augmentation, possono migliorare notevolmente le prestazioni delle reti neurali profonde attraverso la creazione di pattern artificiali, i quali aiutano la rete a generalizzare evitando così l’overfitting, cioè l’eccessivo adattamento della rete ai dati di addestramento. L’applicazione di tali tecniche è diventata sempre più importante, soprattutto nei campi in cui la disponibilità di dati è scarsa, come in quello medico.

Questo lavoro riporta varie tecniche non tradizionali di image augmentation basate su operazioni di mescolanza e filtraggio nel dominio della frequenza, mappatura dei toni, cambiamento di sfondo ed altre. Per valutare le prestazioni ed i benefici, tali tecniche vengono applicate al dataset Kvasir-SEG, utilizzato poi per addestrare il modello di segmentazione semantica DeepLabV3+.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Several studies and publications over the years have identified colorectal cancer as one of the most commonly diagnosed cancers.

For example, to mention a few, in 2008 [1] colorectal cancer was one of the most commonly diagnosed cancers along with lung and breast cancer. In 2012, as reported in [2], colorectal cancer was the second most commonly diagnosed cancer in women and the third among men with an estimated 693,900 deaths and 1.4 million cases. Then, if we focus on the United States, we can observe that in 2015 [3] colorectal cancer was the second leading cause of death and then became third in 2017 [4].

Since the presence of polyps, and therefore a failure to identify them, is the main cause of colorectal cancer, an early detection, in order to have an equally early removal, becomes very important to increase the chances of survival.

However, the identification of polyps is not a trivial task; on the contrary, sometimes, it becomes difficult and temporarily expensive even for the most experienced medical doctors because of the variety of their classes, each with different size, shape, position and colour intensity. In fact, as reported in [5], according to the size and type of polyps there is a percentage between 14% and 30% of unidentified polyps, so the presence of an automated system for the polyps recognition would be much more efficient, useful and effective for decreasing the cases of these types of cancers.

Nowadays we can find the application of computer vision techniques in innumerable fields such as self-driving cars, pedestrian detection, traffic flow analysis, X-ray analysis and many others. In fact, in recent decades with the ever-faster evolution of machine learning and thanks to the performance of deep artificial neural networks, greater than or equal to those of humans, many diagnostic tasks have been entrusted to automated systems.

This work focuses on a specific task of computer vision called semantic segmentation; this technique aims to label each pixel of an image and it is widely used in autonomous driving, robot vision and understanding, localisation, medical image analysis, scene understanding and others.

In detail, various non-traditional data augmentation techniques are proposed and tested with the aim of improving the performance of the DeepLabV3+ semantic segmentation architecture, trained with the Kvasir-SEG dataset that contains images of gastrointestinal polyps.

1.1 Motivation / Purpose

With the spread of deep learning, most semantic segmentation problems are tackled using architectures based on deep neural networks. But to be effective, the optimisation through the learning algorithm of the huge number of parameters that constituting the deep artificial neural networks, a large number of training examples are required.

This structural constraint can, as in the medical field, become a problem. Indeed, in medicine there are very few publicly accessible datasets containing a significant amount of labelled samples because of a several factors such as privacy, time and domain-specific knowledge to manually annotate large datasets.

A solution to this phenomenon, widely used in recent years, is called Data Augmentation: the main topic of this work.

1.2 Aims of this work

The primary objective of this project is to propose non-traditional data augmentation techniques and verify how much these can help the deep neural network in the semantic segmentation of polyps.

In addition to this, other goals include checking how much performance improves by combining the proposed techniques with the basic ones used as post-processing method and how performance changes using combinations of the proposed techniques.

1.3 Thesis Structure

The exposition and explanation of the terminology and theory that make up this work are contained in Chapter 2.

In Chapter 3 are exposed the proposed methods, the composition of the dataset in which they are applied and the architecture on which the results obtained are extracted.

The definition of the used metrics, the results, and the comparison of these with other data augmentation methods are reported in Chapter 4.

Finally, Chapter 5 contains the conclusions extracted from the reported experiments and possible future developments.

Chapter 2

Background

2.1 Data augmentation and its benefits

In general, data augmentation is a method used to increase the performance of a machine learning model through the creation of synthetic data from existing data. This technique is widely used in the various fields of deep learning because of the big necessity of input data required by deep artificial neural networks to learn at their best. As we can read in the literature, examples of different fields in which data augmentation is used are: Natural Language Processing [6], Time Series Analysis [7], Deep Graph Learning [8] and many others.

This procedure has as its main objective to increase the size of the training set in order to make the model more robust to variations of the input data. Specifically, data augmentation is used to improve robustness in machine learning models where the amount of data is scarce and becomes often essential to reduce overfitting.

The term overfitting identifies the phenomenon whereby the model fits (fitting) and memorizes too well (over) the data it has observed - those belonging to the training set - thus failing to generalise with data not already observed (See [9] for a better overview).

Therefore, the data augmentation techniques want to be a solution to the resolution of these problems by increasing the quantity of training examples thus increasing the ability of generalisation of the network and solving problems of class imbalance.

2.2 Image augmentation

Basically, the term “image augmentation” refers to all techniques of data augmentation where patterns of relative model are images. The purpose coincides with that of data augmentation, i.e. to create synthetic images from existing ones to improve network performance.

The use of these techniques may seem trivial, but they can increase generalizability performance of the model through increasing the diversity of examples that make up the training set.

The creation of artificial images can be through different ways of processing or combination of multiple processing, the most popular methods are described in the following sections.

2.2.1 Image Rotation

This technique is one of the simplest, but sometimes it can be really effective because, by randomly rotating the image clockwise or counterclockwise with random number of degrees, the object in the image will occupy a different region.

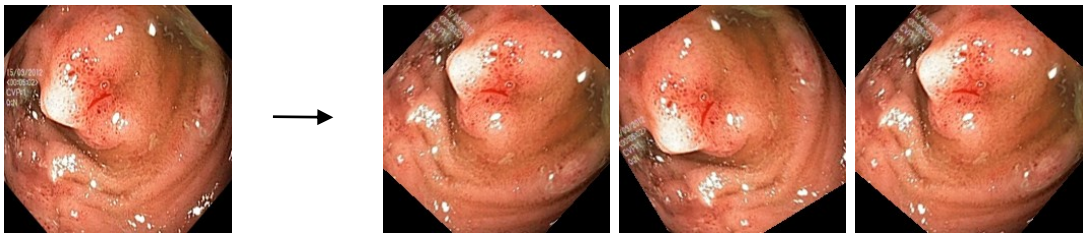


Figure 2.1 Examples of image rotation.

2.2.2 Image Shifting

This technique consists of moving all pixels of the image horizontally or vertically without changing the size of image. This operation can lead to cutting off parts of the image, but it is very useful in problems where the object to be identified does not have a precise position.

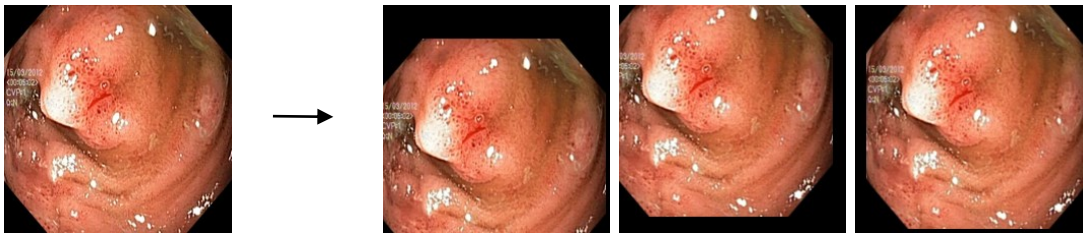


Figure 2.2 Examples of image shifting.

2.2.3 Image Flipping

This operation simply consists of reflecting the image vertically or horizontally. Flipping an image helps the model to avoid the memorization of relationships between groups of pixels that compose an object.

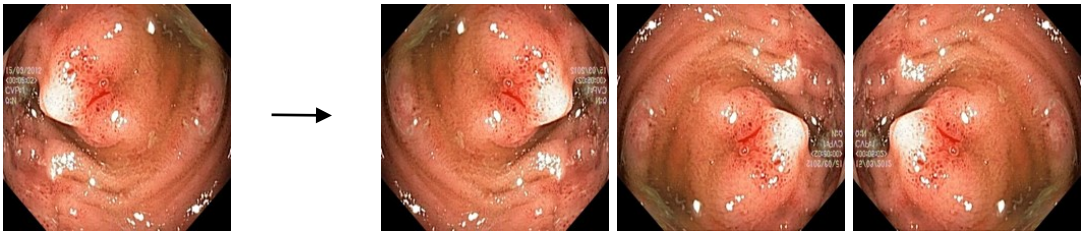


Figure 2.3 Examples of image flipping.

2.2.4 Image Scaling

Scaling an image can be seen as zooming in or zooming out the image. This effect changes the background and position of the object.

Figure 2.4 Examples of image scaling.

2.2.5 Image Noising

Unlike humans, image imperfections can significantly alter the prediction capabilities of neural networks. Several studies [10], [11] have investigated and demonstrated how the presence of imperfections can alter this performance. These studies have identified, for example, noise and blurring as the worst for the most common neural networks.

Figure 2.5 Examples of image noising.

See for [12] more information on noise models.

2.2.6 Image Color Trasformation

There are several techniques for changing the colours of an image, but all of them are based on changing the following four aspects of colour: brightness, contrast, saturation and hue.

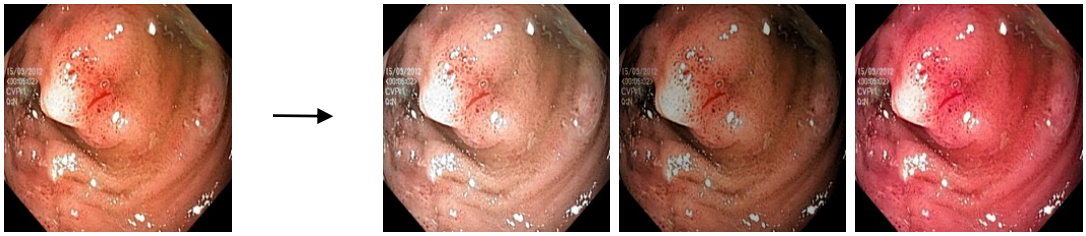


Figure 2.6 Examples of image color trasformation.

2.3 Computer vision tasks

Before giving a definition and understanding what semantic segmentation is, let us take a step back and identify the family to which this technique belongs: computer vision.

Computer vision is the branch of computer science that includes techniques and algorithms to process images and videos through computers in order to understand, interpret and comprehend their content similar to what happens with the human eye.

This field comprises several subdomains and it is used to resolve a variety of tasks.

The most popular tasks, in which the various computer vision techniques are involved, are:

- ***Image Classification***

  It aims to determine the class to which an image belongs. This technique does not provide information at the pixel level. For a survey of image classification see [13].

- ***Object Detection***

  This task focuses on identifying particular objects in the input image. In addition to identifying, by assigning them to their class, it creates a rectangle, technically called a "Bounding Box", which encloses the identified object in order to show its spatial position inside the image. For a survey of object detection see [14].

- ***Semantic Segmentation***

  The semantic segmentation is the process that has as objective the prediction of the class to which each pixel belongs; this technique then creates the corresponding segmentation mask for each class present in the input image.

  This method does not differentiate the different instances of the same object, as can be seen in the example in Figure 2.7 where all the pixels that make up the various cubes have the same label.

  So, it tries to identify objects belonging to the same class by doing an image classification at the pixel level. See next section for read more information about this computer vision task.

- ***Instance Segmentation***

  This technique is an evolution of the previous one, in particular it is a combination of semantic segmentation and object detection, so each resulting segmentation map does not contain all objects of the same class but only one instance of it.

  Therefore, unlike semantic segmentation, this method can identify the different instances of the same class, thus also knowing the number of them within the image.

