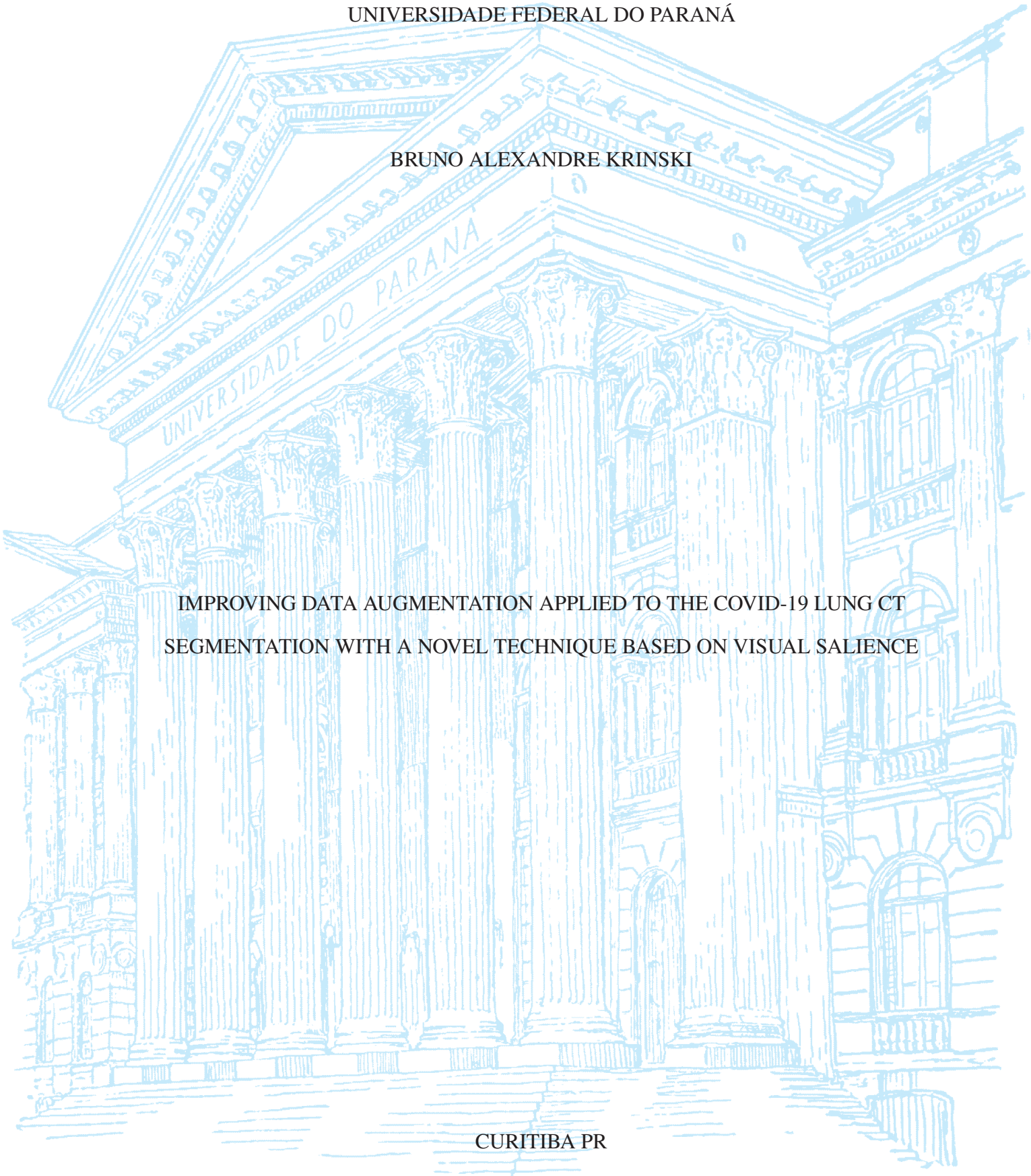
UNIVERSIDADE FEDERAL DO PARANÁ

BRUNO ALEXANDRE KRINSKI



IMPROVING DATA AUGMENTATION APPLIED TO THE COVID-19 LUNG CT SEGMENTATION WITH A NOVEL TECHNIQUE BASED ON VISUAL SALIENCE

CURITIBA PR

2024

BRUNO ALEXANDRE KRINSKI

IMPROVING DATA AUGMENTATION APPLIED TO THE COVID-19 LUNG CT SEGMENTATION WITH A NOVEL TECHNIQUE BASED ON VISUAL SALIENCE

Tese apresentada como requisito parcial à obtenção do grau de Doutor em Ciência da Computação no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Eduardo Todt.

CURITIBA PR

2024

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA DE CIÊNCIA E TECNOLOGIA

Krinski, Bruno Alexandre
Improving data augmentation applied to the Covid-19 lung CT segmentation with a novel technique based on visual salience / Bruno Alexandre Krinski. – Curitiba, 2024.
1 recurso on-line : PDF.

Tese (Doutorado) - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática.

Orientador: Eduardo Todt

1. COVID-19 (Doença). 2. Tomografia. 3. Pulmões. 4. Segmentação semântica. I. Universidade Federal do Paraná. II. Programa de Pós-Graduação em Informática. III. Todt, Eduardo. IV. Título.

Bibliotecário: Elias Barbosa da Silva CRB-9/1894

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da tese de Doutorado de **BRUNO ALEXANDRE KRINSKI** intitulada: **Improving Data Augmentation applied to the COVID-19 lung CT Segmentation with a novel technique based on Visual Salience**, sob orientação do Prof. Dr. EDUARDO TODT, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de doutor está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 21 de Outubro de 2024.

Assinatura Eletrônica
22/10/2024 13:34:07.0
EDUARDO TODT
Presidente da Banca Examinadora

Assinatura Eletrônica
22/10/2024 12:02:24.0
RENATO TINÓS
Avaliador Externo (UNIVERSIDADE DE SÃO PAULO)

Assinatura Eletrônica
24/10/2024 07:48:09.0
SILVIA SILVA DA COSTA BOTELHO
Avaliador Externo (UNIVERSIDADE FEDERAL DO RIO GRANDE)

Assinatura Eletrônica
22/10/2024 12:26:29.0
EDUARDO JAQUES SPINOSA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

For my parents, Daisy M. Krinski and Simão D. Krinski.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank God for being my strength and guide in the writing of this dissertation. Without Him, I would not have had the wisdom or the physical ability to do so. I am extremely grateful to my parents for their love, prayers, caring, sacrifices, and always supporting my education. I express my deep and sincere gratitude to my advisor Eduardo Todt, for allowing me to write this thesis and providing invaluable guidance throughout this research. It was a great privilege and honor to work and study under his guidance. I would also like to thank all friends and colleagues of the Vision, Robotics, and Image (VRI) research group, that helped me to grow as a person and researcher. I gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research, as well as the C3SL-UFPR group for the computational cluster infrastructure. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Thanks to Brazilian Coordination for the Improvement of Higher Education Personnel (CAPES) for the financial support for this research.

RESUMO

Devido à pandemia global de COVID-19, o diagnóstico assistido por computador de imagens médicas ganhou significativa atenção. Fornecer uma segmentação semântica robusta de tomografias computadorizadas (CT) é altamente desejável, pois permite um diagnóstico rápido e reduz a carga de tempo sobre os especialistas. Muitos estudos empregaram técnicas de aprendizado profundo e redes neurais profundas, alcançando resultados impressionantes na segmentação eficaz de tomografias de COVID-19. As redes neurais profundas são amplamente utilizadas em várias tarefas de segmentação devido à sua capacidade de generalizar e aprender a representação de diferentes classes de objetos dentro das imagens. No entanto, esses métodos requerem uma quantidade substancial de dados para o treinamento, e o problema de segmentação de tomografias de COVID-19 carece de dados disponíveis na literatura. Este estudo propõe uma nova técnica de aumento de dados baseada em características de saliência visual para enfrentar esse desafio e pode ser dividido em três fases. A primeira fase deste estudo envolveu uma avaliação extensiva de cento e vinte redes de segmentação, compostas por vinte codificadores combinados com seis decodificadores, em cinco conjuntos de dados. A principal conclusão dessa etapa destacou a necessidade crítica de técnicas eficazes de aumento de dados. Na segunda fase, foram avaliadas vinte técnicas de aumento de dados, cada uma testada com dez probabilidades de aplicação. As técnicas de aumento foram avaliadas de duas maneiras. Primeiro, foram aplicadas separadamente aos conjuntos de treinamento de cada conjunto de dados. Em seguida, os conjuntos de treinamento de todos os conjuntos de dados foram combinados em um conjunto unificado, com as técnicas de aumento aplicadas a esse conjunto unificado. Os resultados indicaram que as técnicas de aumento de dados tiveram melhor desempenho no conjunto unificado. Além disso, as técnicas de transformação espacial alcançaram melhores resultados no geral. No entanto, apesar das melhorias trazidas pelo aumento de dados, as técnicas genéricas não produziram consistentemente melhores resultados em todos os conjuntos de dados, destacando a necessidade de uma abordagem específica de aumento de dados para o problema. A terceira fase deste trabalho envolveu o desenvolvimento e a avaliação de uma nova técnica de aumento de dados. O fluxo de trabalho do aumento proposto utiliza um modelo GAN para gerar tomografias CT saudáveis de regiões pulmonares, combinando-as com lesões de COVID-19 já rotuladas para criar amostras adicionais de treinamento, aprimorando o processo de treinamento de segmentação. Esta fase avaliou duas versões do fluxo de trabalho proposto: uma sem o algoritmo de saliência, onde as lesões foram adicionadas aleatoriamente às tomografias saudáveis, e outra que utiliza a distância de saliência visual para adicionar as lesões. A técnica de aumento proposta com o uso de saliência visual alcançou os melhores resultados no conjunto de dados Ricord1a em comparação com as técnicas genéricas de aumento e a versão aleatória da proposta. Por fim, combinada com técnicas clássicas de aumento, a técnica proposta apresentou os melhores resultados em quatro conjuntos de dados.

Palavras-chave: COVID-19, Segmentação Semantica, Aumento de Dados

ABSTRACT

Due to the COVID-19 global pandemic, computer-assisted diagnosis of medical images has gained significant attention. Providing robust semantic segmentation of Computed Tomography (CT) scans is highly desirable because it allows rapid diagnosis and reduces the time burden on specialists. Many studies have employed deep learning techniques and deep neural networks and achieved impressive results in effectively segmenting COVID-19 CT scans. Deep neural networks are widely used in various segmentation tasks due to their generalization ability to learn the representation of different classes of objects within images. However, these methods require substantial data for training, with the COVID-19 CT problem needing more available data in the literature. This study proposes a novel data augmentation technique based on visual salience features to address this challenge and can be divided into three phases. The first phase of this study involved an extensive evaluation of one hundred and twenty segmentation networks, comprising twenty encoders combined with six decoders across five datasets. The main conclusion from this step highlighted the critical need for effective data augmentation techniques. Twenty data augmentation techniques were evaluated in the second phase, each tested with ten application probabilities. The augmentation techniques were evaluated in two ways. First, they were applied separately to the training sets of each dataset. Second, the training sets of all datasets were combined into a unified training set, with the augmentation techniques applied to this unified set. The results indicated that augmentation techniques performed better on the unified training set. Additionally, spatial transform techniques achieved higher results overall. However, despite the improvements brought by data augmentation, generic techniques did not consistently yield better results across all datasets, underscoring the need for a problem-specific data augmentation approach. The third phase of this work involved developing and evaluating a novel data augmentation technique. The proposed augmentation workflow utilizes a Generative Adversarial Network (GAN) model to generate healthy CT scans of lung regions, combining them with existing labeled COVID-19 lesions to create additional training samples, enhancing the segmentation training process. This phase evaluated two versions of the proposed workflow: one without the saliency algorithm, where lesions were randomly added to the healthy CT scans, and one that uses visual salience distance to add lesions. The proposed augmentation technique with the visual salience achieved the highest results on the Ricord1a dataset compared to generic augmentation techniques and the random version of the proposed augmentation. Finally, combined with classic augmentation techniques, the proposed augmentation yielded the best results in four datasets.

Keywords: COVID-19, Semantic Segmentation, Data Augmentation

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

FCN	Fully Convolutional Network
MTL	Multi-task Learning
GAN	Generative Adversarial Network
GAM	Gate Attention Module
DAM	Decoder Attention Module
RAB	Residual Attention Block
RA	Reverse Attention
IDC	Improved Dilation Convolution
AG	Attention Gate
TA-SegNet	Tri-level Attention-based Segmentation Network
TAU	Tri-level Attention Unite
QAP-Net	Quadruple Augmented Pyramid Network
RAIU-Net	Residual Attention Inception U-Net
CHS-Net	Covid-19 Hierarchical Segmentation Network
CSSE	Systems Science and Engineering
JHU	Johns Hopkins University
RT-PCR	Reverse-Transcription Polymerase Chain Reaction
CT	Computed Tomography
ResNet	Residual Network
CNN	Convolutional Neural Network
ReLu	Rectified Linear Units
FLOPS	Floating Point Operations per Second
SE-Net	Squeeze-and-Excitation Networks
FPN	Feature Pyramid Network
PSPNet	Pyramid Scene Parsing Network
MA-Net	Multi-scale Attention Net
GGO	Ground Glass Opacity
IoU	Intersection over Union
PAB	Position-wise Attention Block
MFAB	Multi-scale Fusion Attention Block
VGG	Visual Geometry Group Network
IAGAN	Inception-Augmentation GAN
RANDGAN	Randomized Generative Adversarial Network
WSS	Weakly-Supervised Segmentation
RNN	Recurrent Neural Network

GAN	Generative Adversarial Network
FC	Fully Connected
AdaIN	Adaptive Instance Normalization
FID	Fréchet inception distance
CLAHE	Contrast Limited Adaptive Histogram Equalization
ImageNet	ImageNet Large Scale Visual Recognition Challenge
RPU	Residual Pooling Unit
RPN	Region Proposal Network
FUNIT	Few-shot UNsupervised Image-to-image Translation
ROI	Region of Interest
ESM	Edge Supervised Module
ASSM	Auxiliary Semantic Supervised Module
AFM	Attention Fusion Module
EAMC	Ensembled Attention-based Multi-scaled Convolution network
MS	Attention-modulated Multi-Scalar
DESUM	Dynamic Element-wise Sum
DFM	Dynamic Feature Matching
MSFCB	Multiscale Feature Capture Block
MSFF	Multiscale Feature Fusion
RBC	Random Brightness Contrast
LBP	Local binary patterns
KNN	K-Nearest Neighbors
COPD	Chronic Obstructive Pulmonary Disease

CONTENTS

1 INTRODUCTION

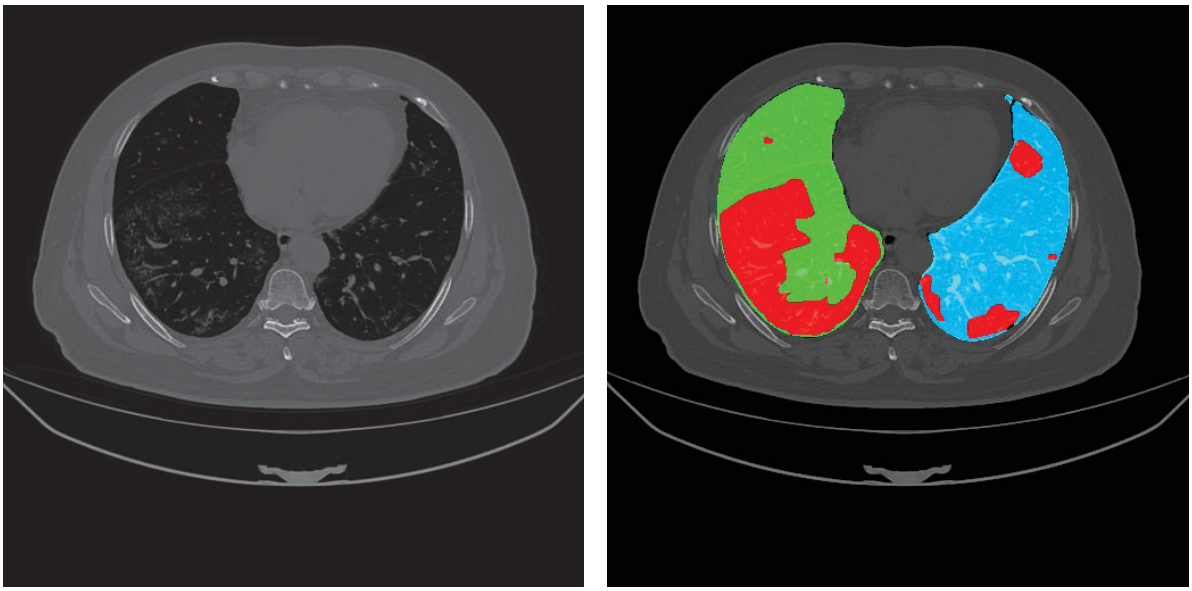
The COVID-19 pandemic has been a global struggle since 2019, with millions of infections and fatalities worldwide (Wang et al., 2020). The latest figures from (Johns Hopkins University of Medicine, 2024), updated on October 10th, 2023, show that the number of COVID-19 cases has surpassed 676 million globally, with over 6.8 million deaths. According to (Chen et al., 2020a) and (Huang et al., 2020), early detection is a highly effective measure in the battle against the virus, with the Reverse-Transcription Polymerase Chain Reaction (RT-PCR) being widely applied in the diagnostic of suspected subjects (Fan et al., 2020a). Unfortunately, many countries need more equipment for performing RT-PCR, which impedes the quick identification of virus presence (Fan et al., 2020a). Moreover, as stated by (Ai et al., 2020), RT-PCR has high false-negative rates.

Given the limited availability of equipment (Fan et al., 2020a) and high false-negative rates associated with RT-PCR (Ai et al., 2020), there is a growing need for alternative methods for detecting COVID-19 (Narin et al., 2021). Automatic detection of COVID-19 through Computed Tomography (CT) scans has emerged as a promising approach, with recent studies, such as (Shi et al., 2021) and (Yazdekhasty et al., 2021), demonstrating its effectiveness in diagnosing and identifying COVID-19 patients. The Semantic Segmentation of CTs is one of several research areas explored for the automatic detection of COVID-19 and has received significant attention since the outbreak (Bizopoulos et al., 2020; Shi et al., 2021; Yazdekhasty et al., 2021). Thus, Deep Learning methods for COVID-19 CT segmentation have become an essential supplementary tool for RT-PCR tests (Zhang et al., 2022).

Even as the COVID-19 pandemic subsides and the number of cases declines, the need for lung semantic segmentation remains crucial and the applications of lung semantic segmentation extend far beyond COVID-19. Performing lung segmentation is indispensable for the detection, diagnosis, and management of a wide range of pulmonary conditions, including Chronic Obstructive Pulmonary Disease (COPD) (Winther et al., 2019; Dogan et al., 2024), lung cancer (Primakov et al., 2022; Riaz et al., 2023), pneumonia (Wang et al., 2023; Zarate et al., 2023), and interstitial lung disease (Gupta and Singh Bhadauria, 2022; Cai et al., 2023).

The Semantic Segmentation (Long et al., 2015), is an image processing operation that performs per-pixel classification of an image. Each image pixel is labeled according to the object class it belongs (Garcia-Garcia et al., 2018; Minaee et al., 2021). In the context of COVID-19 CT segmentation, the objective is to identify each pixel corresponding to a COVID-19 lesion within a patient's lung (Bizopoulos et al., 2020; Shi et al., 2021). The Zenodo dataset (Jun et al., 2020; Ma et al., 2021) provides a useful example of CT segmentation, as illustrated in Figure 1.1. Sub-figure 1.1(a) shows an example CT, while Sub-figure 1.1(b) shows the corresponding segmentation mask. The Zenodo dataset includes labels for the left and right lungs and COVID-19 lesions (Jun et al., 2020; Ma et al., 2021; Saood and Hatem, 2021).

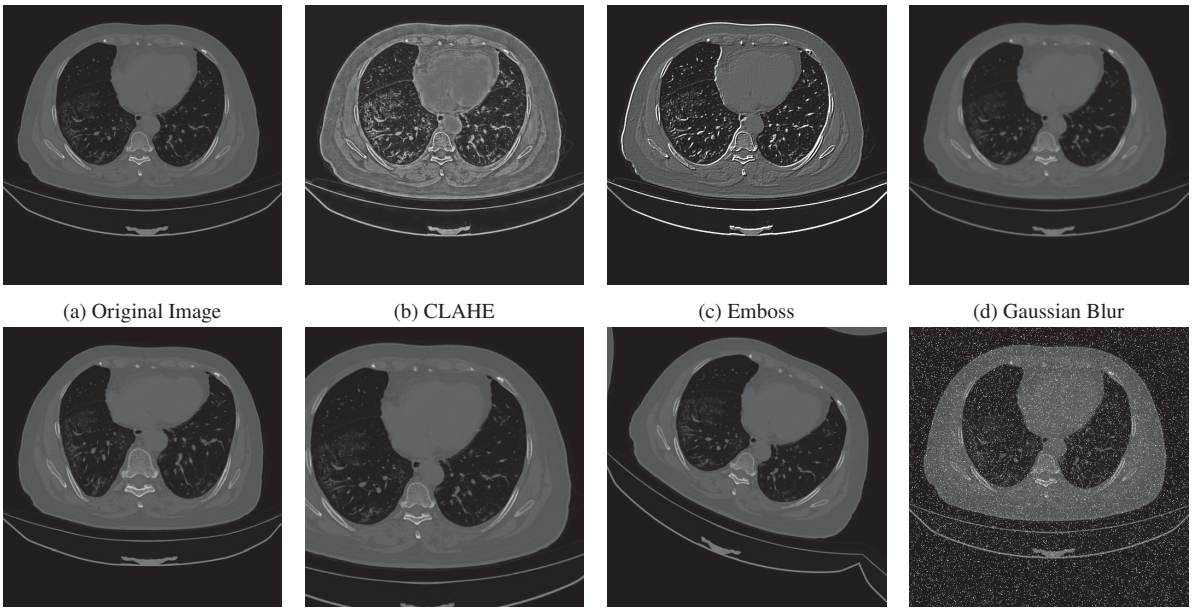
To perform COVID-19 CTs segmentation, many studies in the field have employed Deep Learning techniques and Deep Neural Networks, which have achieved impressive results (Shi et al., 2021; Saood and Hatem, 2021; Alshomrani et al., 2023). Deep Neural Networks are commonly used in segmentation problems due to their excellent generalization capacity and ability to learn to represent different object classes in an image (Garcia-Garcia et al., 2018; Minaee et al., 2021). However, the drawback of Deep Neural Network-based methods is their high demand for data to train the networks (Shorten and Khoshgoftaar, 2019; Ankile et al., 2020). Some datasets, such as ImageNet Large Scale Visual Recognition Challenge (ImageNet) (Deng



(a) An example of CT from the Zenodo dataset.

(b) The corresponding segmentation mask.

Figure 1.1: Example of CT from the Zenodo dataset (Jun et al., 2020; Ma et al., 2021). The Zenodo dataset includes labels for the left and right lungs and COVID-19 lesions. In the segmentation mask, the left lung is represented in green, the right lung in blue, and the COVID-19 lesions in red. The uncolored regions of the image represent the background and are irrelevant to the problem. Source: (Jun et al., 2020; Ma et al., 2021).



(a) Original Image

(b) CLAHE

(c) Emboss

(d) Gaussian Blur

(e) Grid Distortion

(f) Random Sized Crop

(g) Rotate

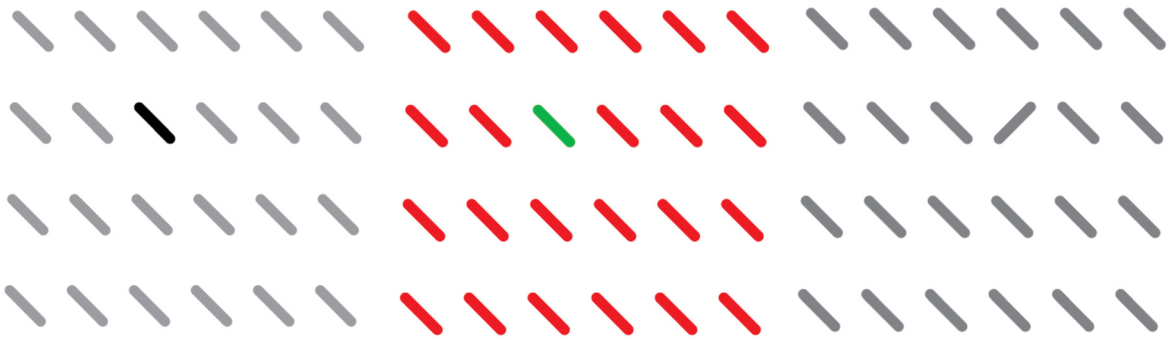
(h) Salt and Pepper

Figure 1.2: Examples of data augmentation techniques that can be applied to a CT with COVID-19 lesions to generate synthetic images. Each operation produces a new image. Sub-figure 1.2(a) shows the original image, and Sub-figures 1.2(b) to 1.2(h) provide examples of the transformations applied to the original image. Source: (Jun et al., 2020; Ma et al., 2021).

et al., 2009) and COCO (Lin et al., 2014), contain thousands of images. The use of Deep Neural Networks for Semantic Segmentation of COVID-19 CTs is also limited by two factors. The first is that labeling Semantic Segmentation is a time-consuming and labor-intensive process, as each pixel of the image must be accurately labeled to avoid the network converging to incorrect results (Cao and Bao, 2020; Cao et al., 2020). Furthermore, highly specialized doctors must label

CT segmentation datasets to identify lesion regions in the image accurately (Fan et al., 2020a; Shi et al., 2021).

To address these limitations, two approaches commonly used in the literature are transfer learning and data augmentation. Transfer learning involves using Deep Neural Networks already trained on generic problems and large datasets, such as ImageNet, to perform specialized training of the last layers of a Deep Neural Network for the target problem (Zhuang et al., 2021). On the other hand, data augmentation aims to increase the number of images in the dataset by applying various operations to the image to generate synthetic data (Shorten and Khoshgoftaar, 2019). Figure 1.2 illustrates data augmentation techniques that can be applied to a CT with COVID-19 lesions to generate synthetic images. Each operation produces a new image. Sub-figure 1.2(a) shows the original image, and Sub-figures 1.2(b) to 1.2(h) provide examples of the transformations applied to the original image.



(a) Example of Visual Saliency created by the difference in intensity (the black bar stands out from the gray bars).

(b) Example of Visual Salience by the difference in color (the green bar stands out from the red bars).

(c) Example of Visual Salience by the difference in orientation (the bar inclined in the opposite direction stands out from the others).

Figure 1.3: Examples of Visual Salience features. Source: (Todt, 2005).

Although data augmentation techniques are widely used in Deep Learning (Shorten and Khoshgoftaar, 2019), COVID-19 CT segmentation research has yet to explore these algorithms thoroughly. Typically, generic techniques commonly used in Deep Learning are relied upon, with only a few studies detailing the specific data augmentation methods applied and their impact on network training for COVID-19 CT segmentation. Therefore, this work aims to comprehensively study data augmentation techniques specifically for the COVID-19 CT segmentation problem. As a product of this research, a novel data augmentation technique is developed as the main contribution of this work. The proposed technique uses Visual Silences to generate new CTs with COVID-19 lesions. Visual Salience refers to a subjective perceptual feature in objects that makes them noticeable from their surrounding regions and immediately captures the human brain's attention (Itti, 2007; Krinski et al., 2019). Figure 1.3 presents three features that make an object or region stand out from its surrounding elements. Sub-figure 1.3(a) illustrates an example of Visual Saliency created by the difference in intensity (the black bar stands out from the gray bars), and Sub-figure 1.3(b) by the difference in color (the green bar stands out from the red bars). Sub-figure 1.3(c) by the difference in orientation (the bar inclined in the opposite direction stands out from the others).

The concept of Visual Salience can also be applied to CTs. Figure 1.4 illustrates this with examples of three CTs of COVID-19 patients from the Zenodo dataset (Jun et al., 2020; Ma et al., 2021). Sub-figures 1.4(a), 1.4(b), and 1.4(c) show the CTs, and their corresponding segmentation masks are presented in Sub-figures 1.4(d), 1.4(e), and 1.4(f). The images and their corresponding ground-truths in Figure 1.4 show that the lesions in Sub-figure 1.4(a) have a

higher contrast when compated with the background and are more visually salient than those in Sub-figure 1.4(b), which are more visually salient than those in Sub-figure 1.4(c). This difference in salience levels is due to the contrast, proportion and position of the lesions within the image. The variation in salience levels can significantly impact the training process of a segmentation network (Liu et al., 2023; Buongiorno et al., 2023). Segmentation models based on Deep Learning may find examples like Sub-figure 1.4(a) and Sub-figure 1.4(b) easier to segment than examples like Sub-figure 1.4(c). The idea behind a data augmentation based on Visual Saliences is to consider these differences when generating new examples, calibrating the new data to balance the dataset with optimized examples in terms of Visual Salience.