Example to better understand the difference between the various tasks:

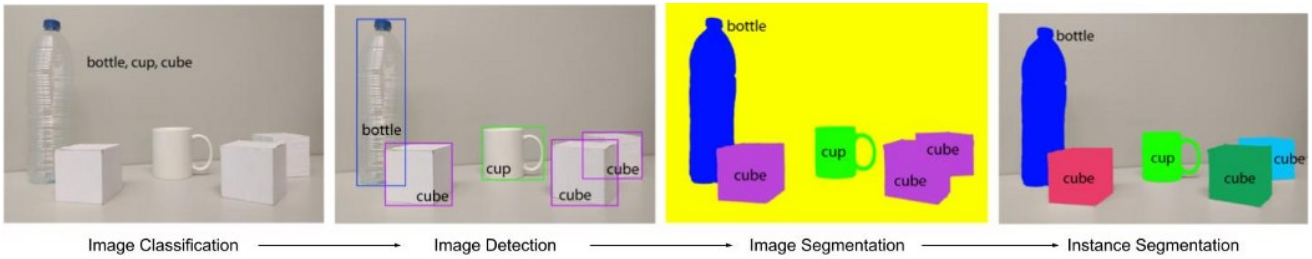


Figure 2.7 Example to better understand the difference between the various computer vision tasks [15]

See [16] for a better overview of computer vision and its other tasks.

2.4 Semantic segmentation before and after

Semantic segmentation is a long-standing computer vision task, before the birth of deep learning many algorithms were designed to try to solve this non-trivial task, e.g. Watershed algorithm, image thresholding, K-means clustering, Conditional Random Fields, etc.

For example, the oldest and simplest image thresholding algorithm, also known as "image binarization", converted the original image to a greyscale image and set the value of each pixel to 0 (black) if the pixel intensity value is greater than a threshold value or 1 (white) if it is less.

The emergence of deep neural networks also revolutionised the world of computer vision; in fact, the first types of networks used for semantic segmentation were Fully Convolutional Networks (FCN) in which, unlike CNNs, each neuron in a layer is connected to all the other neurons in the next layer and in which there are no constraints on input size.

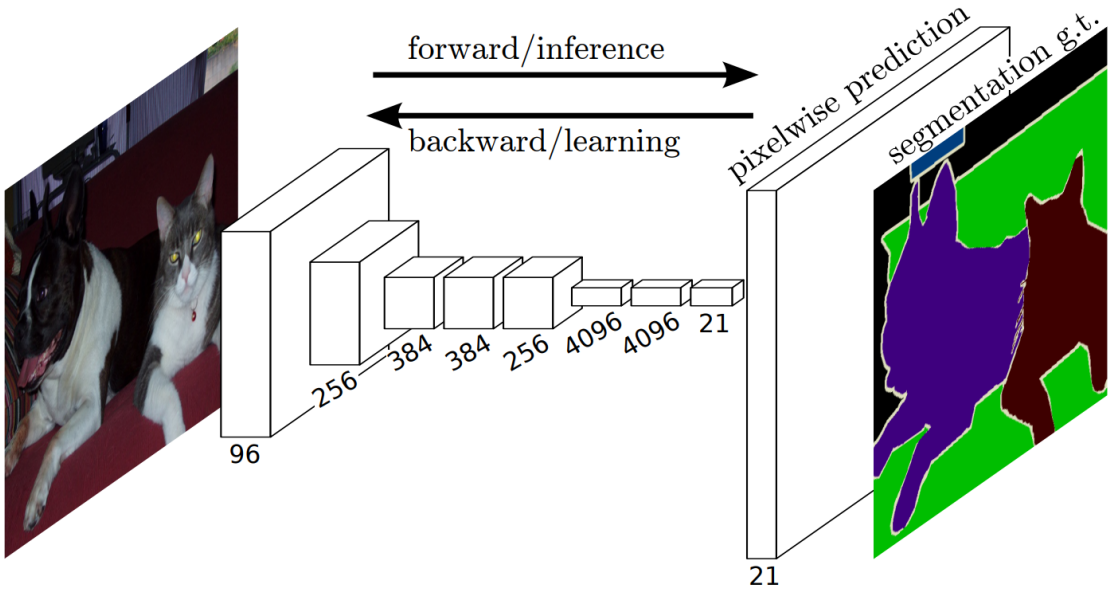


Figure 2.8 Fully Convolutional network architecture for semantic segmentation [17]

As time went by, new more advanced and efficient approaches based on FCNs were developed, such as SegNet [18], U-Net [19] and DeepLab [20]. These new systems always adopt the autoencoder architecture, consisting of two modules: the encoder that performs the downsampling of the input image and the decoder that, on the contrary, does the upper sampling using the feature vector output from the encoder.

In detail, the encoder is a pre-trained feature extraction network, like VGGNet, ResNet, etc. that produces a low dimensionality dense representation of the input image, and the decoder recovers the resolution lost by downsampling of the encoder.

Chapter 3

Materials and methods

3.1 Datasets

All the techniques proposed in this work are applied to the Kvasir-SEG [5] dataset proposed in 2019 by Jha et al.

Kvasir-SEG is an open-access dataset that contains images of gastrointestinal polyps with their corresponding segmentation mask, annotated by medical doctor and subsequently verified by expert gastroenterologist.

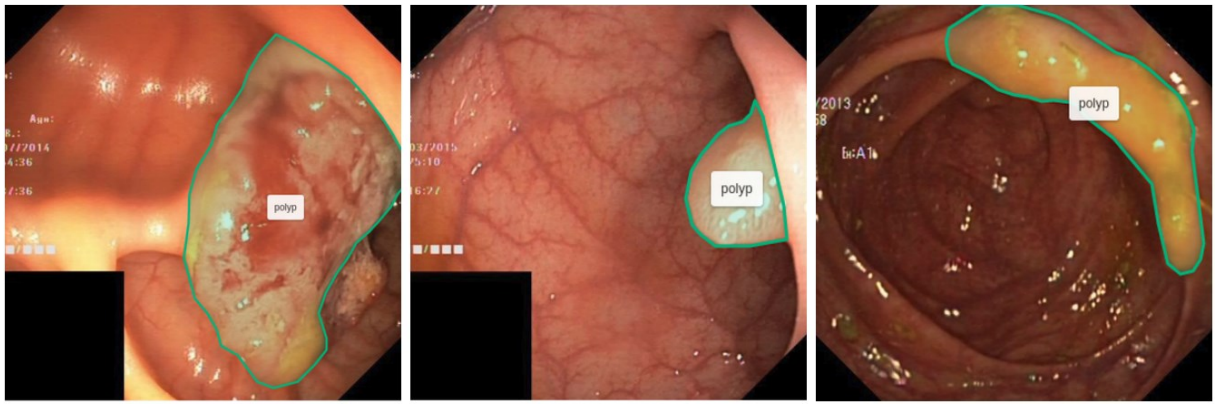


Figure 3.1 Example images from the Kvasir-SEG dataset with additional green marking of the polyp region.

As reported in the related article, the creation of this dataset was intended to overcome the lack of open-access datasets for comparable evaluations in this field. In fact, prior to Kvasir-SEG there were very few easily accessible datasets with scarce data, see for example CVC-ColonDB and CVC-ClinicDB in [21], ASU-Mayo Clinic Colonoscopy Video Database [22] and ETIS-Larib Polyp DB [23].

Kvasir-SEG is based on the previous Kvasir [24] but unlike this one it contains 1000 images with their corresponding 1-bit colour depth masks (white foreground and black background) manually annotated by a doctor.

In this work, as with most of the work reported in the literature - see for example [25] and [26] - the 1000 images in the dataset are divided as follows: 880 images make up the training set, while 120 images make up the validation set.

3.2 Deeplab

The semantic segmentation architecture used in this work is DeepLabV3+ [27], released in 2018 by Google. DeepLabV3+ still represents one of the state-of-art architectures for semantic segmentation; its implementation is open source and articles related to its inner workings are publicly available on Arxiv.

The DeepLabV3+ architecture is part of the larger DeepLab family: a series of encoder-decoder models designed by Google since 2015 that includes DeepLabv1, DeepLabv2, DeepLabv3 and DeepLabv3+.

The DeepLabv1 and DeepLabv2 architectures are very similar, they both use Deep Convolutional Neural Networks (DCNNs) with Atrous Convolution and Fully Connected Conditional Random Field (CRF). In DeepLabV2 new DCNNs are implemented, and a technology called Atrous Spatial Pyramid Pooling (ASPP) is added, see [28],[20] for more information.

DeepLabv3 [29] adds image-level features [30], [31] and batch normalization [32] to the ASPP module to facilitate training.

DeepLabV3+ is the latest version of the DeepLab models, it extends DeepLabV3 by including a decoder module to refine segmentation results (especially along object boundaries) and in addition it is possible to arbitrarily control encoder characteristics.

DeepLabV3+ uses the classical encoder-decoder architecture, where the encoder (also called backbone) is usually a pre-trained network that extracts features from a high-resolution image and transforms it into a vector of lower resolution features; while the decoder has the task of making a “semantic projection” of the low-resolution features, learned from the encoder, into the higher resolution pixel space.

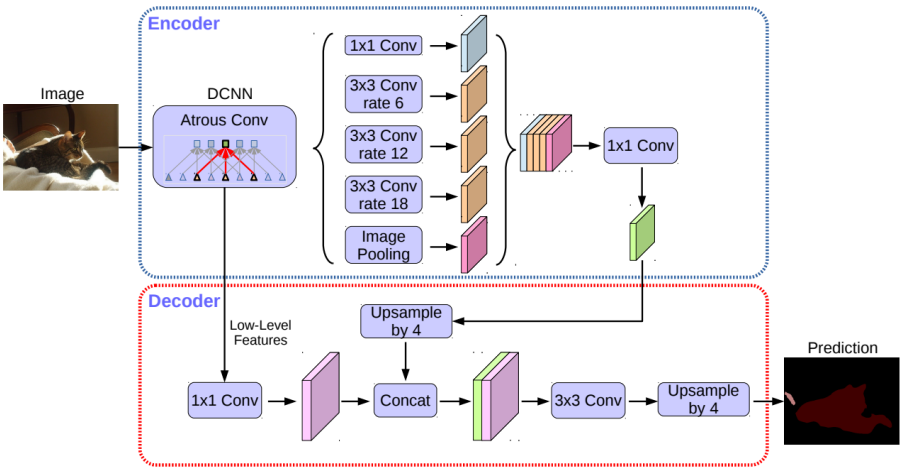


Figure 3.2 Internal structure of DeepLabV3+.

Thus, the key operation of this state-of-the-art semantic segmentation architecture is atrous convolution, which offers a faster and more efficient alternative to standard convolution by allowing the widening of the visual field of filter without increasing parameters and computational costs.

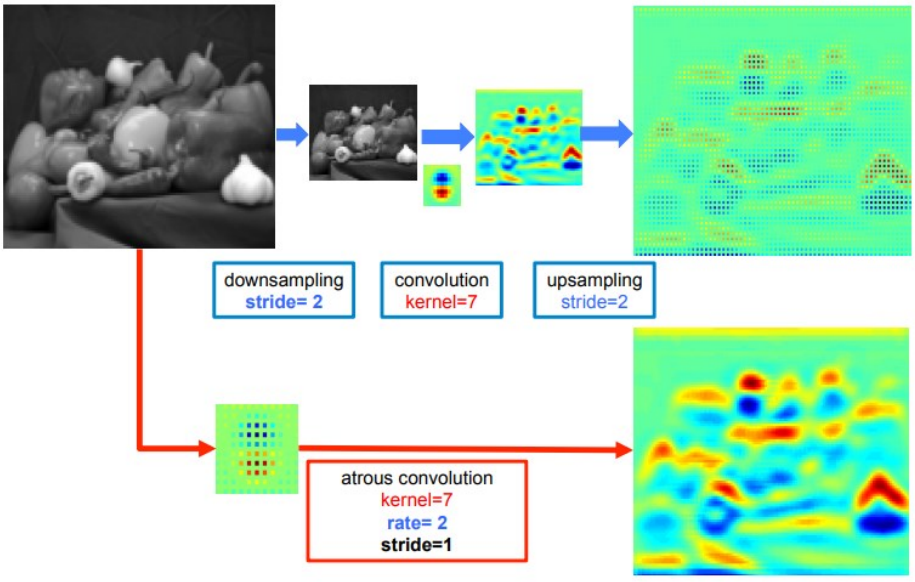


Figure 3.3 Example of comparison between standard convolution and atrous convolution in 2-D [20].

3.3 Approach Overview

The following sections contain descriptions of all data augmentation methods developed and tested in order to increase the size of the starting dataset to improve segmentation performance.

The proposed methods are grouped into families according to their operations, and the five families reported are:

1. Frequency Mixup
2. Frequency filtering
3. Elastic
4. Change background
5. Tone mapping

The operations described in the various sections are also applied to the relative image labels.

Most of the proposed methods use a pre-computed 880x880 similarity matrix containing the degree of similarity between the images of the training set. The degree of similarity between images was calculated using the 2-D correlation coefficient (corr2 in MATLAB) due to its efficiency compared to the other metrics (es: Structural Similarity Index, Mean-Squared Error, …). See more detail in [33].