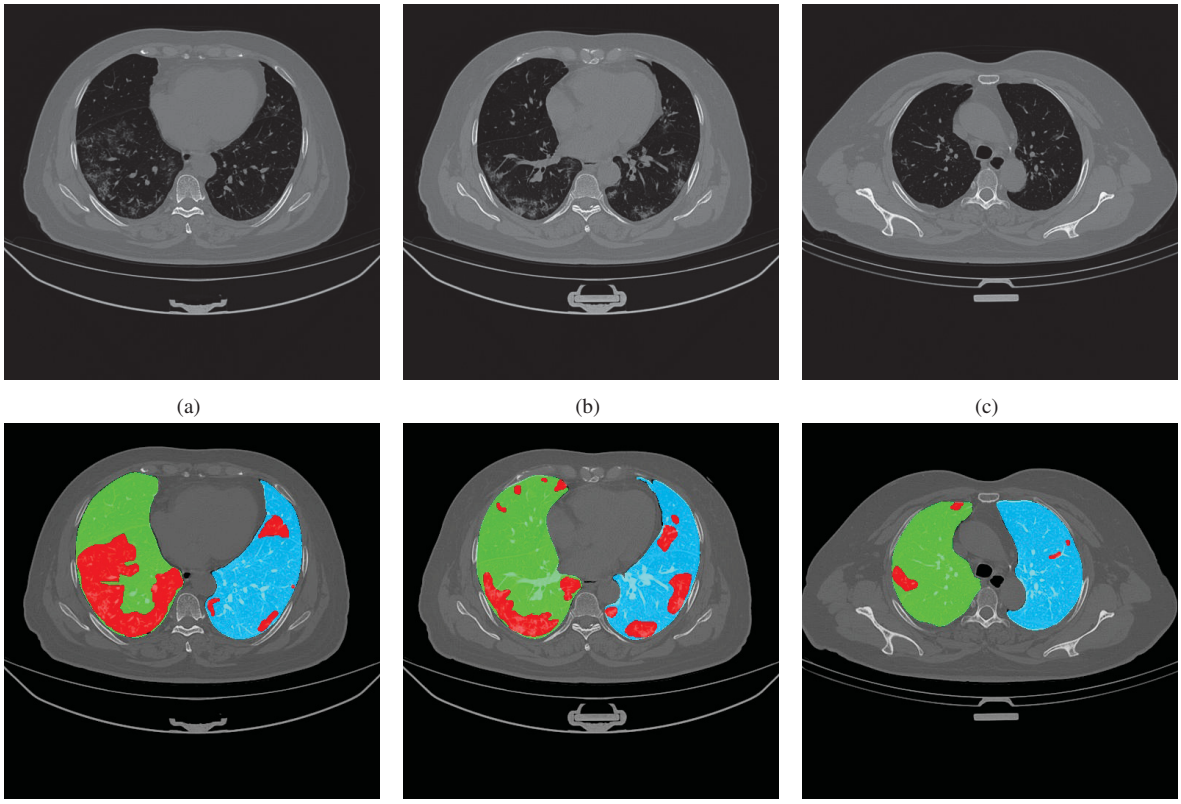


Figure 1.4: Examples of three CTs of COVID-19 patients from the Zenodo dataset (Jun et al., 2020; Ma et al., 2021). Sub-figures 1.4(a), 1.4(b), and 1.4(c) show the CTs, and their corresponding segmentation masks are presented in Sub-figures 1.4(d), 1.4(e), and 1.4(f). The color green represents the Left Lung, blue represents the Right Lung, and red represents the COVID-19 lesion. Source: (Jun et al., 2020; Ma et al., 2021).

Aiming to study the data augmentation techniques on the COVID-19 CT segmentation problem and develop a novel data augmentation technique for this problem, this work is divided into three main phases. The first phase presents an extensive benchmark of encoder-decoder networks on the COVID-19 CT segmentation problem. Encoder-Decoder Networks are Deep Neural Networks largely used for segmentation problems, consisting of an encoder to extract features and a decoder to perform segmentation (Badrinarayanan et al., 2017). In this analysis, twenty encoders were combined with six decoders, resulting in one hundred and twenty encoder-decoder architectures evaluated. The models were trained and evaluated on five different CT datasets: MedSeg (MedSeg, 2021), Zenodo (Jun et al., 2020; Ma et al., 2021), CC-CCII (Zhang et al., 2020), MosMed (Morozov et al., 2020), and Ricord1a (Tsai et al., 2020). Each dataset was validated using a five-fold cross-validation strategy, resulting in 3,000 experiments. The analysis conducted in the first phase uncovered certain weaknesses that needed to be addressed

in the subsequent phases. One of the primary challenges identified was the limited number of images in the datasets and the significant class imbalance, which adversely affected the deep neural networks' learning process.

The second phase of this study involved evaluating twenty conventional data augmentation techniques to address the aforementioned data-related issues. The experiments were performed on the same five datasets used in the first phase. A unified training approach was also examined, where the training sets of the five datasets were merged and used for network training. The data augmentation methods were evaluated on this unified training set with the same ten probabilities. A 5-fold cross-validation strategy was employed to validate all experiments, resulting in over 5,000 experiments for the first data augmentation evaluation and over 1,000 experiments for the unified training augmentation evaluation. The results showed that applying data augmentation methods with the unified training set produced superior outcomes compared to with applying data augmentation on each dataset individually. Furthermore, spatial-level transformations proved to be the most promising in improving the performance of neural networks for the COVID-19 CT segmentation challenge.

In the third phase of this work, a new data augmentation was proposed and evaluated for the specific domain of the COVID-19 segmentation problem. This new data augmentation uses Generative Adversarial Networks (GANs), Stargan (Choi et al., 2020) and Stylegan (Karras et al., 2020), to generate healthy CT images without any COVID-19 lesion. A segmentation model is trained to generate segmentation masks with the lung regions of these new images. The lung masks generated by the segmentation model guides the augmentation algorithm to place the new lesions in the image. In a first implementation of the proposed augmentation, called DACov, a matching algorithm randomly combines the new images generated by GANs with an image with COVID-19 lesions from the dataset. The experiments performed in this random version of the proposed augmentation followed the same steps as the experiments performed in the previews unified training approach evaluation, and the results of the proposed augmentation are compared with the twenty conventional data augmentation techniques. However, because of its multistep image generation process, it was applied in an offline augmentation strategy. A 5-fold cross-validation strategy was used to validate all experiments, resulting in over 200 experiments. The proposed augmentation achieved competitive results with generic data augmentation techniques and demonstrated promising results on the Ricord1a dataset.

A salience based version of the proposed augmentation, called SalDACov, was implemented and evaluated. In this new version, a saliency algorithm chooses the lesion region to be added to the healthy image generated by the GAN. In addition, each lung (right and left) has its own list of compatible lesion based on lung size. So, the healthy lung image generated by the GAN receives lesions from different training samples instead of the same image, and the lesion can be in the same side or opposite side from original image. In total, nine variations of the proposed augmentation is evaluated, considering the salience distance: maximum, minimum, and random. And considering the side of the lesion in the original dataset image: same side, opposite side, and random. A 5-fold cross-validation strategy was used to validate all experiments, resulting in more than 900 experiments. The salience-based version of the proposed augmentation achieved the highest F-scores in the Ricord1a dataset.

Finally, the salience based version of the proposed augmentation was also combined with the twenty generic augmentation techniques previously evaluated. A 5-fold cross-validation strategy was employed to validate all experiments, resulting in over 600 experiments. The combination of the proposed augmentation with the generic augmentation techniques yielded the highest results on three of five datasets: MosMed, Ricord1a, and Zenodo.

1.1 OBJECTIVES AND HYPOTHESIS

The three main objectives of this work are: (i) an extensive comparison between segmentation models for COVID-19 CT segmentation problem, pointing the best models to be applied in this problem; (ii) an extensive comparison between data augmentation techniques for COVID-19 CT segmentation problem, pointing the best data augmentation techniques to be applied in this problem; and (iii) a novel data augmentation based on Visual Salience for COVID-19 CT segmentation problem aiming to outperform traditional data augmentation techniques.

From the main objectives of this work, specific objectives are raised with the following hypotheses, denoted by H_n:

- An extensive comparison between encoder-decoder models in the COVID-19 segmentation problem. H_1: The combination of several encoder and decoder models used as segmentation models perform promising results on the COVID-19 CT segmentation problem.
- An extensive comparison between generic data augmentation techniques in the COVID-19 segmentation problem. H_2: Data augmentation mitigate the problem of small datasets available in the literature of COVID-19 CT segmentation. Thus, improving the segmentation results by generating syntactic data.
- An extensive comparison between generic data augmentation techniques with the training sets of all datasets unified in one big training set. H_3: Training a segmentation model in a unified training set of all datasets generates promising results with data augmentation techniques compared to training the segmentation model with each dataset individually.
- Training GANs for healthy CT generation. H_4: GANs generate consistent CT images without COVID-19 lesions.
- Training a segmentation model for lung segmentation of healthy CTs generated by GANs. H_5: A segmentation model trained for segmentation of lung regions on the available datasets of COVID-19 CT segmentation problem generates promising segmentation mask of the lung regions of the synthetic images generated by GANs.
- Add COVID-19 lesions inside the lung regions of the healthy CTs generated by GANs. H_6: The proposed data augmentation improve the segmentation results by increasing the number of CTs in the training sets.
- Apply Visual Salience concept to improve the proposed data augmentation. H_7:The use of Visual Salience concepts in a data augmentation generate more robust CTs and improve the segmentation results.
- Combination of the proposed augmentation with traditional augmentation techniques to improve the COVID-19 CT segmentation. H_8: The proposed augmentation based on Visual Salience when combined with traditional augmentation techniques generate more promising outcomes to just use traditional augmentation techniques.

1.2 CONTRIBUTIONS AND ROADMAP

Based on the objectives and hypotheses established, this work followed this roadmap: It started with an evaluation of 120 semantic segmentation models for the COVID-19 CT segmentation

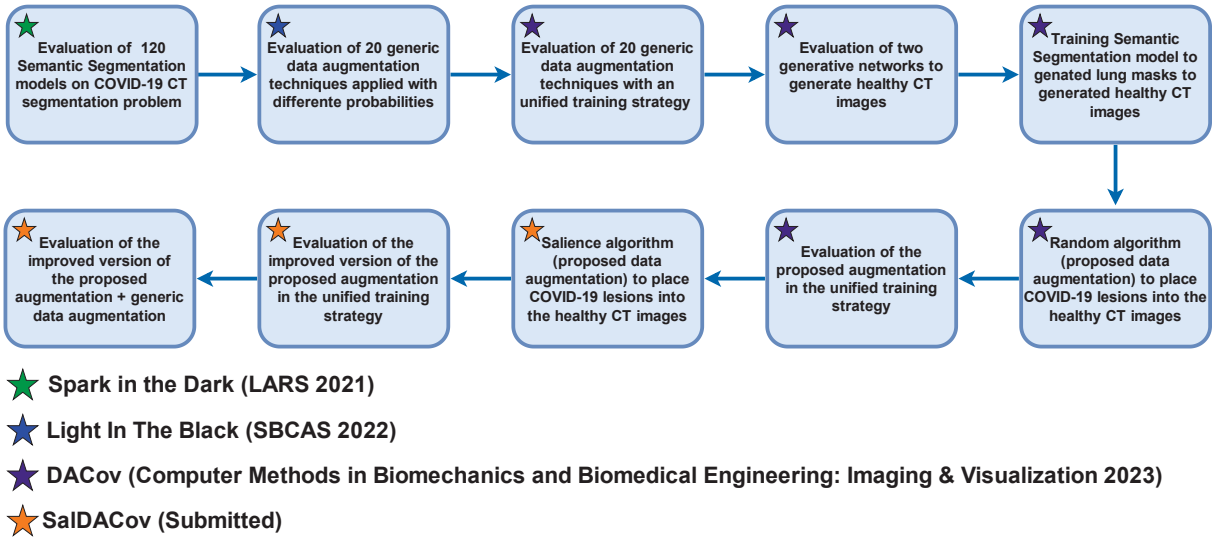


Figure 1.5: This work started with an evaluation of encoder-decoder networks for the COVID-19 CT segmentation problem. Then, an evaluation of generic data augmentation techniques with network training performed on each dataset individually and then on the combined training sets of all datasets. A novel data augmentation technique for the COVID-19 CT segmentation problem was also evaluated.

problem, followed by an assessment of 20 generic data augmentation techniques with network training performed on each dataset individually and then on the combined training sets of all datasets. A novel data augmentation technique for the COVID-19 CT segmentation problem was also evaluated, involving the assessment of two GANs models for healthy image generation, training a semantic segmentation model for lung segmentation of the GAN-generated images, and evaluating two algorithms for adding COVID-19 lesions to the healthy CT images—one generic and the other based on visual saliency. Finally, the proposed augmentation technique was evaluated in combination with the generic data augmentation techniques. Figure 1.5 presents a roadmap outlining this work's main phases and contributions, and the publications of the results from this study are listed below.

- An extensive analysis of encoder-decoder models evaluated on the COVID-19 CT segmentation problem. From this evaluation, the following paper was published: Krinski, B. A., Ruiz, D. V. e Todt, E. (2021). Spark in the Dark: Evaluating Encoder-Decoder Pairs for COVID-19 CT's Semantic Segmentation. In 2021 Latin American Robotics Symposium (LARS), 2021 Brazilian Symposium on Robotics (SBR), and 2021 Workshop on Robotics in Education (WRE). IEEE.

- An extensive analysis of generic data augmentation techniques evaluated on the COVID-19 CT segmentation problem. From this evaluation, the following paper was published: Krinski, B. A., Ruiz, D. V. e Todt, E. (2022). Light In The Black: An Evaluation of Data Augmentation Techniques for COVID-19 CT's Semantic Segmentation. In Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS 2022).

- Evaluation of a training strategy with the training sets unified in one big training set published on: Krinski, B. A., Ruiz, D. V., Laroca, R., and Todt, E. (2023). DACov: a deeper analysis of data augmentation on the computed tomography segmentation problem.Computer Methods inBiomechanics and Biomedical Engineering: Imaging & Visualization, pages 1–18.