The 2-D correlation coefficient is calculated as follows:

$$r = \frac{\sum_m \sum_n (A_{mn} - \bar{A})(B_{mn} - \bar{B})}{\sqrt{(\sum_m \sum_n (A_{mn} - \bar{A})^2)(\sum_m \sum_n (B_{mn} - \bar{B})^2)}} \tag{3.1}$$

where:

- A_{mn} is the intensity of the (m, n) pixel in the image A
- B_{mn} is the intensity of the (m, n) pixel in the image B
- $\bar{A} = mean2(A)$ = average intensity of image A
- $\bar{B} = mean2(B)$ = average intensity of image B

3.4 Frequency mixup

All methods belonging to this family use of the 2-D Discrete Fourier Transform (DFT) and its inverse, calculated through the Fast Fourier Transform algorithm.

The 2-D Discrete Fourier Transform Y of a matrix X of size MxN pixels is defined as:

$$Y(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} X(x,y)\, e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})} \tag{3.2}$$

And its inverse as:

$$X(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} Y(u,v)\, e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})} \tag{3.3}$$

where:

- j is the imaginary unit
- $u, x \in [0,\ M-1]$
- $v, y \in [0,\ N-1]$

3.4.1 FFT mixup 1

This method replaces the values that compose a 4x4 square in the centre of the shifted DFT of the original image with the respective values of a shifted DFT of a random image for all 3 channels, for 3 random images.

Pseudocode:

ALGORITHM 1: FFT MIXUP 1 DATA AUGMENTATION

Input: original dataset $(x, y) \in \{X, Y\}$
For n = 1,…3
  $x_{random} \leftarrow$ random image from X
  For i = each channel of rgb image x
    $x_{freq} \leftarrow F(x[:,:,i])$
    $x_{freq\_rand} \leftarrow F(x_{random}[:,:,i])$
    Shift zero-frequency components of x_{freq} and x_{freq_rand} to the center of the spectrum
    $x_{freq}[center\ square\ 4x4\ pixel] \leftarrow x_{freq\_rand}[center\ square\ 4x4\ pixel]$
    $x_{new}[:,:,i] \leftarrow F^{-1}(x_{freq})$
  End
End
Adding $(x_{new},\ y)$ to $\{X, Y\}$

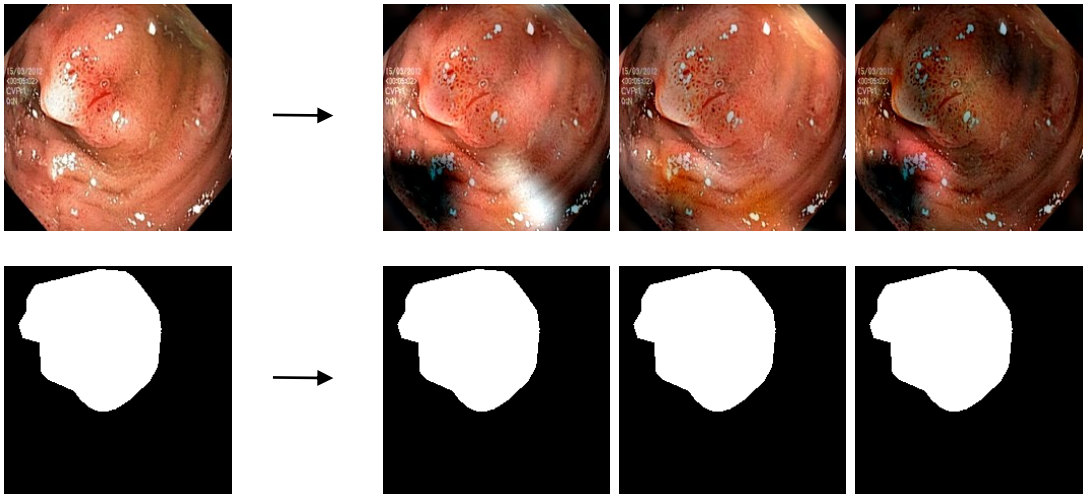


Figure 3.4 Example of FFT mixup 1 method application.

3.4.2 FFT mixup 2

This method is very similar to the previous one, but instead of replacing the FFT values of 3 random images it takes the first 3 images most similar to the original one.

The pseudocode of the algorithm is very similar to that in 3.4.1.

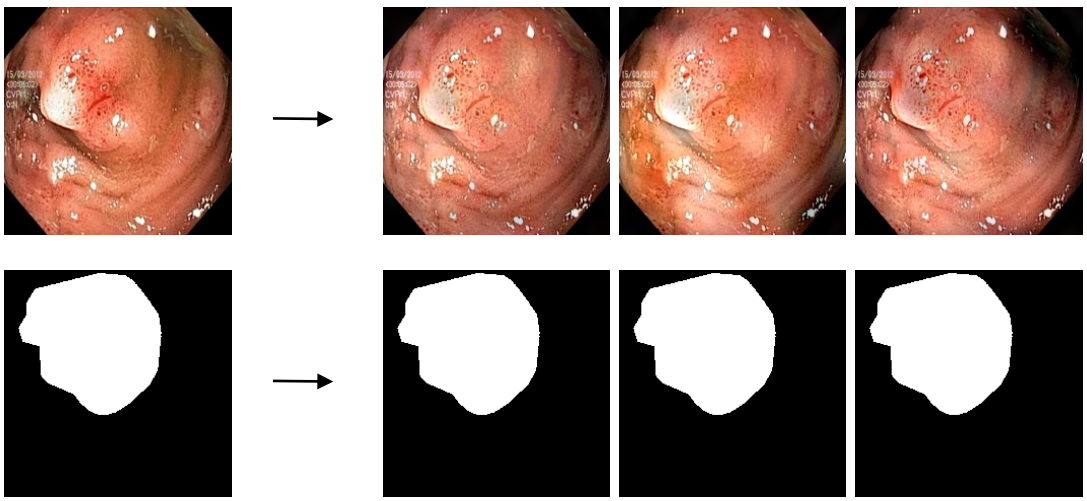


Figure 3.5 Example of FFT mixup 2 method application.

3.4.3 FFT mixup 3

This method changes the values of two vertical bands in the FFT of the current image with the respective values of the FFT of the similar image, for the first 3 most similar images. (Note: the bands have their centres in the siz/4 and (siz/4 * 3) columns and are 20*2 wide).

The pseudocode of the algorithm is very similar to that in 3.4.1.

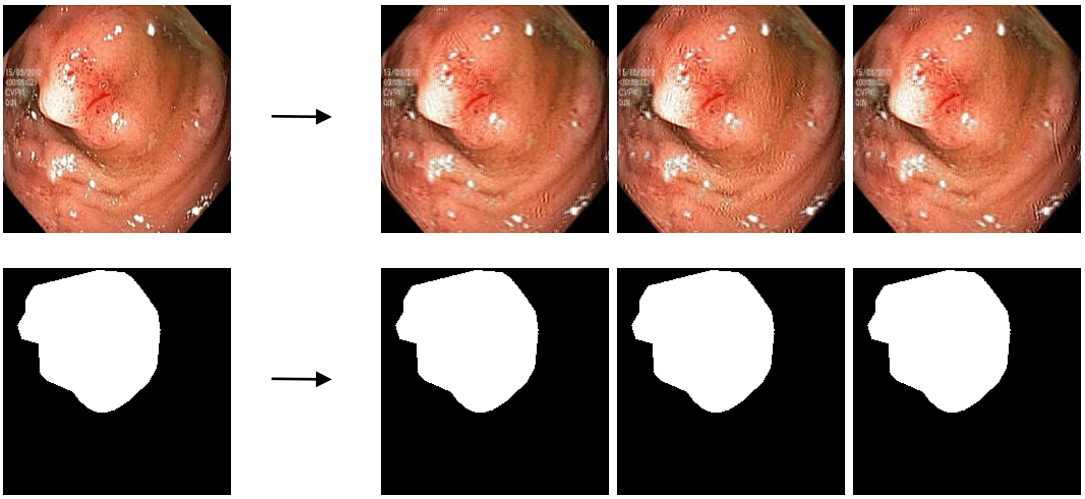


Figure 3.6 Example of FFT mixup 3 method application.

3.4.4 FFT mixup 4

For each image in the dataset, this technique considers the first 10 most similar images, then for each channel it does the FFT and based on the percentage of similarity between the original image and its similar image, it modifies each pixel with 5% probability, replacing its value with the average between the current value and the pixel value of the FFT of the similar image weighted with the percentage of similarity

Pseudocode:

ALGORITHM 2: FFT MIXUP 4 DATA AUGMENTATION

Input: original dataset $(x, y) \in \{X, Y\}$

1 S = {set of 10 most similar images} = $\{s_1, s_2, \ldots s_{10}\}$

2 V = {set that contains 10 respective degrees of similarity of images in S} = $\{v_1, v_2, \ldots v_{10}\}$

For $s_j \in S, v_j \in V$
  For i = each channel of rgb image x
    $x_{freq} \leftarrow F(x[:,:,i])$
    $x_{freq\_sim} \leftarrow F(s_j[:,:,i])$
    $m =$ random matrix with the same size as x where each pixel has 5% to be 1 and 95% to be 0.
    $$x_{freq}(m == 1) = \frac{(1 - v_j) * x_{freq}(m == 1) + v_j * x_{freq_{sim}}(m == 1)}{2}$$
    $x_{new}[:,:,i] = F^{-1}(x_{freq})$
  End
End
Adding $(x_{new},\ y)$ to $\{X, Y\}$

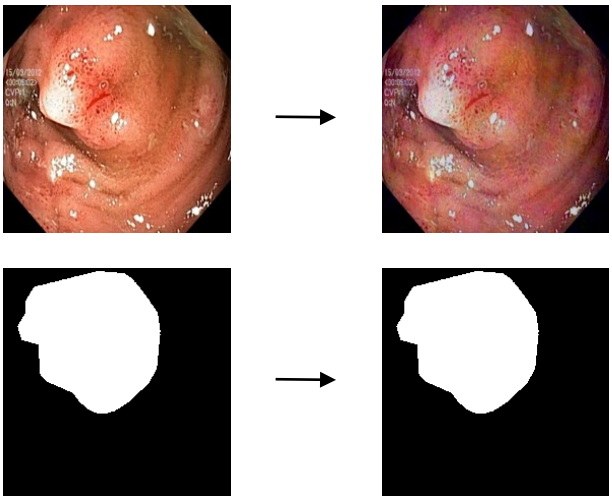


Figure 3.7 Example of FFT mixup 4 method application.

3.4.5 FFT mixup 5

Same technique as described in 3.4.1 but taking the first 5 most similar images and substituting a 3x3 square in the centre of the FFT.

See the pseudocode in 3.4.1.