- Evaluation of a novel data augmentation technique (random version) for COVID-19 CT segmentation problem. The proposed augmentation was divided into the evaluation of two GANs trained to generate healthy CT images without COVID-19 lesions, training a segmentation model trained to perform segmentation of lung regions in a CT image, and a random algorithm to add COVID-19 lesions of healthy CT images. The results were also published on: Krinski, B. A., Ruiz, D. V., Laroca, R., and Todt, E. (2023). DACov: a deeper analysis of data augmentation on the computed tomography segmentation problem.Computer Methods inBiomechanics and Biomedical Engineering: Imaging & Visualization, pages 1–18.

- Evaluation of a novel data augmentation technique (salience version) for COVID-19 CT segmentation problem. The proposed augmentation was divided into the evaluation of two GANs trained to generate healthy CT images without COVID-19 lesions, training a segmentation model trained to perform segmentation of lung regions in a CT image, and a salience algorithm to add COVID-19 lesions of healthy CT images.

- Evaluation of the novel data augmentation technique (salience version) in combination with traditional augmentation techniques.

The code used in the experiments performed in this work and the link for the repositories of each publication individually are available on the following Github: `https://github.com/BrunoKrinski`.

1.3 OUTLINE

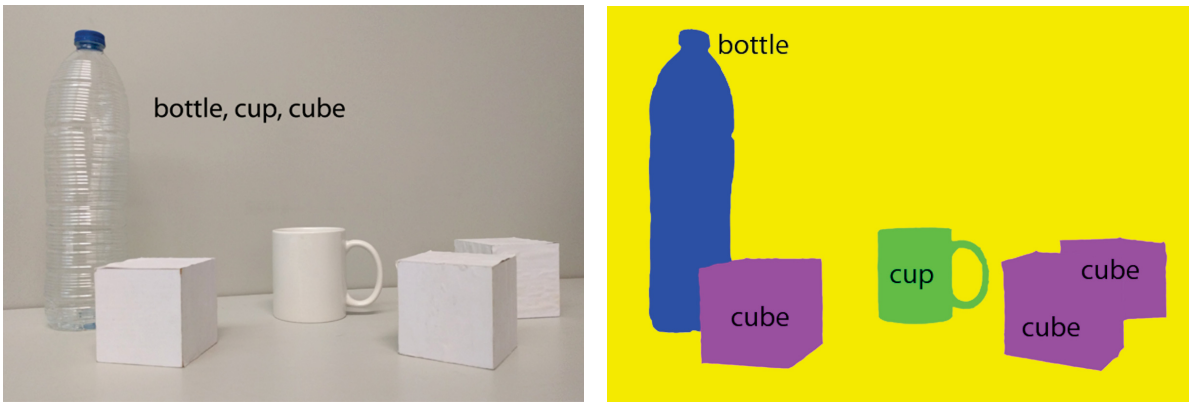
This work is divided as follows: Chapter 2 presents provides a general background of Semantic Segmentation, Deep Learning, GANs, and the architectures used in this work. Chapter 3 presents a literature review of segmentation studies proposed for the COVID-19 CT segmentation problem and data augmentation techniques. Chapter 4 details the proposed data augmentation. Chapter 5 presents the datasets and evaluation metrics. Chapter 6 presents the experiments performed to evaluate the segmentation networks, the data augmentations, and the proposed augmentation technique. This Chapter also presents the discussion about the experiments. Finally, Chapter 7 ends this study by providing the conclusions and future works.

2 MAIN CONCEPTS

This chapter introduces the main concepts used to develop this work. Section 2.1 covers the Semantic Segmentation problem, while Section 2.2 presents the concept of Computed Tomography (CT) segmentation. Then, Section 2.3 overviews the Deep Learning research field and Convolutional Neural Networks (CNNs), and Section 2.4 presents the encoder-decoder architecture of deep neural networks used for segmentation problems. Furthermore, this chapter provides detailed information about the Deep Neural Networks employed in this work. Firstly, Section 2.5 discusses the CNNs utilized as encoders, while Section 2.6 describes the segmentation networks, also known as decoders. Additionally, Section 2.7 introduces the concept of Generative Adversarial Networks (GANs) and provides an overview of the GANs used in this work. Finally, Section 2.8 offers concluding remarks on the discussed neural networks.

2.1 SEMANTIC SEGMENTATION

Semantic Segmentation is a computer vision task that divides an image into multiple semantic regions and each pixel receives a label representing the object's class or region. In other words, the goal is to classify each pixel in an image into a set of predefined classes (Thoma, 2016; Cao and Bao, 2020). To illustrate that, Figure 2.1 presents an example of the Semantic Segmentation problem. Sub-figure 2.1(a) presents an image with different objects (bottle, cub, cube) and Sub-figure 2.1(b) presents the corresponding segmentation, with each pixel receiving a different color to represent its class. The background is represented in yellow, the bottle in blue, the cube in purple, and the cup in green.



(a) (b)

Figure 2.1: Example of Semantic Segmentation. Sub-figure 2.1(a) presents an image with different objects (bottle, cub, cube) and Sub-figure 2.1(b) presents the corresponding segmentation, with each pixel receiving a different color to represent its class. The background is represented in yellow, the bottle in blue, the cube in purple, and the cup in green. Source: (Garcia-Garcia et al., 2018).

Another point to highlight about Semantic Segmentation is that it does not differentiate between the different instances of objects in the image. Sub-figure 2.1(b) presents the three cubes receiving the same label and color. Moreover, Semantic Segmentation differs from other segmentation tasks in that it aims to classify an entire image rather than just a specific object or region, making it a useful tool for many applications, such as object detection, image recognition, and autonomous driving (Garcia-Garcia et al., 2018; Minaee et al., 2021).

Nowadays, the state of art methods proposed to solve Semantic Segmentation problems are based on Deep Learning techniques (Long et al., 2015; Garcia-Garcia et al., 2017; Li et al., 2018; Csurka et al., 2023; Vezakis et al., 2024). Deep Learning techniques, especially Convolutional Neural Networks (CNNs), have successfully tackled the Semantic Segmentation problem (Cao and Bao, 2020; Minaee et al., 2021; Csurka et al., 2023). These models can learn to automatically extract features from images and classify each pixel into a specific class, even in complex scenarios with multiple overlapping objects and occlusions (Long et al., 2015; Garcia-Garcia et al., 2017; Li et al., 2018; Csurka et al., 2023). Section 2.3 presents an introduction to Deep Learning and how it is used to solve Semantic Segmentation problems.

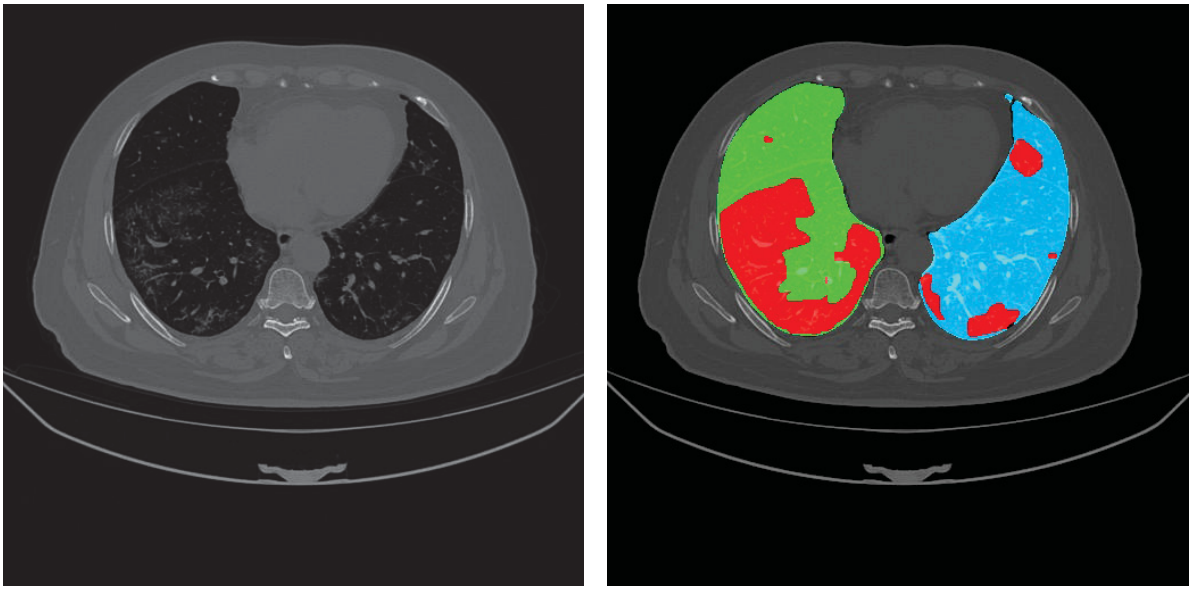
2.2 COVID-19 COMPUTED TOMOGRAPHY (CT) SEGMENTATION

Computed Tomography (CT) is a diagnostic medical imaging technique that uses X-rays and computer processing to produce detailed cross-sectional body images (of Biomedical Imaging and Bioengineering, 2021; Vincenti et al., 2022). A CT scanner takes multiple X-ray images from different angles around the body and then uses a computer to combine them into a series of 2D or 3D images. This allows doctors to see detailed images of internal organs, bones, blood vessels, and other structures in the body (of Biomedical Imaging and Bioengineering, 2021; Vincenti et al., 2022). Futhermore, CT scans are commonly used to diagnose and monitor various medical conditions, including cancer, trauma, cardiovascular disease, and neurological disorders. They can also guide minimally invasive procedures, such as biopsies and catheter insertions (of Biomedical Imaging and Bioengineering, 2021; Vincenti et al., 2022).

Figure 2.2: Example of a CT scanner that generates CT images. A CT scanner consists of a donut-shaped machine with an X-ray tube and a detector. The patient lies on a table that slides into the machine, which takes X-ray images from different angles. The images are then sent to a computer that uses complex algorithms to reconstruct them into detailed images of the body. Source: (USA, 2021).

Figure 2.2 presents an example of a CT scanner that generates CT images. A CT scanner consists of a donut-shaped machine with an X-ray tube and a detector. The patient lies on a table

that slides into the machine, which takes X-ray images from different angles. The images are then sent to a computer that uses complex algorithms to reconstruct them into detailed images of the body.



(a) An example of CT from the Zenodo dataset.

(b) The corresponding segmentation mask.

Figure 2.3: Example of CT from the Zenodo dataset (Jun et al., 2020; Ma et al., 2021). The Zenodo dataset includes labels for the left and right lungs and COVID-19 lesions. In the segmentation mask, the left lung is represented in green, the right lung in blue, and the COVID-19 lesions in red. The uncolored regions of the image represent the background and are irrelevant to the problem. Source: (Jun et al., 2020; Ma et al., 2021).

This study focuses on the segmentation of CT images of COVID-19 patients, specifically those taken from the lung area (Shi et al., 2021). This segmentation aims to identify the lung regions affected by COVID-19 and create a segmentation mask that classifies each pixel in the CT image according to its corresponding class (Shi et al., 2021). Figure 2.3 depicts a CT image of a COVID-19 patient and its corresponding segmentation mask. Sub-figure 1.1(a) shows an example CT, while Sub-figure 1.1(b) shows the corresponding segmentation mask. The dataset used in this example is Zenodo (Jun et al., 2020; Ma et al., 2021), which provides labels for left and right lungs and COVID-19 lesions. In the segmentation mask, the left lung is represented in green, the right lung in blue, and the COVID-19 lesions in red. The uncolored regions of the image represent the background and are irrelevant to the problem (Jun et al., 2020; Ma et al., 2021; Saood and Hatem, 2021).

2.3 DEEP LEARNING

Deep Learning is a sub-field of Machine Learning which uses Artificial Neural Networks composed by multiple processing layers, also called Deep Neural Networks, to solve complex learning problems (LeCun et al., 2015). Each layer in the network performs a set of computations on the input data to extract features, and the output of one layer is used as the input for the next layer. This process continues until the final layer's output is produced, representing the prediction or classification made by the model (Schmidhuber, 2015). In recent years, deep learning based methods achieved impressive results in many applications, including image and speech recognition, natural language processing, and autonomous vehicles. However, such models can be computationally intensive and require large amounts of training data to perform well (Garcia-Garcia et al., 2018).

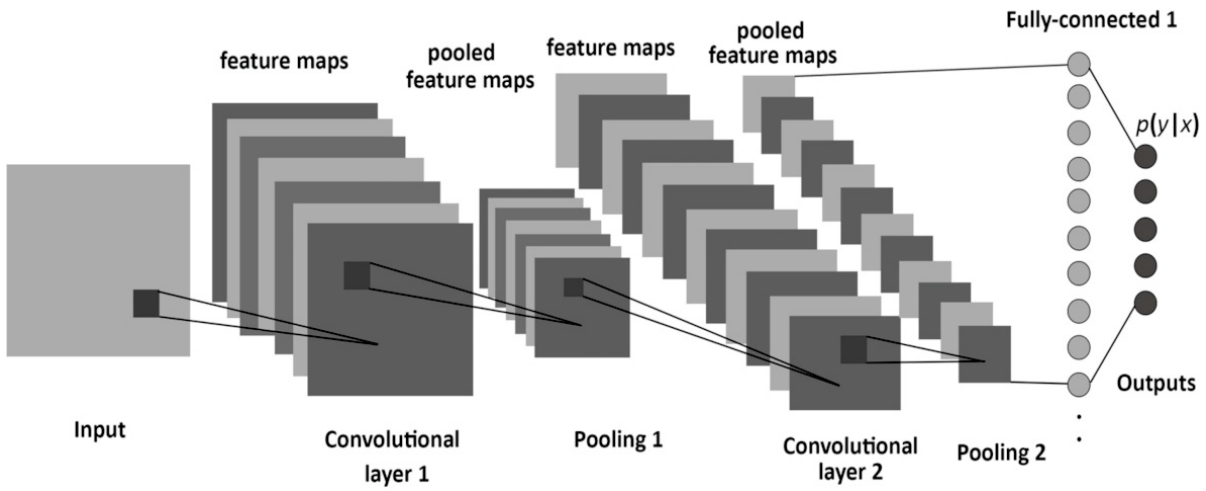


Figure 2.4: The structure of a basic CNN. The CNN can be divided into two steps: the convolutional step and the classification step. The convolutional step comprises convolutional and pooling layers to extract features from the images. The classification step is composed by fully connected layers to classify the features extracted in the convolutional step. Source: (Albelwi and Mahmood, 2017).

For image classification problems, an Artificial Neural Network called CNN (Lecun et al., 1998; Krizhevsky et al., 2017) is widely used in the literature (Garcia-Garcia et al., 2018). CNNs use convolution to analyze and learn features from the input data. The convolution operation involves sliding small filters over the input image, computing a dot product between the filter and the local region of the image it is currently overlapping, and producing a new output feature map. By stacking multiple convolutional layers in a deep neural network, CNNs can learn increasingly complex and abstract features, such as edges, shapes, textures, and patterns (Lee et al., 2015; Ankile et al., 2020). Furthermore, a CNN is divided into two steps: the convolutional step and the classification step. The convolutional step comprises convolutional and pooling layers to extract features from the images. Meanwhile, the classification step is composed by fully connected layers to classify the features extracted in the convolutional step (Albelwi and Mahmood, 2017). Figure 2.4 illustrates a basic CNN structure. For Semantic Segmentation problems, deep learning-based methods achieved impressive results through CNNs (Minaee et al., 2021). In order to adapt CNNs to perform Semantic Segmentation, the fully connected layers are replaced by upsampling layers to perform per-pixel classification (Long et al., 2015). Section 2.4 details how CNNs are adapted to perform segmentation and the structure of a segmentation CNN known as encoder-decoder architecture.

2.4 ENCODER-DECODER NETWORKS

An Encoder-Decoder network is a CNN adapted to perform images segmentation (Badrinarayanan et al., 2017). This category of deep neural networks is divided into two steps: the encoder and the decoder (Krinski et al., 2021). The encoder uses a CNN without the fully connected layers to extract features from the input image. Meanwhile, the decoder is composed by a sequence of deconvolutional and upsampling layers to convert the features extracted in the encoder into a segmentation mask (Elharrouss et al., 2021).

Figure 2.5 presents the SegNet (Badrinarayanan et al., 2017), one of the first networks proposed to perform segmentation of images. The SegNet model utilizes an encoder-decoder architecture in which the encoder comprises convolutional and pooling layers (colored in blue and green, respectively). At the same time, the decoder mirrors the encoder with the same

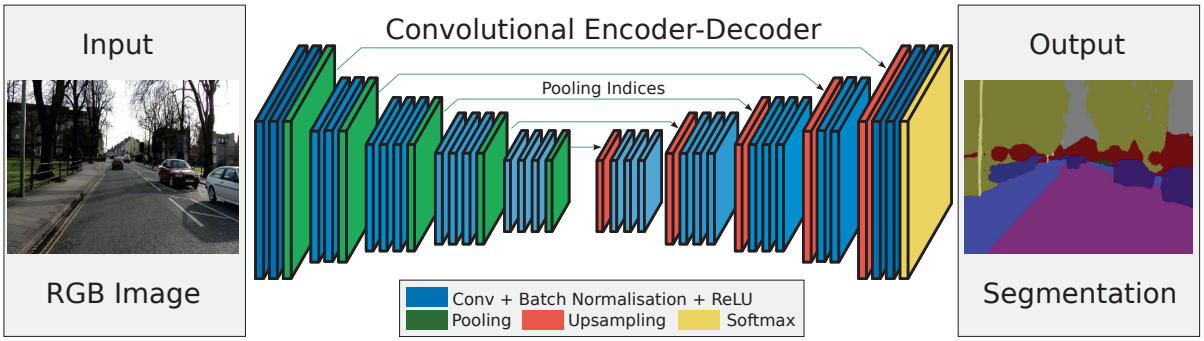


Figure 2.5: The SegNet model proposed by (Badrinarayanan et al., 2017). The SegNet model utilizes an encoder-decoder architecture in which the encoder comprises convolutional and pooling layers (colored in blue and green, respectively). At the same time, the decoder mirrors the encoder with the same convolutional layers arranged on the opposite side of the network. However, instead of pooling layers, the decoder utilizes upsampling layers (colored in red) to reconstruct the original image. The final layer of the decoder is a softmax layer (colored in yellow) that performs per-pixel classification, generating the segmentation mask. Source: (Badrinarayanan et al., 2017).

convolutional layers arranged on the opposite side of the network. However, instead of pooling layers, the decoder utilizes upsampling layers (colored in red) to reconstruct the original image. The final layer of the decoder is a softmax layer (colored in yellow) that performs per-pixel classification, generating the segmentation mask (Badrinarayanan et al., 2017).

2.5 CONVOLUTIONAL NEURAL NETWORKS (CNNS) - ENCODERS

The architecture of a CNN can vary widely depending on the specific task and input data, with many variations being proposed. This section details the seven CNNs used in this work: Visual Geometry Group (VGG) (Simonyan and Zisserman, 2015), Residual Network (ResNet) (He et al., 2016), Res2Net (Gao et al., 2021), ResNeXt (Xie et al., 2017), Dense Convolutional Network (DenseNet) (Huang et al., 2017), Squeeze-and-Excitation Networks (SE-Net) (Hu et al., 2018), and RegNet (Radosavovic et al., 2020). These models are widely used in different classification and segmentation problems, and are used in this work as encoders of encoder-decoder segmentation models.

2.5.1 Visual Geometry Group (VGG)

The Visual Geometry Group (VGG) (Simonyan and Zisserman, 2015) architecture is a CNN with promising results in classification problems (Li et al., 2022; Krichen, 2023). This CNN uses small 3×3 filters throughout the convolutional layers, which allows the network to learn more complex features with fewer parameters than larger filters. More precisely, this strategy increases the number of non-linear Rectified Linear Units (ReLu) layers and the decision functions' discrimination capacity. Moreover, the number of parameters of the architecture is reduced without decreasing the number of effective receptive fields. The original VGG architecture consists of 16 layers (VGG-16), with 13 convolutional layers and 3 fully connected layers. The convolutional layers are arranged in groups of two or three, with each group followed by a max-pooling layer. The number of filters in each convolutional layer increases as the spatial size of the feature maps decreases, with the final layer having 512 filters. There are also variations with 19 layers (VGG-19), and 11 layers (VGG-11). These variations have a similar structure to the original VGG architecture but with fewer or more layers.

Figure 2.6 presents a table with all VGG architectures proposed (VGG-11, VGG-13, VGG-16, and VGG-19), varying the number of layers in the network. The first layer of all

VGG models is the original configuration's input layer of size 224×224. Then, a sequence of convolutional blocks is linked in the VGG models, with each convolutional block being composed by a stack of convolutional layers. The VGG-11 and VGG-13 are composed by convolutional blocks with one and two convolutional layers, while the VGG-16 and VGG-19 are deeper architectures, with convolutional blocks of three and four convolutional layers. Between the convolutional blocks, a max-pooling layer is attached to reduce the dimension of the feature maps. After the last max-pooling layer, three fully connected and a softmax layers are attached to classify the feature extracted in the convolutional blocks.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 2.6: Table with all VGG architectures proposed (VGG-11, VGG-13, VGG-16, and VGG-19). The VGG-11 and VGG-13 are composed by stacks of two convolutional layers, while the VGG-16 and VGG-19 are deeper architectures, with stacks of three and four convolutional layers. Source: (Simonyan and Zisserman, 2015).

2.5.2 Residual Networks (ResNet)

The Residual Network (ResNet) (He et al., 2016) is also a promising CNN architectures used in classification problems (Li et al., 2022; Krichen, 2023). This model aims to solve the vanishing gradients problem in very deep neural networks by introducing skip connections, or residual connections, that allows information to flow directly from the input of a layer to the output of a subsequent layer. In a standard deep neural network, each layer takes the previous layer's output as its input. As the number of layers increases, the gradients can become very small or even vanish, making it difficult to train the network effectively. The ResNet addresses this problem by

adding residual connections that allow the input of a layer to be added directly to the output of a subsequent layer, bypassing the intermediate layers, which ensures that the gradients can still flow through the network, even in very deep architectures. An example of skip connection is presented in Figure 2.7. The input of the first weight (convolutional) layer "x" is linked through a skip connection to the output of the second weight layer "$F(x)$". A sequence of layers linked through a skip connection is called building block.

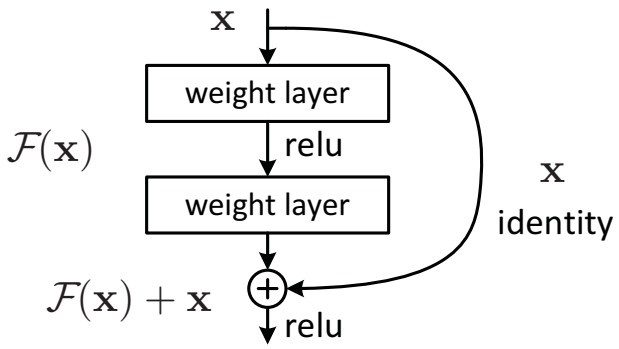


Figure 2.7: The weight layers are convolutional blocks, and the input of the first weight layer ("x") is linked through a skip connection to the output of the second weight layer ("$F(x)$"). Source: (He et al., 2016).

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 2.8: Table with details about each ResNet version proposed by (He et al., 2016). Five versions was proposed: ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152. Source: (He et al., 2016).

The basic building block of a ResNet, also called residual block, consists of two or more convolutional layers followed by a skip connection that adds the input of the block to its output. ResNets can achieve very deep architectures by stacking multiple residual blocks while maintaining good performance. Figure 2.8 presents a table with details of the layers of each ResNet version proposed by (He et al., 2016). Five versions have been proposed: ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152. The major difference between each proposed version is the number of residual blocks attached in sequence and thus varying the network depth. Also, the ResNet-18 and ResNet-34 are composed by blocks with only two convolutional blocks, as exemplified in Figure 2.8. Meanwhile, the ResNet-50, ResNet-101, and ResNet-152 are composed by blocks with three convolutional blocks, and the input of the first convolutional layer is linked to the output of the last convolutional layer in the block.

2.5.3 ResNeXt

The ResNeXt (Xie et al., 2017) aims to increase the number of receptive fields and improve the ability of the network to find objects in the image by adding parallel paths. It is a variation of the ResNet architecture, which uses residual blocks to enable the training of very deep neural networks. The ResNeXt extends ResNet by introducing a new "cardinality" module that increases the network's representational power without increasing its depth. The cardinality is a hyperparameter that controls the number of independent paths or "cardinalities" within each network block. In other words, the ResNet uses just one path with a large number of filters (large depth), while the ResNeXt uses many parallel paths with a small number of filters (small depth).

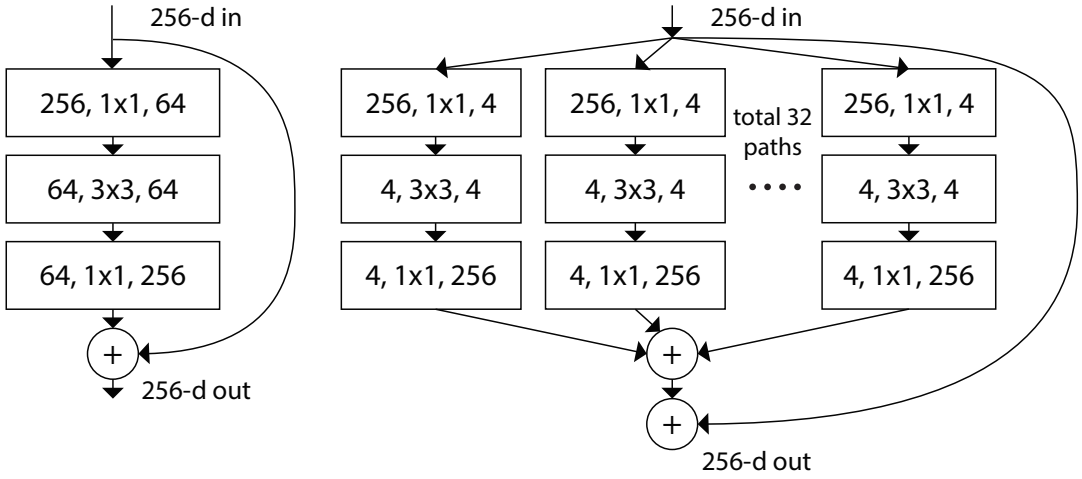


Figure 2.9: Comparison between ResNet and ResNext. In the left, the ResNet is presented, with only one path with filters of size 64, and in the right, the ResNext is presented with Cardinality 32 and filters of size 4. Source: (Xie et al., 2017).

Each block in ResNeXt consists of a set of parallel pathways, where each pathway is a small CNN. These pathways are aggregated through a summation operation, which allows the model to capture diverse patterns in the input data. Multiple pathways allow ResNeXt to achieve high accuracy on a wide range of image classification tasks with fewer parameters than previous state-of-the-art models and has demonstrated state-of-the-art performance on benchmark datasets such as ImageNet Large Scale Visual Recognition Challenge (ImageNet) (Deng et al., 2009) and CIFAR-10/100 (Krizhevsky et al., 2009). More precisely, the ResNeXt replaces the standard path of ResNet with layers of X filters to C parallel paths with layers of d filters, where C is the cardinality, and d is the depth and equals to $X/16$. Figure 2.9 presents the difference between both architectures. In the left, the ResNet is presented with only one path with filters of size 64. In the right, the ResNext is presented with cardinality 32 and filters of size 4. At the end of the ResNeXt, all 32 paths are aggregated to produce the same size output of the ResNet. Also, the skip connection used in the ResNet works in the same way in the ResNeXt, with the input of the block being aggregated with the output of the block.

2.5.4 Dense Convolutional Network (DenseNet)

The Dense Convolutional Network (DenseNet) (Huang et al., 2017) is a based on the idea of densely connected convolutional layers. In a traditional CNN, information flows from the input to the output layer through a sequence of convolutional layers, with occasional pooling and fully connected layers. In contrast, DenseNet connects each layer to every other layer in a feed-forward

fashion. This connectivity pattern creates a dense block, where each layer receives inputs from all preceding layers and passes its output to all subsequent layers.

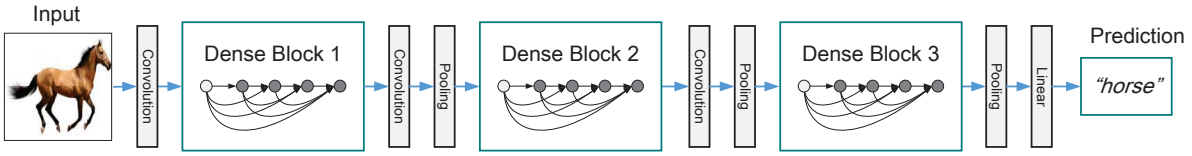


Figure 2.10: DenseNet architecture with three dense blocks and each dense block with five blocks. The first block (white circle) output in each dense block is concatenated with the input of all following blocks (grey circles). The same is made for all blocks inside the dense block. Source: (Huang et al., 2017).