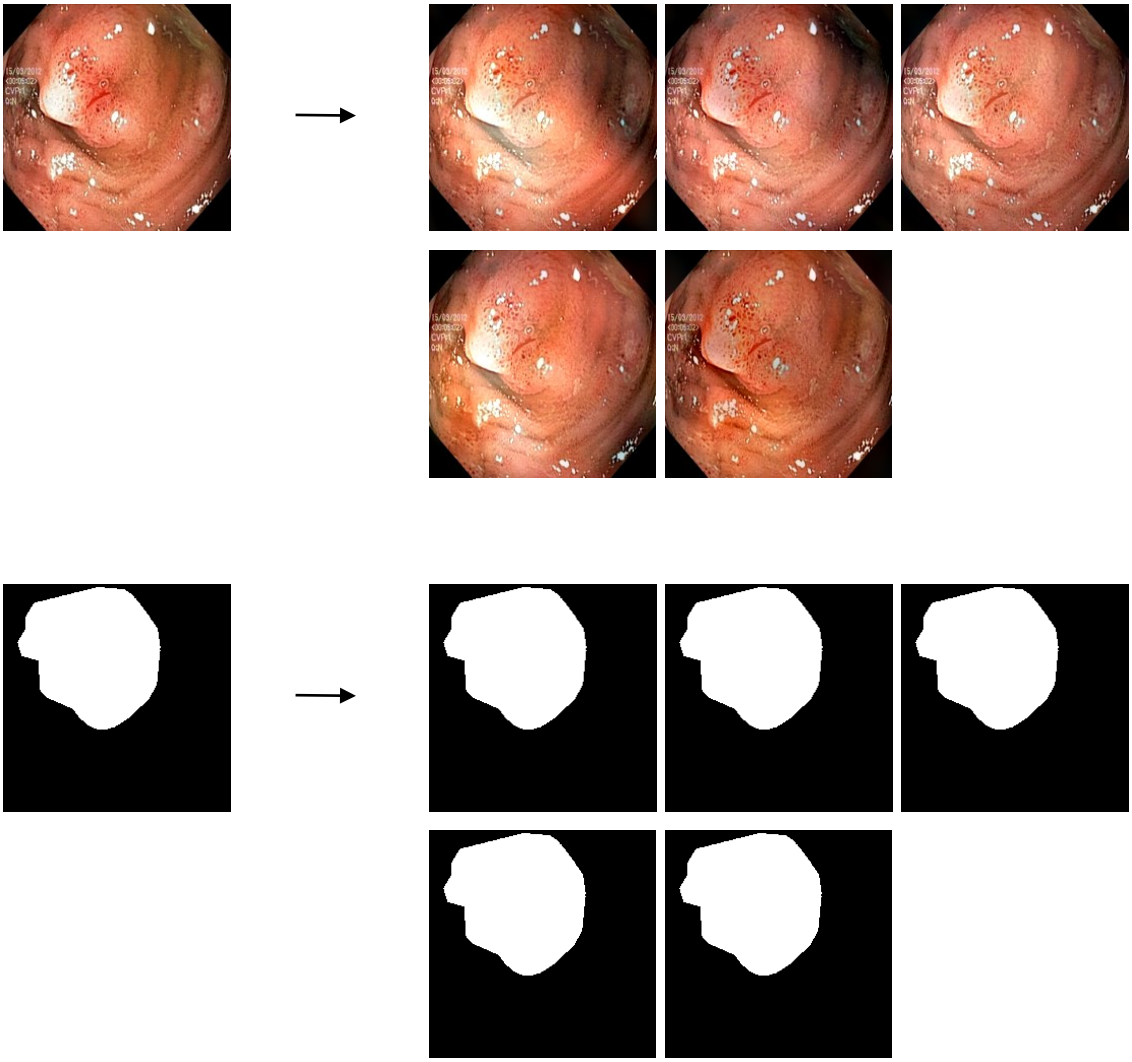


Figure 3.8 Example of FFT mixup 5 method application.

3.5 Frequency filtering

3.5.1 Frequency filtering [34]

Implementation of the frequency filtering method presented in the article [34] with $r_0 = 2$, $\alpha = 50, l = 60$ and theta0 angle of the shape mask different for all channels of the image.

Algorithm 1 Frequency domain data augmentation

Input: original dataset $(X, y) \in \{\mathcal{X}, \mathcal{Y}\}$, and other mask parameters r_0, l, and α

if dimension of $X = 3$ **then**
  $X^{\text{temp}} \leftarrow X[:, :, 0]$
end if
if dimension of $X = 1$ **then**
  $X^{\text{temp}} \leftarrow X$
end if
$X^{\text{freq}} \leftarrow \mathcal{F}(X^{\text{temp}})$
$\theta_0 \leftarrow$ randomly sampling from $h(\theta; X^{\text{freq}})$
$r_l, r_h, \theta_l, \theta_h \leftarrow r_0, r_0 + l, \theta_0 - \frac{\alpha}{2}, \theta_0 + \frac{\alpha}{2}$
$X^{\text{reject}} \leftarrow M(\theta, \alpha, l, r) \otimes X^{\text{freq}}$
$X^{\text{new}} \leftarrow \mathcal{F}^{-1}(X^{\text{reject}})$
$X^{\text{new}}[:, :, i] \leftarrow X^{\text{new}}$ ▷ for $i = 0, 1, 2$
Adding (X^{new}, y) to $\{\mathcal{X}, \mathcal{Y}\}$
return new images $X^{\text{new}} \in \mathcal{R}^{C \times C \times 3}$

Figure 3.9 Pseudocode reported in [34]

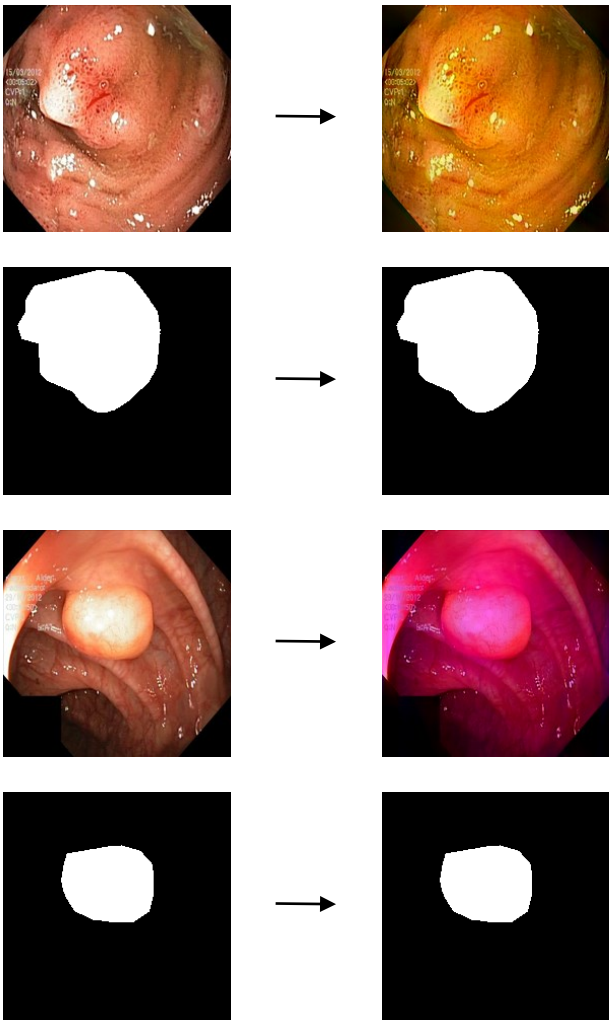


Figure 3.10 Example of Frequency filtering [34] method application.

3.6 Elastic

This family uses the techniques presented in APP9 of article [35] in order to evaluate the efficiency of this type of deformation. For a better description of the relative procedure used, see the individual sections.

3.6.1 Elastic1

The image created by this method uses ElasticDeformation, a technique based on the Albumentations library (available at https://albumentations.ai/ (accessed 05/06/22)

- type = 'log'
- alpha value = 3000

As reported in [35], this method applies a randomly generated displacement field to all pixel of image. The displacement field is defined as:

$$\Delta_x(x, y) = rand(-1, +1) \tag{3.4}$$

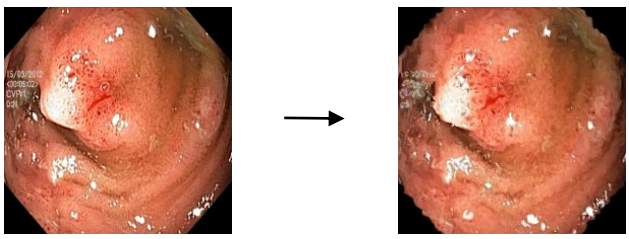
$$\Delta_y(x, y) = rand(-1, +1) \tag{3.5}$$

where $rand(-1, +1)$ represents a random value extracted from the standard uniform distribution in [-1, 1].

Next, three low-pass filters are then applied to the generated horizontal and vertical displacement fields:

- circular averaging filter
- rotationally symmetric Gaussian low-pass filter
- rotationally symmetric Laplacian of Gaussian filter

Finally, before being applied to the original image, each filtered displacement matrix is multiplied by an $alpha$ (α) value.



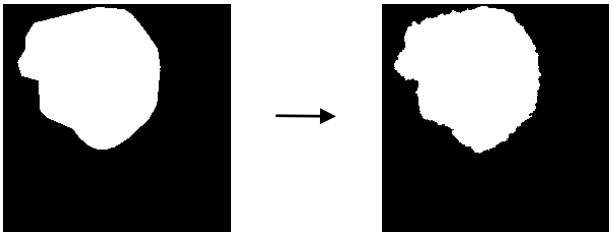


Figure 3.11 Example of elastic1 method application.

3.6.2 Elastic2

Same as 3.6.1 (so always type = 'log') but with 3 different alpha values: 1000, 2000, 3000.

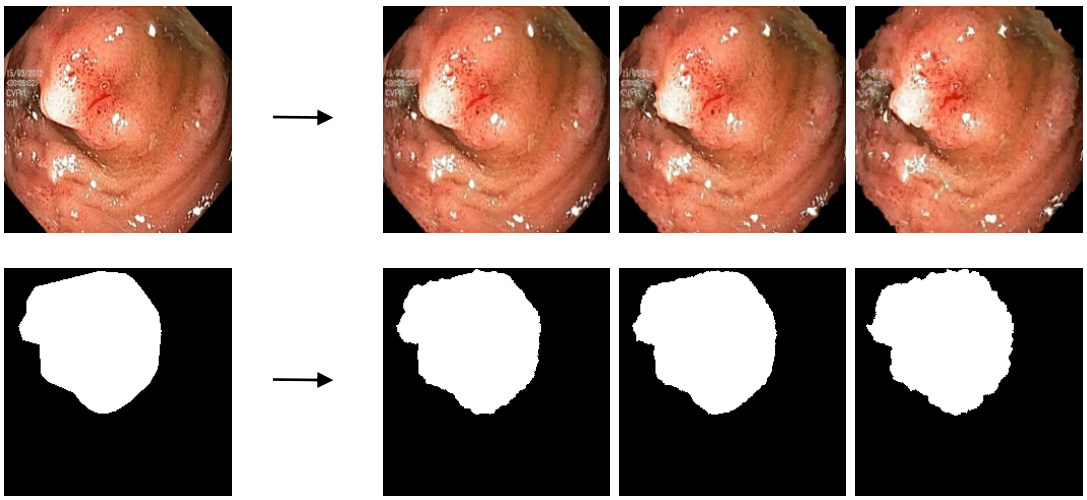


Figure 3.12 Example of elastic2 method application.

3.6.3 Elastic3

This method generates 6 new images:

- Elastic deformation KU with alpha value 7000
- Elastic deformation KU with alpha value 10000
- Elastic deformation KU with alpha value 13000
- Elastic Deformation SC type 'gauss' and alpha value 3000
- Elastic Deformation SC type 'disk' and alpha value 3000
- Elastic Deformation SC type 'log' and alpha value 3000

As reported in [35], the elastic deformation KU method differs from Elastic Deformation SC (used in 3.6.1 and 3.6.2) in the definition of the randomly generated displacement field:

$$\Delta_x(x, y) = \alpha * rand(-1, +1) \tag{3.6}$$

$$\Delta_y(x, y) = \alpha * rand(-1, +1) \tag{3.7}$$

where $rand(-1, +1)$ represents a random value extracted from the standard uniform distribution in [-1, 1] and α is a scaling factor.

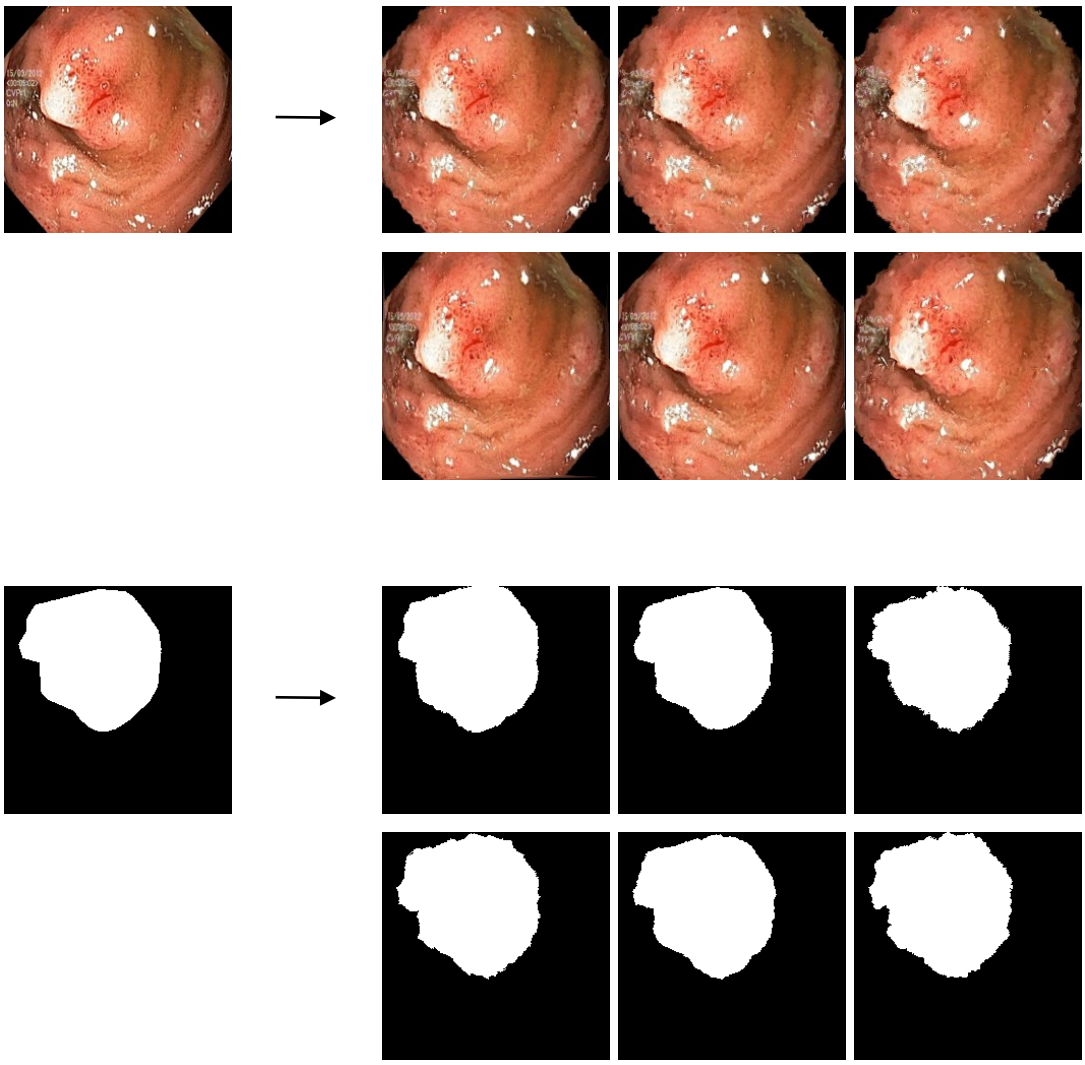


Figure 3.13 Example of elastic3 method application.

3.7 Change background

This family consists of methods that attempt to cut and paste polyps from one image to another, enhancing the simple cut-and-paste with constraints or procedures in order to improve the harmony of the resulting image.

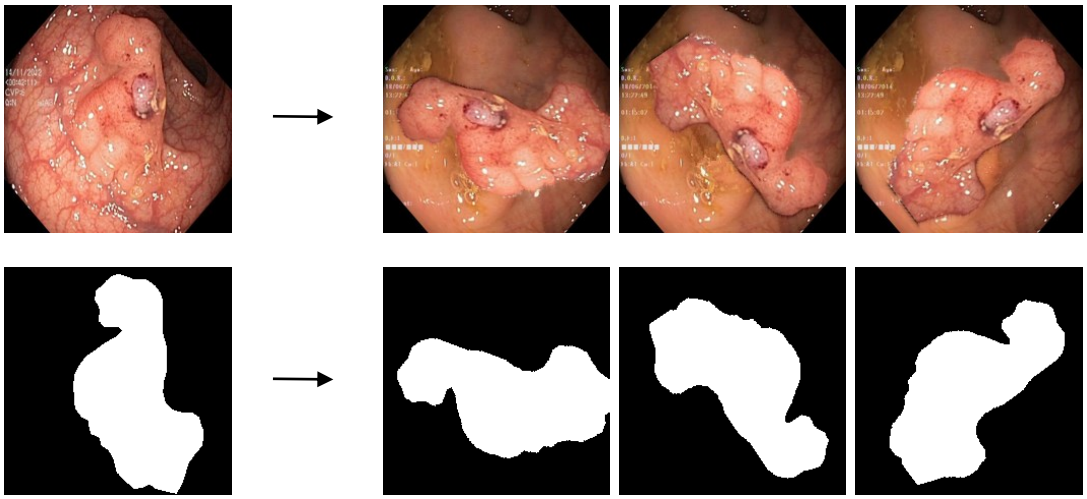
3.7.1 Change background 1

We can summarise the operations performed by this method in these 4 steps:

1. Considers the most similar image to the current image, reconstructs the region where the polyp is present in the selected similar image with inpaintCoherent and inpaintExemplar.
2. Pastes the polyp contained in the current image into the reconstructed background of the similar image at the position where the removed polyp was present.
3. In addition to the polyp, the 15-pixel-wide edge/contour of the polyp shape is also taken into account, in detail: a weighted average is made according to the function $y(x) = e^{-\frac{x}{2}}$ where x represents the distance to the polyp shape.
   The pixels of the edge are then calculated as:

$$y(x) * original\_pixel + \big(1 - y(x)\big) * pixel\_background \tag{3.8}$$

4. The pasted polyp is rotated at random angles 3 times, thus generating 3 new images.



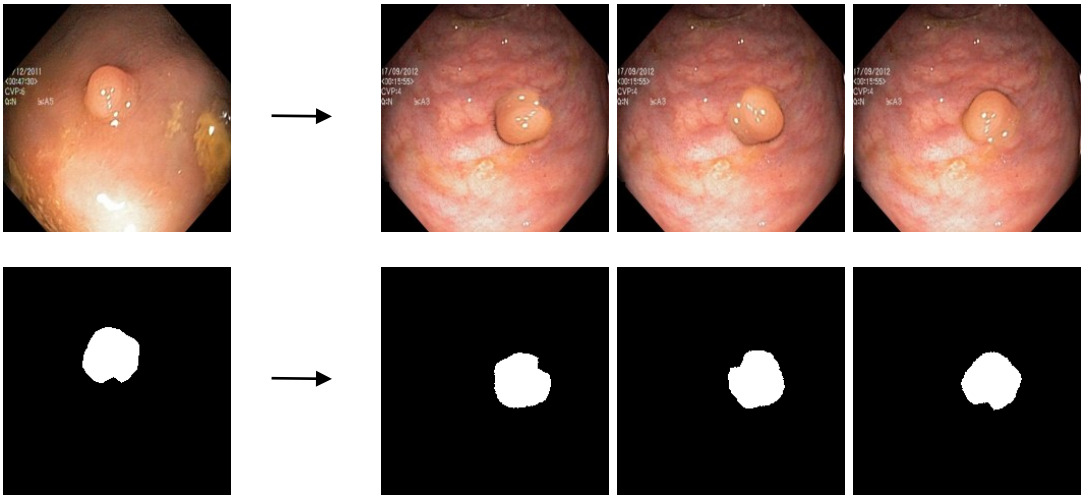
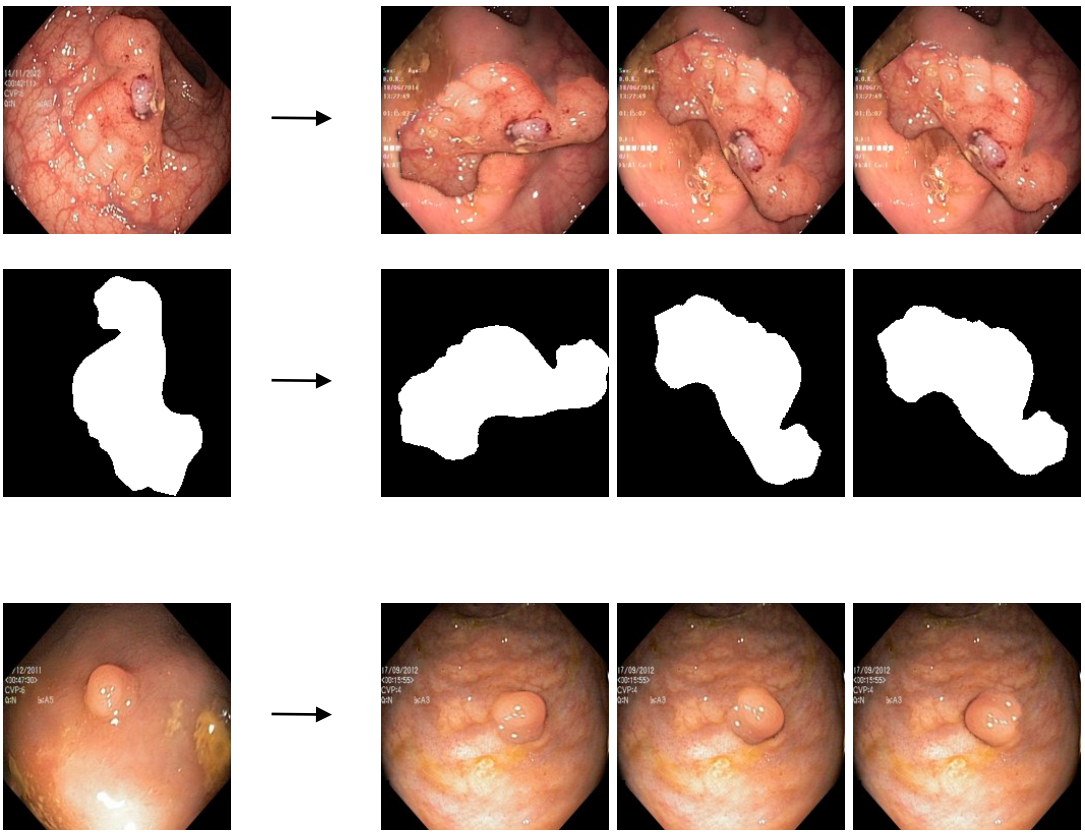


Figure 3.14 Example of change background 1 method application.

3.7.2 Change background 2

Algorithm similar to 3.7.1, but with the addition of a Reinhard tone mapping in the similar image found used as a background.



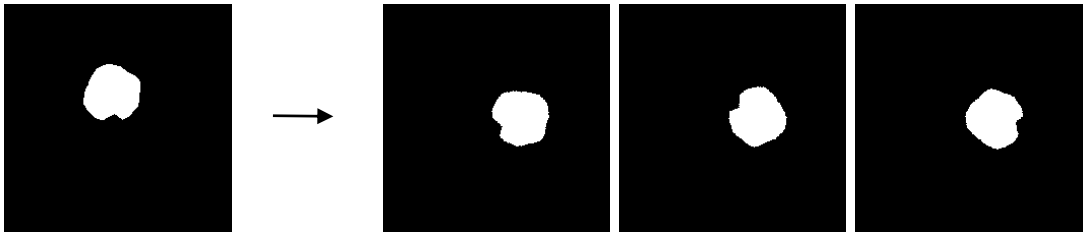


Figure 3.15 Example of change background 2 method application.

3.7.3 change background 3

Same as 3.7.1, but instead of only considering the image most similar to the current one, this method takes the image with the closest possible polyp size among the most similar images. In addition, the polyp that will pasted must be smaller than 150x150 and the reconstructed region in the found similar image must be smaller than 120x120.

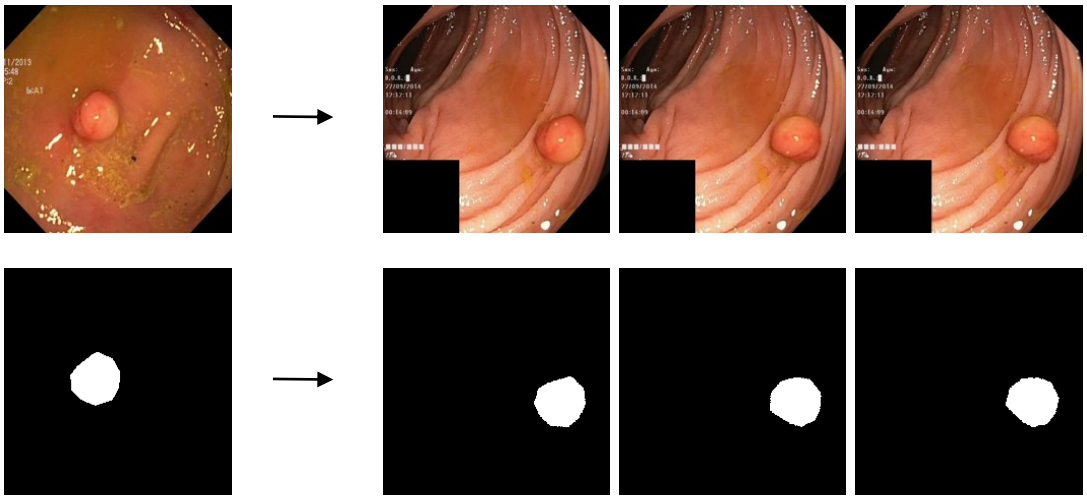


Figure 3.16 Example of change background 3 method application.