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Figure 2.11: Table with details about each DenseNet version proposed by (Huang et al., 2017). Four versions was proposed: DenseNet-121, DenseNet-169, DenseNet-201, and DenseNet-264. Each dense block is composed by a sequence of two convolutional layers of size 1×1 and 3×3 repeated n times, with n varying depending on the DenseNet version. Source: (Huang et al., 2017).

The main advantages of the DenseNet are its ability to reduce the number of parameters needed in the network while maintaining high accuracy, improving gradient flow, and facilitating feature reuse. In addition, the DenseNet alleviates the vanishing gradient problem by propagating gradients directly from the output to all layers in the block. A typical DenseNet architecture consists of multiple dense blocks, where each dense block is followed by a transition layer that performs downsampling and feature map compression. The last dense block is followed by a global average pooling layer and a fully connected layer for classification. Figure 2.10 presents an example of DenseNet architecture with three dense blocks and each dense block with five blocks. The first block (white circle) output in each dense block is concatenated with the input of all following blocks (grey circles). The same is made for all blocks inside the dense block. The convolutional and pooling layers between two dense blocks are called transition layers.

Figure 2.11 presents a table with details about each DenseNet version proposed by (Huang et al., 2017). Four versions was proposed: DenseNet-121, DenseNet-169, DenseNet-201, and DenseNet-264. Each dense block is composed by a sequence of blocks repeated n times, with n varying depending on the DenseNet version. The major difference between each proposed

version is the number of blocks on third and fourth dense blocks, with the larger versions of DenseNet built with more blocks inside these two dense blocks.

2.5.5 Squeeze-and-Excitation Networks (SE-Net)

The Squeeze-and-Excitation Networks (SE-Net) (Hu et al., 2018) aims to selectively emphasize informative features and suppress unimportant information by adaptively recalibrating the feature maps. In traditional CNNs, feature maps are learned independently and equally contribute to the final prediction, regardless of their importance. In contrast, SE-Net introduces a novel module called the "Squeeze-and-Excitation block" that can adaptively recalibrate feature maps. The block consists of two main operations: a squeeze operation that globally aggregates the spatial dimensions of a feature map into a channel descriptor and an excitation operation that learns to weigh each channel descriptor before rescaling the feature maps. Figure 2.12 presents the difference between a standard convolutional block in the left side and a SE-Net convolutional block in the right side. The SE-Net convolutional block has an additional building block to weigh the channels, built with the following layers: average pooling, Fully Connected (FC), ReLu, FC, Sigmoid. The average pooling operation squeezes the output of the convolutional block. Then, the FC and ReLu layers add non-linearity, and the final FC and Sigmoid layers activate the channels. In the end, each channel of the convolutional block is weighted based on the output of the proposed building block.

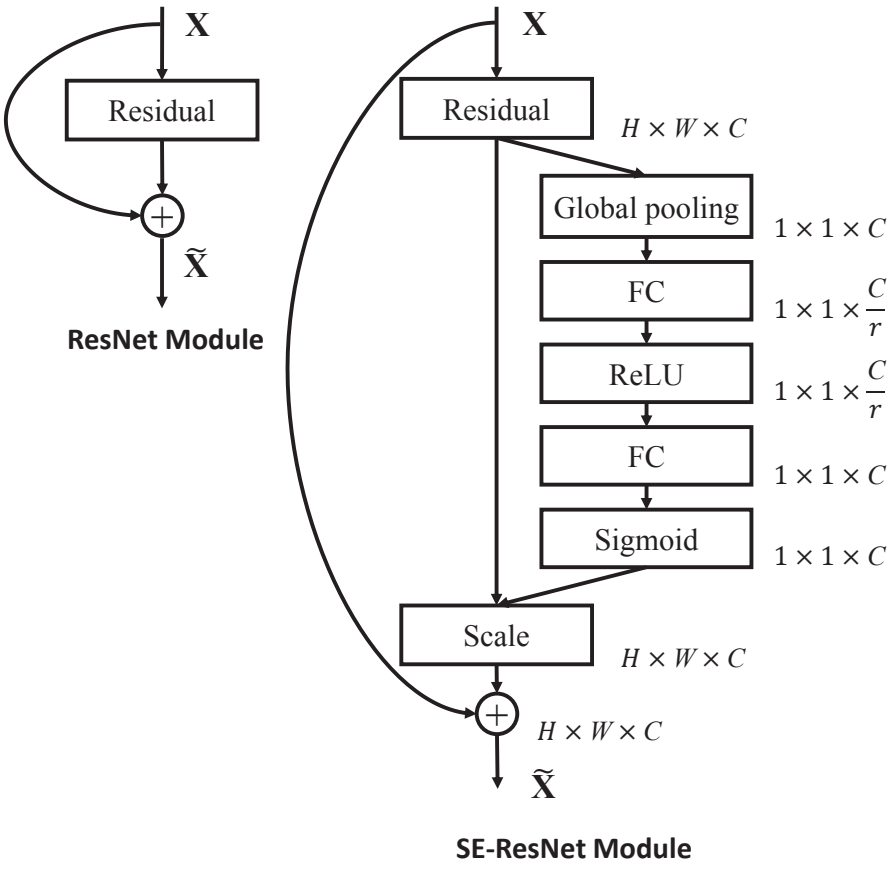


Figure 2.12: Difference between a standard convolutional block in the left side and a SE-Net convolutional block in the right side. The SE-Net convolutional block has an additional building block to weigh the channels, built with the following layers: average pooling FC, ReLu, FC, Sigmoid. Source: (Hu et al., 2018).

The squeeze operation reduces the spatial dimensions of the feature maps into a single-channel descriptor by applying global average pooling. The excitation operation then takes the descriptor and applies two fully connected layers, followed by a sigmoid activation function to learn a set of channel-wise scaling factors. These scaling factors are then used to rescale the feature maps to emphasize informative channels and suppress unimportant ones. Furthermore, the SE-Net can be easily integrated into existing CNN architectures as a plug-and-play module without significant additional computational overhead. It has achieved promising results on image classification problems (Ankile et al., 2020; Li et al., 2022; Krichen, 2023), including the ImageNet (Deng et al., 2009) dataset. Moreover, SE-Net has shown to be effective in other computer vision tasks, such as object detection and segmentation, as well as in other domains, such as speech recognition and natural language processing.

2.5.6 RegNet

The RegNet (Radosavovic et al., 2020) uses a design space search algorithm to automatically discover a family of efficient network architectures tailored to a specific computational budget. The design space search algorithm explores the space of possible network architectures and selects the ones that achieve the best trade-off between accuracy and computational cost. The RegNet architecture consists of a series of "Bottleneck Blocks" stacked on top of each other to form the full network. Each Bottleneck Block contains a sequence of 3×3 convolutional layers, batch normalization, and ReLu activation functions. The main innovation in RegNet is the "grouped convolutions", where the convolutional filters are partitioned into multiple groups, each operating on a different subset of the input channels. This allows more efficient use of computational resources and reduces the number of parameters needed in the network.

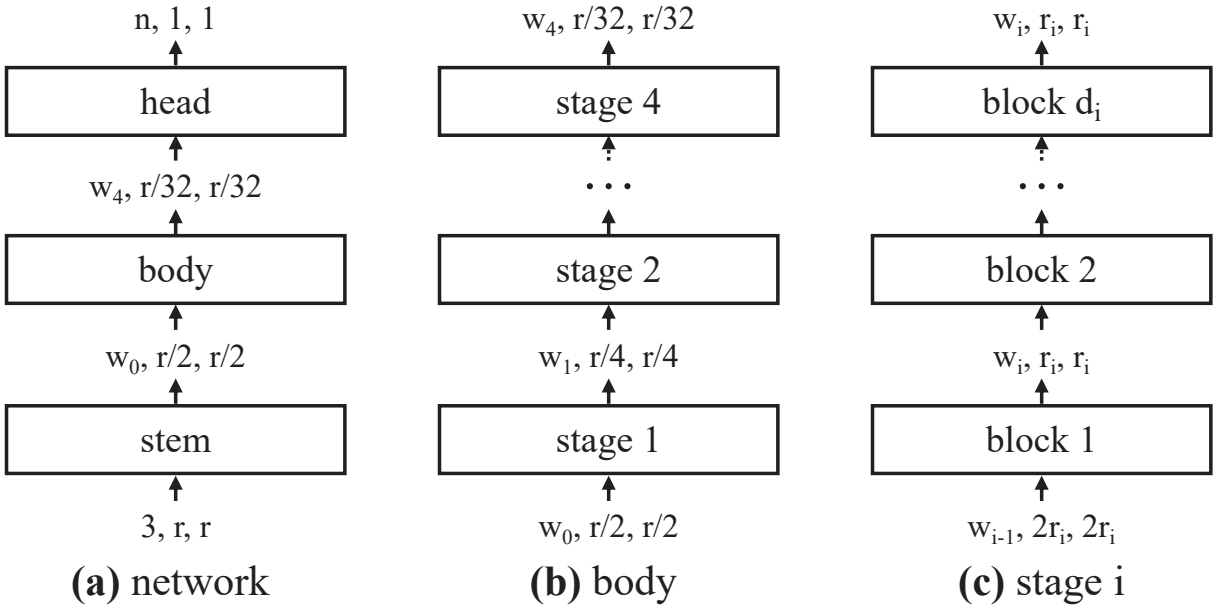


Figure 2.13: The general RegNet structure. The RegNet incorporates a general network structure that can be broken down into three main components: a stem, a network body, and a head. The stem, which comprises a stride-two 3×3 convolution with 32 output channels (w_0), serves as the initial input to the network. The network body, which is responsible for most of the computation, follows the stem and is then followed by a head that utilizes average pooling and a fully connected layer to generate predictions for n output classes.Source: (Radosavovic et al., 2020).

In addition, RegNet uses a "SE-ResNet-style" skip connection between each pair of Bottleneck Blocks, which helps to propagate gradients and stabilize the training process. The RegNet achieves state-of-the-art performance on several image classification benchmarks,

including the ImageNet (Deng et al., 2009) dataset, while being computationally efficient and easy to train. Moreover, RegNet is effective in transfer learning and object detection tasks, demonstrating its versatility and scalability. Figure 2.13 presents the general RegNet structure. The RegNet incorporates a general network structure that can be broken down into three main components: a stem, a network body, and a head. The stem, which comprises a stride-two 3×3 convolution with 32 output channels (w_0), serves as the initial input to the network. The network body, which is responsible for most of the computation, follows the stem and is then followed by a head that utilizes average pooling and a fully connected layer to generate predictions for n output classes. The network body comprises a series of stages that operate at progressively reduced resolutions, r_i. Each stage contains a sequence of identical blocks, except for the first block, which uses stride-two convolution. Although the overall structure of the RegNet is straightforward, the number of possible network configurations is vast due to the many possible combinations of stage and block hyperparameters.

2.5.7 Res2Net

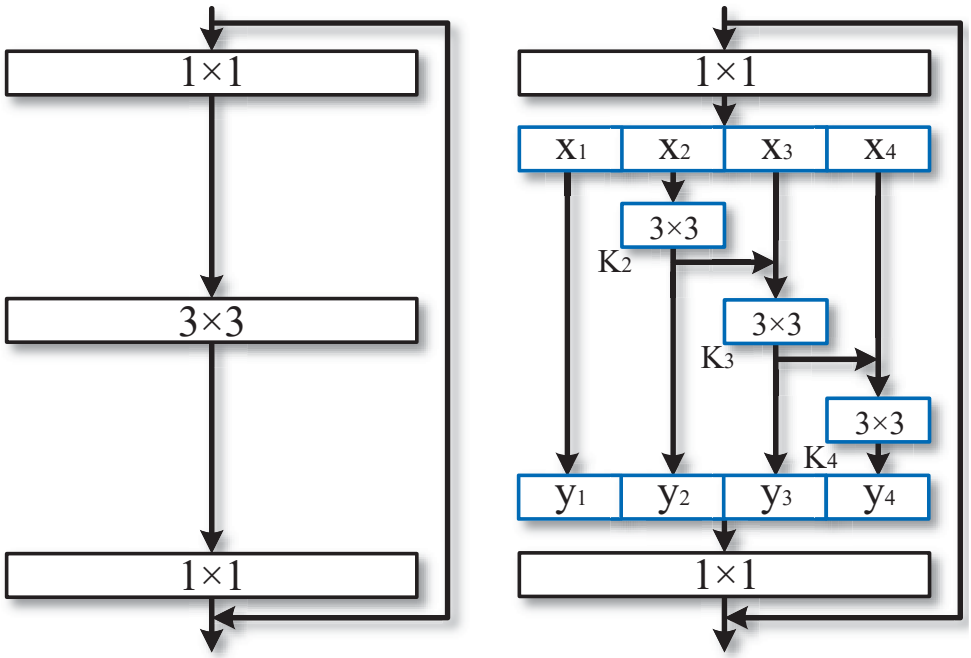


Figure 2.14: Difference between the Res2net, in the right, and the ResNet, in the left. A multi-scale module replaces the middle convolutional layer of size (3×3) in the ResNet with a set of smaller filter groups. Source: (Gao et al., 2021).

The Res2Net (Gao et al., 2021) aims to leverage the multi-scale representations of deep CNNs by introducing the Residual Pooling Unit (RPU) and achieved promising results in classification problems (Heo et al., 2023; Yang and Zhang, 2024). This module enables efficient feature reuse across different scales. The RPU module combines scale features by applying multiple parallel convolutions with different kernel sizes on the input feature maps. These parallel convolutions are then aggregated using a weighted sum, which allows the network to capture fine-grained details and global context information in the input data. More precisely, the standard block of ResNet is replaced by a 4-scale-(3×3) residual hierarchical architecture. A multi-scale module replaces the middle convolutional layer of size (3×3) in the ResNet with a set of smaller

filter groups. With this strategy, the number of receptive fields inside the block increases, which enables the network to grab the different scale levels for the objects inside the image. Figure 2.14 presents the difference between the Res2net in the right, and the ResNet, in the left. Also, this strategy can be used on both architectures, ResNet and ResNeXt.

The multi-scale module divided the feature maps in K smaller feature maps $(X_1, X_2, X_3, X_4, ...)$. The first slice X_1 goes without any convolutional operation. The second slice suffers a convolutional operation. The second slice is added to the third slice and suffers a convolutional operation. The third slice is added to the fourth slice and suffers a convolutional operation, and so on depending on the value of K. With this strategy, the number of receptive fields is increased, and the CNN extracts more granular features in different levels of scale and improves the ability of the network to find objects in the image. One of the main advantages of the Res2Net over other state-of-the-art models is that it can achieve better performance with fewer parameters and computation, making it more efficient for real-world applications. Res2Net has demonstrated state-of-the-art performance on benchmark datasets, including ImageNet (Deng et al., 2009) and COCO (Lin et al., 2014).