3.8 Tone mapping

This last family of techniques is based on the Stain Normalisation toolbox created by Nicholas Trahearn and Adnan Khan: a collection of MATLAB implementations of several existing techniques for stain normalisation of histological images and recently proposed stain normalisation algorithms such as [36]

(available online at https://warwick.ac.uk/fac/cross_fac/tia/software/sntoolbox/, accessed 05/06/22)

In particular, the methods in this family use the following techniques:

- RGB Histogram Specification
- Reinhard (more details in [37])
- Macenko (more details in [38])

3.8.1 Foreground similar mapping

This method generates the new image by Reinhard tone mapping with the most similar image, only of the polyp.

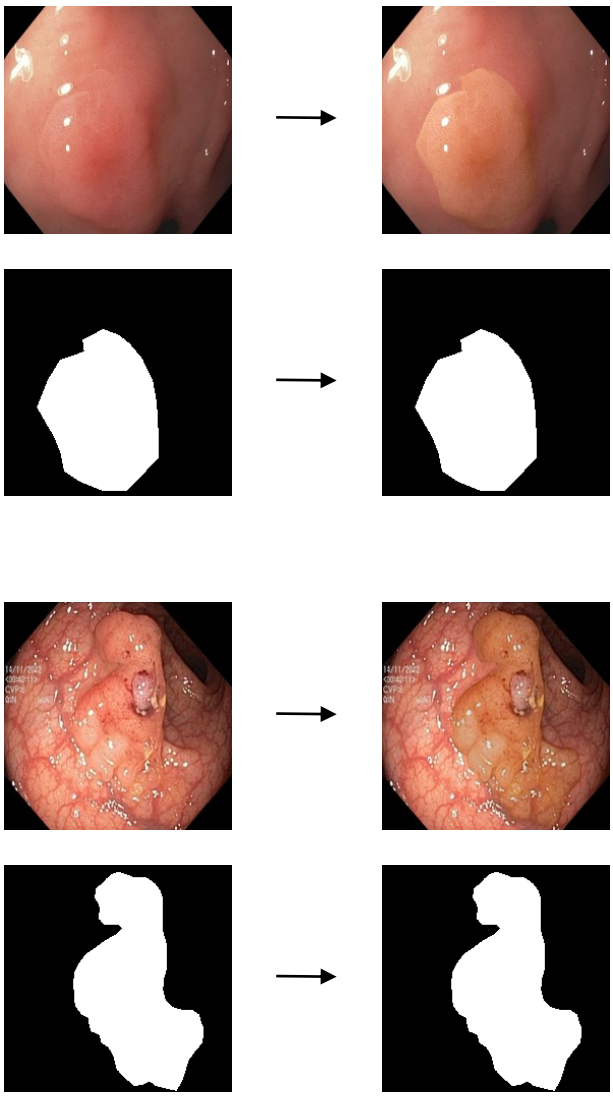


Figure 3.17 Examples of foreground similar mapping method application.

3.8.2 Background similar mapping

Same as 3.8.1 but mapping is only done on the background (no polyp).

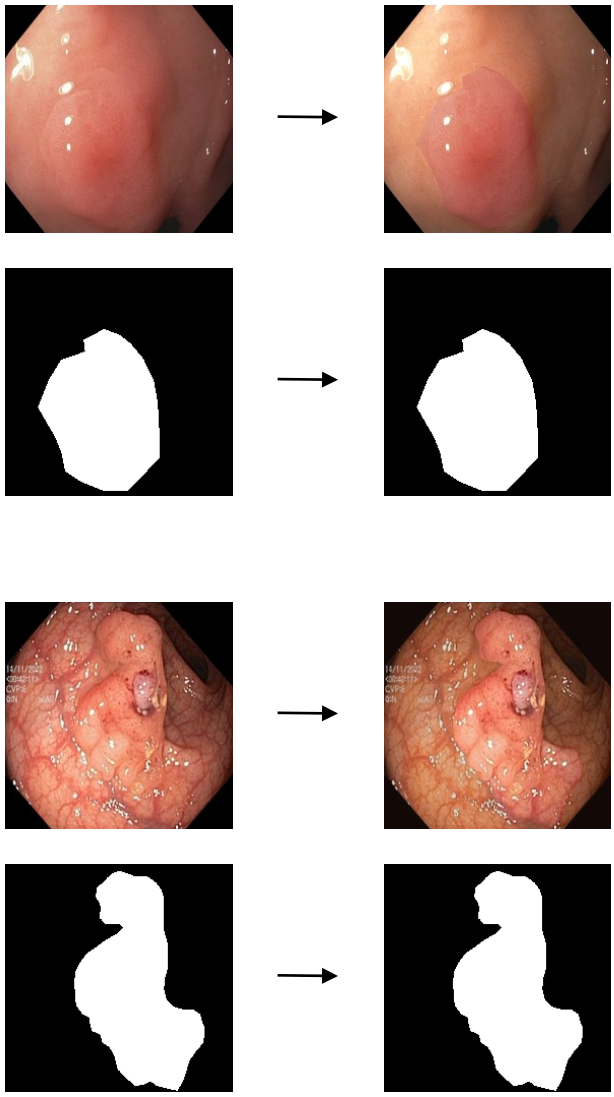


Figure 3.18 Examples of background similar mapping method application.

3.8.3 Background random mapping 3

This method generates 3 new images by doing 3 different tone mappings respectively: Macenko, Reinhard, RGBHist, each mapping with a random image, only on the background.

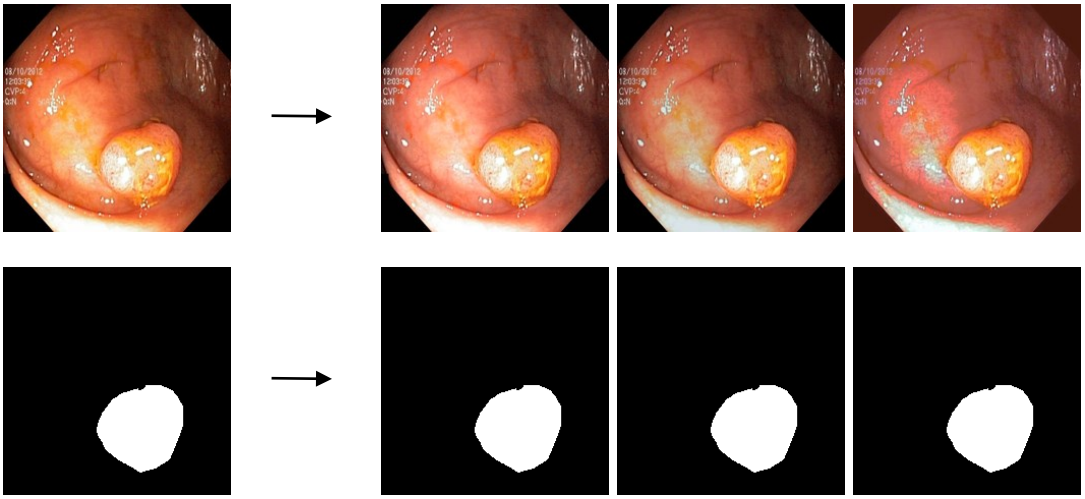


Figure 3.19 Examples of background random mapping 3 method application.

3.8.4 Background similar mapping 3

This method generates 3 new images by doing 3 different tone mappings respectively: Macenko, Reinhard, RGBHist, each mapping with the image most similar to the current one, only on the background.

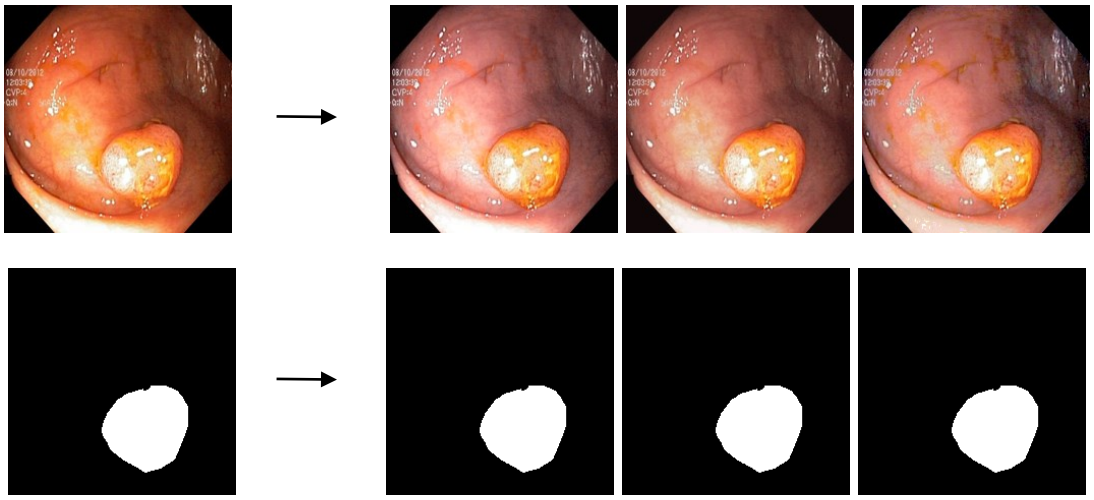


Figure 3.20 Examples of background similar mapping 3 method application.

3.8.5 Foreground similar mapping 3

This method generates 3 new images by doing 3 different tone mappings respectively: Macenko, Reinhard, RGBHist, each mapping with the image most similar to the current one, only on the polyp.

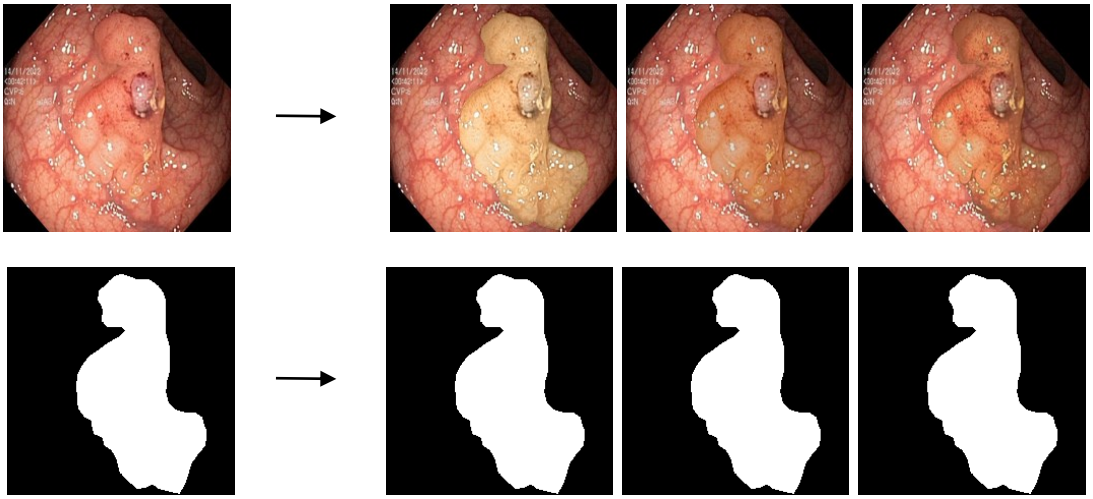


Figure 3.21 Examples of foreground similar mapping 3 method application.

Chapter 4

Experimental results

4.1 Evaluation metrics

In order to estimate the network's performances, after the application of the various data augmentation techniques previously introduced, some standard metrics are used.

All metrics described below are based on the elements that make up the confusion matrix, which are:

- True Positive (TP): a positive pattern is correctly classified as positive.
- True Negatives (TN): a negative pattern is correctly classified as negative.
- False Positives (FP): a negative pattern is wrongly classified as positive.
- False Negatives (FN): a positive pattern is wrongly classified as negative.

Where, in this work, the pattern is the single pixel and the positives or negatives are represented by pixels that actually make up the polyps or not in the ground truth masks.

4.1.1 Accuracy

Accuracy is the simplest and most intuitive metric, and it is defined as the ratio between correctly predicted observations and the total observations, therefore in this work as the ratio of correctly classified pixels to the total number of pixels of all images.

Formally, it is defined as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4.1}$$

The problem with this metric is that it only provides a reliable assessment when we have a symmetrical dataset, i.e. without a severe class imbalance, in fact it is always a good practice to consider other metrics to better evaluate the performance of the model.

4.1.2 Precision

Precision is literally the ratio between the true positives and all positives.

In this problem, it gives the percentage of pixels correctly classified as true positives out of all those classified as positives (including false positives).

Mathematically:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.2}$$

4.1.3 Recall

Recall indicates the ratio between the total number of correctly classified pixels and the total number of pixels that the system would have to classify correctly if it worked perfectly, it is defined as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.3}$$

4.1.4 F2-score

It is a metric that combines precision and recall, precisely is an instance of the Fbeta-measure with a beta value of 2. F2-score has the effect of increase the importance of recall and lowering the importance of precision, so it puts more attention on minimizing false negatives than minimizing false positives.

It is mathematically defined as:

$$\text{F2} - \text{score} = \frac{5 \cdot \text{Precision} \cdot \text{Recall}}{4 \cdot \text{Precision} + \text{Recall}} = \frac{TP}{TP + 0.2 \cdot \text{FP} + 0.8 \cdot \text{FN}} \tag{4.4}$$

4.1.5 Intersection over Union (IoU)

This metric is another standard metric to evaluate a segmentation method, in particular it measures the ratio of the overlapping and union area between the ground truth map B and the predicted mask A produced by the classification model.

Formally, this is defined by the following equation:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{|\text{A} \cap \text{B}|}{|\text{A} \cup \text{B}|} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{4.5}$$

This measure best describes the quality of the resulting segmentation.

4.1.6 Dice coefficient

The Dice coefficient is another standard metric that is calculated as ratio between twice of the overlap area of the predicted mask A and ground truth mask B divided by the total number of pixels:

$$Dice = \frac{2 \cdot |A \cap B|}{|A| + |B|} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{4.6}$$

4.2 Explanation of experiments

The results reported in this paper emerge from five experiments in which the methods proposed in Chapter 3 were all tested with the following configuration:

- Input size = 224
- Rete = resnet18
- Loss = dice
- Initial learning rate = 0.01
- Number of epochs = 20
- Mini batch size = 25
- Optimizer = SGDM
- Learning Rate Drop Period = 5
- Learning Rate Drop Factor = 0.2

- Momentum = 0.9
- L2Regularization = 0.005
- Shuffle training images every epoch

The first experiment aims to evaluate the performance of individual methods by training DeepLabv3+ with the configuration just described. In particular, the dataset is divided as described in section 3.1 and the new images generated by the respective methods are added to the training set.

The second experiment, on the other hand, is aimed at verifying how the application of a post-processing method can improve the performance of the network.

In particular, this method (called RandBasicAug from now on) consists of applying all the following basic data augmentation methods to the images generated by the presented methods, each with 30% probability:

- Random flipping
- Random rotation between -45° and 45°
- Random scaling between x0.8 and x1.2
- Random translation with $\Delta_x \in [-20, 20]$ and $\Delta_y \in [-50, 50]$
- Random horizontal shear in the range [-30, 30].
- Random synthetic noise:
    - salt & pepper with density = 0.05
    - gaussian with zero-mean and variance=0.01
    - gaussian with sigma=1+0.2*rand

  Please note: only one of the following noises is chosen (each has 33,3% of being chosen)
- Randomly alter color of pixels on:
    - Saturation
    - Brightness
    - Contrast
    - Hue

  Please note: only one of the following noises is chosen (each has 25% of being chosen)

The remaining 3 experiments consist of evaluating the performance of the combinations of proposed methods with RandBasicAug post-processing. In detail, these last experiments combine the methods first in pairs, then in triples and finally in quadruples. Please note that the combinations were created with the best performing methods, as can be seen in the tables in the following sections.

Please note: all code was developed in MATLAB.

4.3 Results by category

In order to evaluate the efficiency of the proposed data augmentation techniques, we must first consider that with the configuration described in section 4.2, the dataset alone, without data augmentation, performs as follows:

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
Only kvasir	0.723	0.812	0.817	0.867	0.832	0.945

Table 4.1 Performance without data augmentation

The following tables contain the results of the first experiment, where the methods with the best performance are highlighted in bold:

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
FFT mixup 1	0.755	0.841	0.850	0.863	0.872	0.951
FFT mixup 2	0.761	0.846	**0.856**	0.868	**0.876**	**0.952**
FFT mixup 3	0.749	0.835	0.846	0.855	0.864	0.952
FFT mixup 4	0.758	0.845	0.849	0.875	0.861	0.952
FFT mixup 5	**0.766**	**0.848**	0.850	**0.882**	0.863	0.952

Table 4.2 Performance of frequency mixup techniques

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
Frequency filtering [34]	**0.760**	**0.846**	**0.857**	**0.859**	**0.877**	**0.952**

Table 4.3 Performance of frequency filtering techniques

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
elastic1	0.752	0.839	**0.845**	0.862	**0.861**	0.954
elastic2	**0.760**	**0.842**	0.844	**0.872**	0.857	**0.955**
elastic3	0.750	0.835	0.844	0.856	0.859	0.953

Table 4.4 Performance of elastic techniques

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
change background 1	0.743	0.834	0.850	0.845	**0.875**	0.948
change background 2	**0.757**	**0.843**	**0.845**	**0.877**	0.861	**0.952**
change background 3	0.742	0.830	0.832	0.870	0.846	0.950

Table 4.5 Performance of change background techniques

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
foreground sim mapping	0.749	0.835	0.843	0.858	0.860	0.954
background sim mapping	0.738	0.825	0.829	0.873	0.846	0.949
background rand mapping 3	0.757	**0.843**	0.847	**0.880**	0.862	0.951
background sim mapping 3	**0.760**	0.843	0.850	0.871	0.868	0.953
foreground sim mapping 3	0.757	0.843	**0.857**	0.859	**0.880**	**0.955**

Table 4.6 Performance of tone mapping techniques

4.4 Results of proposed methods with RandBasicAug

This section contains the table with the results of the second experiment, as we can see the application of RandBasicAug as post processing generally improves the performance of the proposed methods:

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
FFT mixup 1	0.777	0.856	0.864	0.877	0.883	0.957
FFT mixup 2	0.787	0.863	0.870	0.883	0.886	0.959
FFT mixup 3	0.772	0.855	0.860	0.881	0.876	0.955
FFT mixup 4	0.762	0.848	0.852	0.881	0.865	0.952
FFT mixup 5	**0.799**	**0.871**	**0.878**	0.887	**0.892**	**0.961**
Frequency filtering [34]	0.754	0.841	0.848	0.873	0.865	0.952
elastic1	0.760	0.844	0.844	0.875	0.853	0.955
elastic2	0.773	0.855	0.862	0.881	0.878	0.956
elastic3	0.769	0.851	0.858	0.876	0.872	0.953
change background 1	0.756	0.841	0.847	0.871	0.862	0.952
change background 2	0.764	0.848	0.853	0.875	0.866	0.953
change background 3	0.749	0.834	0.833	**0.888**	0.846	0.953
foreground sim mapping	0.753	0.840	0.847	0.856	0.863	0.954
background sim mapping	0.765	0.851	0.852	0.880	0.862	0.954
background rand mapping 3	0.777	0.857	0.866	0.877	0.881	0.957
background sim mapping 3	0.765	0.845	0.853	0.873	0.872	0.954
foreground sim mapping 3	0.776	0.855	0.862	0.884	0.877	0.955

Table 4.7 Performance of proposed methods + RandBasicAug

4.5 Results of method compositions with RandBasicAug

This section reports the results obtained by training the configuration described in Section 4.2 by combining the methods presented in Chapter 3:

- Combination 1 ➔ FFT mixup 5 + FFT mixup 1
- Combination 2 ➔ FFT mixup 5 + FFT mixup 2
- Combination 3 ➔ FFT mixup 5 + FFT mixup 3
- Combination 4 ➔ FFT mixup 5 + FFT mixup 4
- Combination 5 ➔ FFT mixup 5 + Frequency filtering [34]
- Combination 6 ➔ FFT mixup 5 + change background 1
- Combination 7 ➔ FFT mixup 5 + change background 2
- Combination 8 ➔ FFT mixup 5 + change background 3
- Combination 9 ➔ FFT mixup 5 + elastic1
- Combination 10 ➔ FFT mixup 5 + elastic2
- Combination 11 ➔ FFT mixup 5 + elastic3
- Combination 12 ➔ FFT mixup 5 + foreground sim mapping
- Combination 13 ➔ FFT mixup 5 + background sim mapping
- Combination 14 ➔ FFT mixup 5 + background rand mapping 3
- Combination 15 ➔ FFT mixup 5 + background sim mapping 3
- Combination 16 ➔ FFT mixup 5 + foreground sim mapping 3

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
Combination 1	0.794	0.866	0.877	0.884	0.897	0.958
Combination 2	0.798	0.869	0.873	**0.899**	0.888	0.960
Combination 3	0.794	0.872	0.879	0.889	0.891	0.958
Combination 4	0.782	0.859	0.868	0.878	0.882	0.955
Combination 5	0.787	0.864	0.876	0.877	0.899	0.958
Combination 6	0.788	0.865	0.877	0.879	0.897	0.958
Combination 7	0.791	0.866	0.873	0.887	0.891	0.959
Combination 8	0.790	0.866	0.873	0.889	0.889	0.959
Combination 9	0.795	0.869	0.881	0.880	0.903	0.959
Combination 10	0.783	0.860	0.865	0.887	0.879	0.955
Combination 11	0.794	0.870	0.880	0.885	0.896	0.962
Combination 12	0.790	0.865	0.875	0.881	0.896	0.958
Combination 13	0.789	0.866	0.878	0.882	0.896	0.958
Combination 14	0.792	0.866	0.870	0.896	0.885	0.957
Combination 15	**0.804**	**0.877**	**0.891**	0.881	**0.912**	**0.962**
Combination 16	0.803	0.875	0.890	0.883	0.909	0.961

Table 4.8 Performance of combinations of proposed methods step 1 with RandBasicAug

- Combination 18 ➜ FFT mixup 5+background sim mapping 3+FFT mixup 1
- Combination 19 ➜ FFT mixup 5+background sim mapping 3+FFT mixup 2
- Combination 20 ➜ FFT mixup 5+background sim mapping 3+FFT mixup 3
- Combination 21 ➜ FFT mixup 5+background sim mapping 3+FFT mixup 4
- Combination 22 ➜ FFT mixup 5+background sim mapping 3+Frequency filtering [34]
- Combination 23 ➜ FFT mixup 5+background sim mapping 3+elastic1
- Combination 24 ➜ FFT mixup 5+background sim mapping 3+elastic2
- Combination 25 ➜ FFT mixup 5+background sim mapping 3+elastic3
- Combination 26 ➜ FFT mixup 5+background sim mapping 3+change background 1
- Combination 27 ➜ FFT mixup 5+background sim mapping 3+change background 2
- Combination 28 ➜ FFT mixup 5+background sim mapping 3+change background 3
- Combination 29 ➜ FFT mixup 5+background sim mapping 3+foreground sim mapping
- Combination 30 ➜ FFT mixup 5+background sim mapping 3+background similar mapping
- Combination 31 ➜ FFT mixup 5+background sim mapping 3+background rand mapping 3
- Combination 32 ➜ FFT mixup 5+background sim mapping 3+foreground sim mapping 3

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
Combination 18	0.796	0.869	0.875	**0.892**	0.893	0.959
Combination 19	0.795	0.868	0.877	0.887	0.897	0.960
Combination 20	0.789	0.864	0.872	0.883	0.887	0.957
Combination 21	0.799	0.872	0.886	0.877	**0.906**	0.962
Combination 22	0.784	0.860	0.867	0.887	0.884	0.957
Combination 23	0.798	0.871	0.880	0.885	0.897	0.959
Combination 24	0.797	0.869	0.883	0.881	0.906	0.958
Combination 25	0.798	0.867	0.874	0.887	0.884	0.960
Combination 26	0.805	**0.878**	**0.888**	0.889	0.905	0.962
Combination 27	0.783	0.857	0.874	0.875	0.898	0.958
Combination 28	**0.808**	0.878	0.887	0.891	0.906	**0.964**
Combination 29	0.794	0.867	0.874	0.892	0.888	0.958
Combination 30	0.793	0.868	0.881	0.879	0.900	0.959
Combination 31	0.795	0.867	0.874	0.892	0.891	0.960
Combination 32	0.791	0.864	0.880	0.874	0.904	0.959
Combination 33	0.796	0.869	0.875	0.892	0.893	0.959

Table 4.9 Performance of combinations of proposed methods step 2 with RandBasicAug

- Combination 33 ➔ FFT mixup 5+background sim mapping 3 +change background 3+change background 2
- Combination 34 ➔ FFT mixup 5+background sim mapping 3 +change background 3+FFT mixup 4
- Combination 35 ➔ FFT mixup 5+background sim mapping 3 +change background 3+elastic1