2.6 SEMANTIC SEGMENTATION NETWORKS - DECODERS

This section details the six Semantic Segmentation Networks used in this work: U-Net (Ronneberger et al., 2015), U-Net++ (Zhou et al., 2018), Feature Pyramid Network (FPN) (Lin et al., 2017), Pyramid Scene Parsing Network (PSPNet) (Zhao et al., 2017), LinkNet (Chaurasia and Culurciello, 2017), and Multi-scale Attention Net (MA-Net) (Fan et al., 2020b). These models are widely used for Semantic Segmentation problems, with U-Net and U-Net++ particularly prevalent in medical segmentation tasks. In this work, they are utilized as decoders in encoder-decoder segmentation models.

2.6.1 U-Net

The U-Net (Ronneberger et al., 2015) architecture comprises an encoder and a decoder network connected by a bottleneck layer. The encoder network is similar to a standard CNN architecture, consisting of a series of convolutional and pooling layers that gradually reduce the spatial dimensions of the input image. On the other hand, the decoder network consists of a series of convolutional and upsampling layers that gradually increase the spatial dimensions of the input.

The U-Net architecture has skip connections between the encoder and decoder networks. These skip connections concatenate the feature maps from the encoder network with the corresponding feature maps in the decoder network, allowing the decoder network to reconstruct the fine-grained details of the original input image. Another important aspect of the U-Net architecture is the "cropping" in the decoder network. This cropping removes the unnecessary border regions in the feature maps generated by the upsampling layers in the decoder network, allowing the reconstructed image to have the same size as the original input image. Figure 2.15 presents the structure of a U-Net architecture. The blue boxes represent the feature maps. In the left side, the red arrows are Max-Pooling operations to reduce the dimension of the feature maps, and in the right side, the green arrows are transposed convolutions to reconstruct the dimension of the feature map. The grey arrow indicates the concatenation of the feature map generated on the encoder side, with the feature map generated on the decoder side.

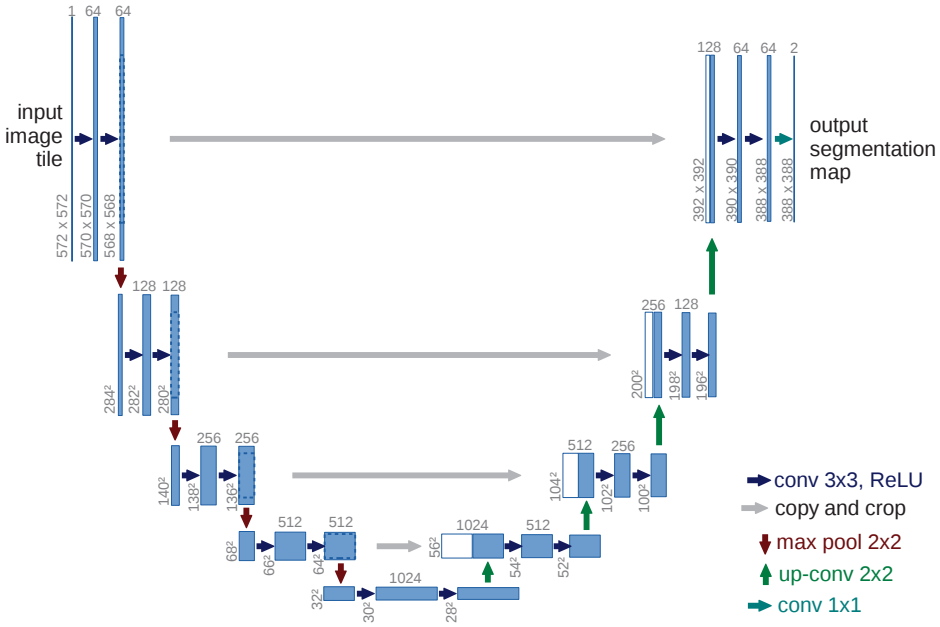


Figure 2.15: The complete structure of a U-Net architecture. The blue boxes represent the feature maps. In the left size, the red arrows are Max-Pooling operations to reduce the dimension of the feature maps, and in the right side, the green arrows are transposed convolutions to reconstruct the dimension of the feature map. The grey arrow indicates the concatenation of the feature map generated on the encoder side, with the feature map generated on the decoder side. Source: (Ronneberger et al., 2015).

2.6.2 Feature Pyramid Network (FPN)

The Feature Pyramid Network (FPN) (Lin et al., 2017) aims to construct a pyramid of feature maps at multiple scale, which are then used to detect objects at different sizes and resolutions. This is achieved by combining feature maps from different levels of a CNN into a single multi-scale representation. Specifically, FPN constructs a pyramid of feature maps by taking the output of each level of the CNN and upsampling it to the same spatial resolution as the next higher-level feature map. This creates a set of feature maps at different scales, with the lower-level feature maps capturing finer details and the higher-level feature maps capturing more global information.

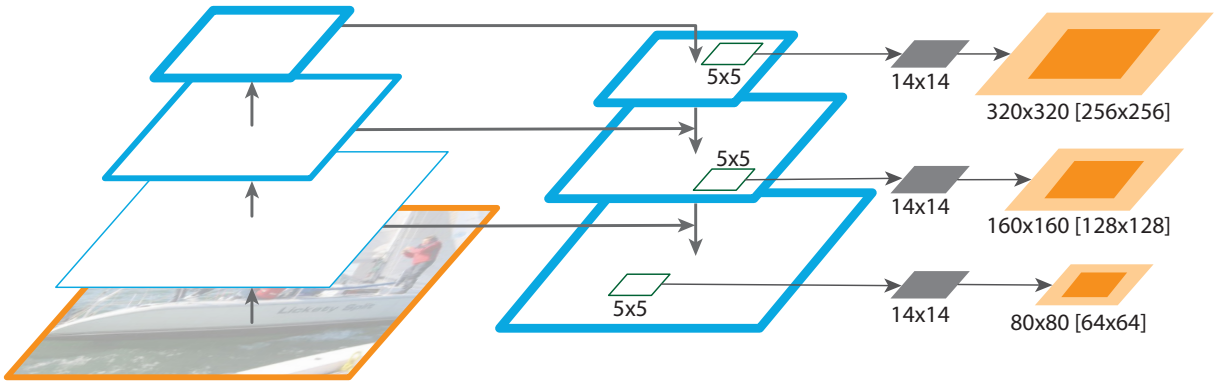


Figure 2.16: The Feature Pyramid Network (FPN) structure proposed by (Lin et al., 2017). The blue squares are convolutional and upsample blocks, and the orange squares represent the segmentation masks generated in each level of the decoder. Source: (Lin et al., 2017).

To improve the quality of the multi-scale representation, the FPN introduces a "top-down" pathway that passes information from the higher-level feature maps to the lower-level feature maps. This pathway consists of a series of lateral connections that perform 1×1

convolutions to match the number of channels in the higher-level feature maps with the number of channels in the corresponding lower-level feature maps. These lateral connections concatenate the upsampled higher-level feature maps with the lower-level feature maps, producing a final multi-scale representation. The resulting FPN architecture can be used for object detection by attaching Region Proposal Networks (RPNs) to each level of the feature pyramid. The RPNs use the multi-scale feature maps to generate a set of candidate object proposals at different scales and resolutions, which are then refined and classified by a subsequent detection network. Figure 2.16 presents the FPN structure. The blue squares are convolutional and upsample blocks, and the orange squares represent the segmentation masks generated in each level of the decoder.

2.6.3 Pyramid Scene Parsing Network (PSPNet)

The Pyramid Scene Parsing Network (PSPNet) (Zhao et al., 2017) has the same objective as the FPN: construct a pyramid of feature maps at multiple scales, which are then used to perform semantic segmentation at different levels of granularity. This is achieved by using a global pooling layer to capture the context of the entire image, combined with a series of convolutional layers to extract features at different scales. The PSPNet uses a modified version of the ResNet architecture as the backbone network, which is pre-trained on large-scale image classification datasets. The output of the last convolutional layer in the ResNet is passed through a global average pooling layer, which produces a global feature vector that captures the context of the entire image. To capture features at different scales, the PSPNet then applies a series of convolutional layers with different dilation rates to the output of the last convolutional layer in the ResNet. A collection of feature maps at varying scales is generated as a result of applying convolutional layers with different dilation rates to the output of the last convolutional layer in the ResNet. The feature maps at lower levels can capture finer details, while the ones at higher levels capture more global information.

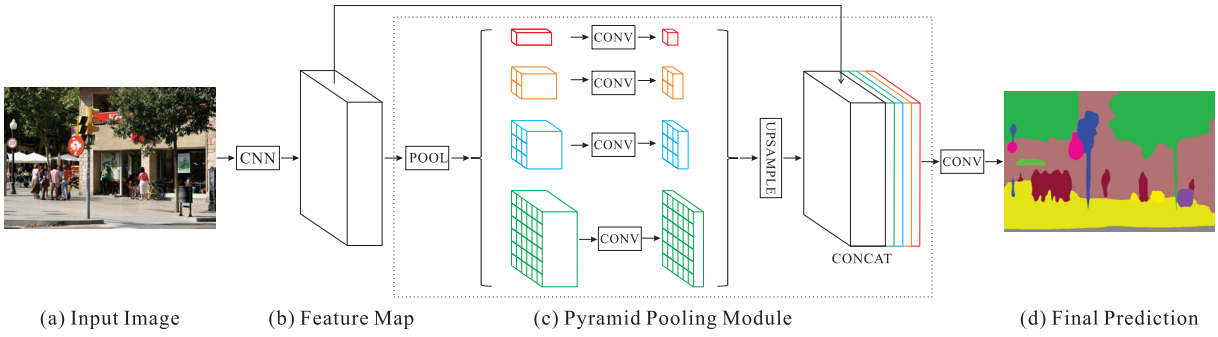


Figure 2.17: The PSPNet proposed by (Zhao et al., 2017). The input image presented in (a) inputs a CNN module to extract features maps, presented in (b). Then, in the Pyramid Pooling Module (c), the pooling operation is applied with four different sizes, generating four parallel feature maps with different scales. These four feature maps cross a sequence of convolutional layers until being upsampled and concatenated again to generate the final feature map, which will be used to generate the final prediction, as presented in (d). Source: (Zhao et al., 2017).

To fuse the multi-scale feature maps, the PSPNet introduces a "pyramid pooling module" that uses global max pooling to extract information from each of the feature maps at multiple scales. The resulting features are then concatenated and passed through a series of convolutional layers to produce a final segmentation map. Figure 2.17 presents the structure of the PSPNet. The input image presented in (a) inputs a CNN module to extract features maps, presented in (b). Then, in the Pyramid Pooling Module (c), the pooling operation is applied with four different sizes, generating four parallel feature maps with different scales. These four feature maps cross a

sequence of convolutional layers until being upsampled and concatenated again to generate the final feature map, which will be used to generate the final prediction, as presented in (d).

2.6.4 LinkNet

The LinkNet (Chaurasia and Culurciello, 2017) uses a skip connection architecture inspired by the ResNet architecture to improve the flow of information between the encoder and decoder. The encoder module uses a modified ResNet-18 architecture with fewer filters, which helps reduce the computational cost of the model. The decoder module includes upsampling blocks, each consisting of a bilinear upsampling layer followed by a convolutional layer. The skip connections in LinkNet are added between the encoder and decoder at each stage of the encoder, allowing for the transfer of high-resolution information from the encoder to the decoder. As presented in Figure 2.18, the LinkNet is built with a sequence of encoder and decoder blocks. In the first step of the network, the LinkNet has a sequence of encoder blocks, with each encoder block being a sequence of convolutional layers to extract features from the image and reduce the dimension of the feature maps. The second step has a sequence of decoder blocks, with each decoder block being a sequence of convolutional and fully-convolutional layers (Long et al., 2015) to rebuild the image and generate the segmentation mask.

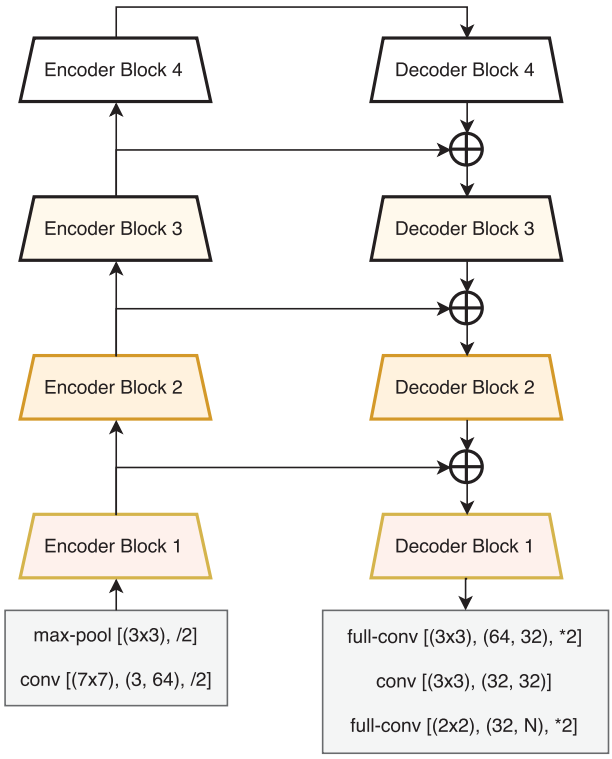


Figure 2.18: The LinkNet architecture proposed by (Chaurasia and Culurciello, 2017). The LinkNet is built with a sequence of encoder and decoder blocks. Also, to mitigate the information loss, each encoder block has a connection to its corresponding decoder block. Source: (Chaurasia and Culurciello, 2017).

In addition, LinkNet uses batch normalization and ReLu activation functions throughout the network to improve training and reduce overfitting. The final layer of the decoder module uses a softmax activation function to produce the pixel-wise classification map. Figure 2.19 presents the structure of an encoder block. Each encoder block has a sequence of convolutional blocks, with each convolutional block is composed by two convolutional layers, and the input of the first convolutional layer is skipped and added to the output of the second convolutional layer.

Figure 2.20 presents the structure of a decoder block. Each decoder block has three layers in the following sequence: convolutional layer, fully-convolutional layer, and another convolutional layer. Before the first encoder block, the LinkNet has the first convolutional and max-pooling layers, and after the last decoder block, the LinkNet has a fully-convolutional layer followed by a convolutional and a fully-convolutional layers. Also, to mitigate the information loss, each encoder block has a connection to its corresponding decoder block.