Method name	IoU	Dice	F2-score	Precision	Recall	Accuracy
Combination 33	**0.808**	**0.880**	**0.890**	**0.890**	**0.907**	**0.963**
Combination 34	0.797	0.870	0.884	0.879	0.905	0.963
Combination 35	0.785	0.860	0.874	0.873	0.900	0.957

Table 4.10 Performance of combinations of proposed methods step 3 with RandBasicAug

4.6 Comparison with other data augmentation method

The following table contains performances reported in various works in the literature:

Method name	Backbone	IoU	Dice	F2-score	Precision	Recall	Accuracy
UNet [39]	-	0.471	0.597	0.598	0.672	0.617	0.894
ResUNet [39]	-	0.572	0.690	0.699	0.745	0.725	0.917
ResUNet++ [39]	-	0.613	0.714	0.720	0.784	0.742	0.917
FCN8 [39]	VGG 16	0.737	0.831	0.825	0.882	0.835	0.952
HRNet [39]	-	0.759	0.845	0.847	0.878	0.859	0.952
DoubleUNet [39]	VGG 19	0.733	0.813	0.821	0.861	0.840	0.949
PSPNet [39]	ResNet50	0.744	0.841	0.831	0.890	0.836	0.953
DeepLabv3+ [39]	ResNet50	0.776	0.857	0.855	0.891	0.862	0.961
DeepLabv3+ [39]	ResNet101	0.786	0.864	0.857	0.906	0.859	0.961
UNet [39]	ResNet34	0.810	0.876	0.862	0.944	0.860	0.968
ColonSegNet [39]	-	0.724	0.821	0.821	0.844	0.850	0.949
DDANet [40]	-	0.780	0.858	-	0.864	0.888	-
HarDNet-MSEG [41]	-	0.848	0.904	0.915	0.907	0.923	0.969

Table 4.11 State of art performance in literature

Please note that some performances reported in the table above refer to networks much deeper than ResNet18 used for the experiments reported.

Chapter 5

Conclusions

Semantic segmentation, i.e. the classification of each pixel of an image, is a very useful and important task in different fields and for different purposes, such as the early identification of severe diseases to reduce the risk of serious consequences in medical diagnosis.

This work again confirms how useful data augmentation techniques are in order to improve a segmentation system, focusing especially on colonoscopy examinations.

As concerns the presented methods, all of them improve (a little or a lot) the performance of polyp segmentation. In particular, as Table 4.7 shows, the use of RandBasicAug (described in section 4.2) as post-processing to the proposed methods generally improves performance.

Moreover, as the results in Tables 4.8, 4.9 and 4.10 show, using combinations of data augmentation techniques leads to better performance than using them individually, especially if they focus on and bring out different aspects of the images with the disadvantage, however, of having a longer training time.

The results in Table 4.6 confirm that the use of the pre-computed similarity matrix, and in general the development of techniques whose processing is between similar images performs slightly better in general.

In general, as can be seen from the tables in Chapter 4, it can be inferred that a larger dataset generates better results, without forgetting the quality of the images that constitute it.

5.1 Future Works

In future works, the best and most effective of presented methods will also be used in neural network ensembles to see if they can offer improvements in this field as well.

The effectiveness of the presented methods will be tested on other datasets, on other networks and with other loss functions.

In addition, the impact of using the similarity matrix, also generated using other metrics, will be studied in more depth.

The methods presented will then be studied and compared with other automatic data augmentation approaches, known as 'learned'.

Acknowledgments

Ringrazio innanzitutto il relatore *Prof. Ing. Loris Nanni* per avermi dato la possibilità di sviluppare ed approfondire questo interessante argomento, guidandomi attraverso la sua esperienza e disponibilità.

Il più grande e sentito ringraziamento va alla mia *Famiglia* per avermi permesso di intraprendere e concludere questo percorso di studi, ognuno, sostenendomi ed influenzandomi a suo modo ha reso possibile questo mio traguardo.

Un ringraziamento speciale va a mia sorella *Elena* e alla mia ragazza *Lisa*, per avermi ascoltato e sostenuto nelle mie scelte, stando sempre al mio fianco, anche nei momenti meno luminosi.

Infine, ringrazio tutte le persone, amici, compagni, professori che hanno contribuito alla mia crescita professionale ed umana.

Padova, 21 luglio 2022
A.D.

References

[1] J. Ferlay, H. R. Shin, F. Bray, D. Forman, C. Mathers, and D. M. Parkin, "Estimates of worldwide burden of cancer in 2008: GLOBOCAN 2008," *International Journal of Cancer*, vol. 127, no. 12, pp. 2893–2917, Dec. 2010, doi: 10.1002/IJC.25516.

[2] L. A. Torre, F. Bray, R. L. Siegel, J. Ferlay, J. Lortet-Tieulent, and A. Jemal, "Global Cancer Statistics, 2012," *CA Cancer J Clin*, vol. 65, pp. 87–108, 2015, doi: 10.3322/caac.21262.

[3] R. L. Siegel, K. D. Miller, and A. Jemal, "Cancer Statistics, 2015", doi: 10.3322/caac.21254.

[4] R. L. Siegel *et al.*, "Colorectal cancer statistics, 2017," *CA: A Cancer Journal for Clinicians*, vol. 67, no. 3, pp. 177–193, May 2017, doi: 10.3322/CAAC.21395.

[5] D. Jha *et al.*, "Kvasir-SEG: A Segmented Polyp Dataset," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11962 LNCS, pp. 451–462, Nov. 2019, doi: 10.48550/arxiv.1911.07069.

[6] S. Y. Feng *et al.*, "A Survey of Data Augmentation Approaches for NLP", Accessed: Apr. 04, 2022. [Online]. Available: https://github.com/styfeng/DataAug4NLP.

[7] Q. Wen *et al.*, "Time Series Data Augmentation for Deep Learning: A Survey".

[8] K. Ding, Z. Xu, H. Tong, and H. Liu, "Data Augmentation for Deep Graph Learning: A Survey".

[9] M. Decuyper, M. Stockhoff, S. Vandenberghe, al -, and X. Ying, "An Overview of Overfitting and its Solutions," *Journal of Physics: Conference Series*, vol. 1168, no. 2, p. 022022, Feb. 2019, doi: 10.1088/1742-6596/1168/2/022022.

[10] M. Koziarski and B. Cyganek, "Image Recognition with Deep Neural Networks in Presence of Noise-Dealing with and Taking Advantage of Distortions," *Integrated Computer-Aided Engineering*, vol. 1, pp. 1–14, 2017.

[11] S. Dodge and L. Karam, "Understanding How Image Quality Affects Deep Neural Networks".

[12] A. Kumar Boyat and B. Kumar Joshi, "Signal & Image Processing," *An International Journal (SIPIJ)*, vol. 6, no. 2, 2015, doi: 10.5121/sipij.2015.6206.

[13] D. Lu, Q. Weng, and Q. Weng{, "International Journal of Remote Sensing A survey of image classification methods and techniques for improving classification performance A survey of image classification methods and techniques for improving classification performance," 2007, doi: 10.1080/01431160600746456.

[14] Z. Zou, Z. Shi, Y. Guo, J. Ye, and S. Member, "Object Detection in 20 Years: A Survey".

[15] "The Evolution of Deeplab for Semantic Segmentation | by Beeren Sahu | Towards Data Science." https://towardsdatascience.com/the-evolution-of-deeplab-for-semantic-segmentation-95082b025571 (accessed Apr. 05, 2022).

[16] "Computer Vision: Everything You Need to Know." https://www.v7labs.com/blog/what-is-computer-vision#computer-vision-tasks (accessed Apr. 10, 2022).

[17] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, Nov. 2014, doi: 10.48550/arxiv.1411.4038.

[18] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, Nov. 2015, doi: 10.48550/arxiv.1511.00561.

[19] O. Ronneberger, P. Fischer, and T. Brox, "LNCS 9351 - U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015, doi: 10.1007/978-3-319-24574-4_28.

[20] L.-C. Chen, G. Papandreou, S. Member, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs", Accessed: Apr. 06, 2022. [Online]. Available: http://liangchiehchen.com/projects/

[21] "Databases – CVC-Colon." http://www.cvc.uab.es/CVC-Colon/index.php/databases/ (accessed Apr. 06, 2022).

[22] N. Tajbakhsh, S. R. Gurudu, and J. Liang, "Automated polyp detection in colonoscopy videos using shape and context information," *IEEE Transactions on Medical Imaging*, vol. 35, no. 2, pp. 630–644, Feb. 2016, doi: 10.1109/TMI.2015.2487997.

[23] J. S. Silva, A. Histace, O. Romain, X. Dray, and B. Granado, "Towards embedded detection of polyps in WCE images for early diagnosis of colorectal cancer," *International Journal of Computer Assisted Radiology and Surgery*, vol. 9, no. 2, pp. 283–293, 2014, doi: 10.1007/s11548.

[24] K. Pogorelov *et al.*, "Kvasir: A Multi-Class Image Dataset for Com-puter Aided Gastrointestinal Disease Detection," *ACM Reference format*, vol. 6, 2017, doi: 10.1145/3083187.3083212.

[25] S. Shrestha, B. Khanal, and S. Ali, "Ensemble U-Net Model for Efficient Polyp Segmentation," 2020, doi: 10.1136/gutjnl-2015-310912.

[26] S. Ali and N. K. Tomar, "Iterative deep learning for improved segmentation of endoscopic images," *Nordic Machine Intelligence*, vol. 1, no. 1, pp. 38–40, Nov. 2021, doi: 10.5617/NMI.9137.

[27] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation", Accessed: Mar. 30, 2022. [Online]. Available: https://github.com/tensorflow/models/tree/master/

[28] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs," pp. 1–12, Dec. 2014, doi: 10.48550/arxiv.1412.7062.

[29] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," Jun. 2017, doi: 10.48550/arxiv.1706.05587.

[30] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 6230–6239, Dec. 2016, doi: 10.48550/arxiv.1612.01105.

[31] W. Liu, A. Rabinovich, and A. C. Berg, "ParseNet: Looking Wider to See Better," Jun. 2015, doi: 10.48550/arxiv.1506.04579.

[32] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *32nd International Conference on Machine*

Learning, ICML 2015, vol. 1, pp. 448–456, Feb. 2015, doi: 10.48550/arxiv.1502.03167.

[33] Z. M. Ramadan, "Using Entropy and 2-D Correlation Coefficient as Measuring Indices for Impulsive Noise Reduction Techniques," *International Journal of Applied Engineering Research*, vol. 12, pp. 11101–11106, 2017, Accessed: Jul. 09, 2022. [Online]. Available: http://www.ripublication.com

[34] J. H. Nam and S. C. Lee, "FREQUENCY FILTERING FOR DATA AUGMENTATION IN X-RAY IMAGE CLASSIFICATION," *Proceedings - International Conference on Image Processing, ICIP*, vol. 2021-September, pp. 81–85, 2021, doi: 10.1109/ICIP42928.2021.9506587.

[35] L. Nanni, M. Paci, S. Brahnam, and A. Lumini, "Comparison of Different Image Data Augmentation Approaches," *Journal of Imaging 2021, Vol. 7, Page 254*, vol. 7, no. 12, p. 254, Nov. 2021, doi: 10.3390/JIMAGING7120254.

[36] A. M. Khan, N. Rajpoot, D. Treanor, and D. Magee, "A nonlinear mapping approach to stain normalization in digital histopathology images using image-specific color deconvolution," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 6, pp. 1729–1738, 2014, doi: 10.1109/TBME.2014.2303294.

[37] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 34–41, Sep. 2001, doi: 10.1109/38.946629.

[38] M. Macenko *et al.*, "A method for normalizing histology slides for quantitative analysis," *Proceedings - 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, ISBI 2009*, pp. 1107–1110, 2009, doi: 10.1109/ISBI.2009.5193250.

[39] D. Jha *et al.*, "Real-Time Polyp Detection, Localization and Segmentation in Colonoscopy Using Deep Learning," *Ieee Access*, vol. 9, p. 40496, 2021, doi: 10.1109/ACCESS.2021.3063716.

[40] N. Kumar Tomar *et al.*, "DDANet: Dual Decoder Attention Network for Automatic Polyp Segmentation".

[41] C.-H. Huang, H.-Y. Wu, and Y.-L. Lin, “HarDNet-MSEG: A Simple Encoder-Decoder Polyp Segmentation Neural Network that Achieves over 0.9 Mean Dice and 86 FPS,” Jan. 2021, doi: 10.48550/arxiv.2101.07172.