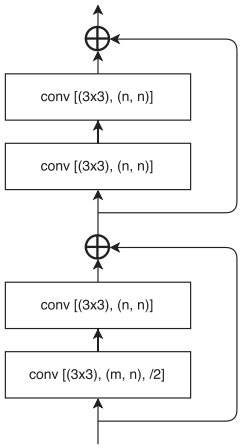


Figure 2.19: The LinkNet encoder structure proposed by (Chaurasia and Culurciello, 2017). Each encoder block has a sequence of convolutional blocks, with each convolutional block is composed by two convolutional layers, and the input of the first convolutional layer is skipped and added to the output of the second convolutional layer. Source: (Chaurasia and Culurciello, 2017).

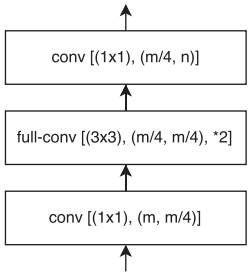


Figure 2.20: The LinkNet decoder structure proposed (Chaurasia and Culurciello, 2017). Each decoder block has three layers in the following sequence: convolutional layer, fully-convolutional layer, and another convolutional layer. Source: (Chaurasia and Culurciello, 2017).

2.6.5 U-Net++

The U-Net++ (Zhou et al., 2018) is based on the same encoder-decoder structure as the U-Net, but it introduces a more complex skip connection structure between the encoder and decoder networks. Instead of a single set of skip connections, the U-Net++ architecture uses multiple levels of nested skip connections, with each level aggregating feature maps from a larger range of scales. Specifically, the U-Net++ architecture introduces a "recursive U-Net" structure, where each level of the encoder network is itself a U-Net architecture that generates multiple sets of feature maps at different scales. The corresponding levels in the decoder network then concatenate these feature maps across the different scales, using a weighted sum based on the similarity between the feature maps. This helps to capture more fine-grained details and improve the accuracy of the segmentation.

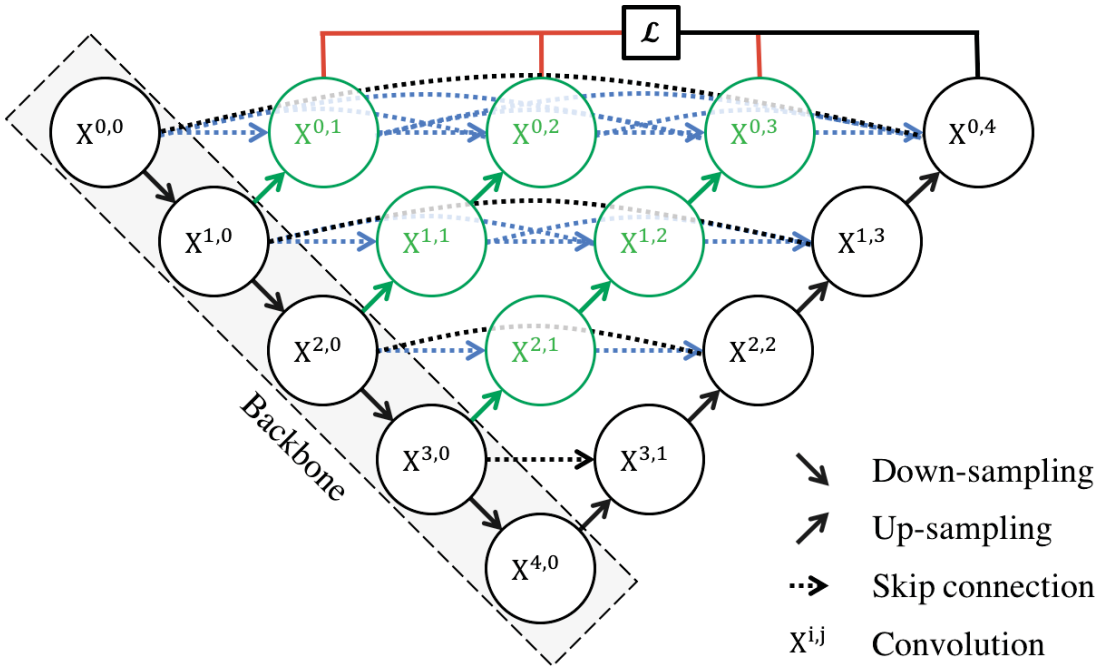


Figure 2.21: The U-Net++ architecture proposed by (Zhou et al., 2018). The black circles and arrows show the standard U-Net architecture. The green and blue regions, circles, and arrows are the dense convolutional blocks, and the red structure is the deep supervised module. Source: (Zhou et al., 2018).

In addition to the nested skip connections, the U-Net++ architecture also introduces a "deep supervision" mechanism, where intermediate feature maps are fed to auxiliary classifiers that predict the segmentation masks at multiple scales. This helps to regularize the training process and improve the model's overall performance. Figure 2.21 presents the U-Net++ complete structure. The black circles and arrows show the standard U-Net architecture. The green and blue regions, circles, and arrows are the dense convolutional blocks, and the red structure is the deep supervised module.

2.6.6 Multi-scale Attention Net (MA-Net)

The MA-Net (Fan et al., 2020b) uses attention mechanisms to selectively focus on different regions of input images at different scales. The architecture is designed to extract relevant features from images and consists of three main components: a feature extractor, a multi-scale attention module, and a classifier. The feature extractor is a CNN that processes the input image and produces a set of feature maps. The multi-scale attention module operates on these feature maps and uses a set of attention mechanisms to selectively attend to different regions of the image at multiple scales. The attention mechanisms are designed to identify salient features important for the task and suppress irrelevant information. The multi-scale attention module also includes a set of convolutional layers that combine the attended feature maps across different scales to produce a final set of feature maps. These feature maps are then passed through a classifier, which produces the network's final output.

The MA-Net has the ability to selectively attend to different regions of an image at multiple scales, making it well-suited for tasks requiring fine-grained image feature analysis. Figure 2.22 presents the MA-Net architecture. The encoder is composed by a sequence of residual blocks (Res-Blocks) and convolutional blocks, presented in white and blue, respectively. The residual blocks are used to mitigate the vanishing gradient problem. The decoder step is composed by a sequence of Multi-scale Fusion Attention Block (MFAB) and Up-Sampling layers,

presented in red and purple, respectively. The Multi-scale Fusion Attention Block (MFAB) layer is a structure proposed by the authors that use SE-Net (Hu et al., 2018) blocks to extract the interdependence between feature channels by combining High and Low-level feature maps. The MA-Net also has a Position-wise Attention Block (PAB) module, presented in yellow, between the encoder and decoder steps to capture spatial dependencies between any two position feature maps. Also, for each block in the encoder step, a skip connection is linked to the corresponding block in the decoder step to mitigate the information loss.

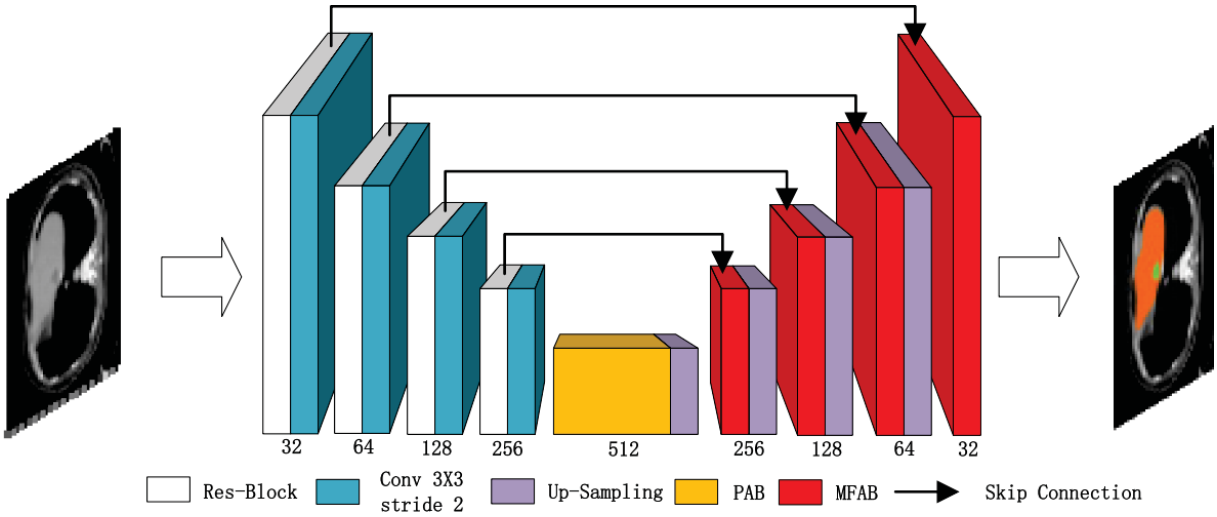


Figure 2.22: The MA-Net architecture proposed by (Fan et al., 2020b). The encoder is composed by a sequence of residual blocks (Res-Blocks) and convolutional blocks, presented in white and blue, respectively. The decoder step is composed by a sequence of MFAB and Up-Sampling layers, presented in red and purple, respectively. The authors proposed a Position-wise Attention Block (PAB) module between the encoder and decoder steps, presented in yellow. Also, for each block in the encoder step, a skip connection is linked to the corresponding block in the decoder step. Source: (Fan et al., 2020b).

2.7 GENERATIVE ADVERSARIAL NETWORKS (GANS)

GANs are deep learning algorithm designed to generate images and consists of two neural networks: a generator and a discriminator (Goodfellow et al., 2014). The generator is trained to create synthetic data that resembles real data, while the discriminator is trained to distinguish between synthetic and real data (Goodfellow et al., 2020). During training, the generator generates fake data and presents it to the discriminator, which then tries to distinguish between the real and synthetic data. The feedback from the discriminator is used to update the generator to generate synthetic data that is increasingly similar to the real data (Ruiz et al., 2020a).

The main idea behind GANs is to have the generator and discriminator competing against each other, with the generator trying to generate more realistic data and the discriminator trying to better distinguish between real and synthetic data. This competition between the two networks drives the learning process, with both networks continually improving until the generator produces synthetic data that is indistinguishable from real data (Choi et al., 2018; Karras et al., 2019). GANs have been successfully applied to various applications, including face generation (Karras et al., 2018), image-to-image translation (Isola et al., 2017), inpainting (Pathak et al., 2016), video generation (Vondrick et al., 2016), and 3d-modeling (Wu et al., 2016).

Figure 2.23 presents the structure of a GAN proposed by (Goodfellow et al., 2014). The main goal of the generator in a GAN is to generate synthetic data. The generator step is a CNN in reverse mode. When a CNN is used for a classification task, it receives an image as

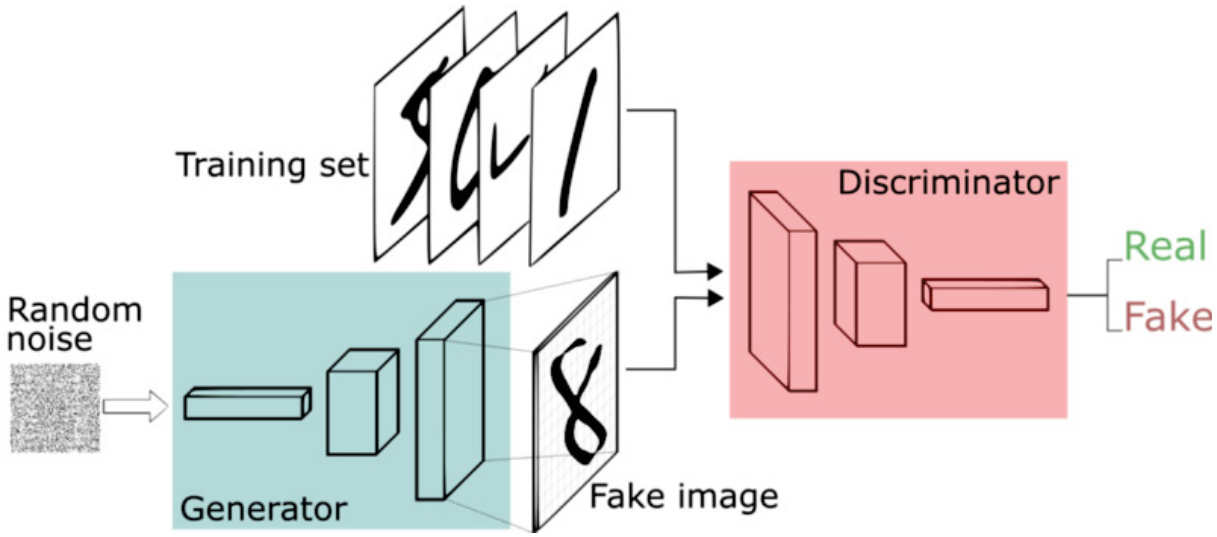


Figure 2.23: Structure of a basic GAN. The GAN is composed by two neural networks called generator and discriminator. The better the images produced by the generator, the more difficult it is for the discriminator to distinguish fake data from real. So, the generator's objective is to trick the discriminator, and the objective of the discriminator is to reveal the generator fails. Source: (Silva, 2020).

input and generates a feature map representing that image. However, in the reverse mode, the CNN receives a feature map with random values and generates a fake image. The discriminator network is also a CNN trained to differentiate between real and fake data generated by the generator. As the generator produces better-quality images, distinguishing between fake and real data becomes more challenging for the discriminator. Therefore, the objective of the generator is to deceive the discriminator, while the objective of the discriminator is to expose the generator's inaccuracies (Karras et al., 2020; Choi et al., 2020). The following Subsections details the two GANs used in this work: StyleGANv2 (Karras et al., 2020) and StarGAN (Choi et al., 2020).

2.7.1 StyleGAN

The StyleGAN (Karras et al., 2019) architecture is composed by a multi-stage generator network capable of generating images at different levels of abstraction and a mapping network that converts a random vector into a latent space, which is then used by the generator network to produce images. The StyleGAN uses the Adaptive Instance Normalization (AdaIN) for style transfer, which allows transferring style information from a reference image to the generated image while maintaining control over other aspects of the image synthesis process.

The StyleGAN employs a strategy of style transfer (Huang and Belongie, 2017) to generate images, and introduces a new generator model, illustrated in Figure 2.24. The conventional generator (presented in the left side) takes a latent space vector as input and produces images from it. However, the updated generator architecture of the StyleGAN (presented on the right side) inputs the latent space vector z in a mapping network f, generating an intermediate latent space W, which is used to control the generator through AdaIN at each convolution layer. Additionally, Gaussian noise is applied after each convolution and before the non-linearity function. The "A" symbol represents a learned affine transform, and "B" applies learned per-channel scaling factors to the noise input.

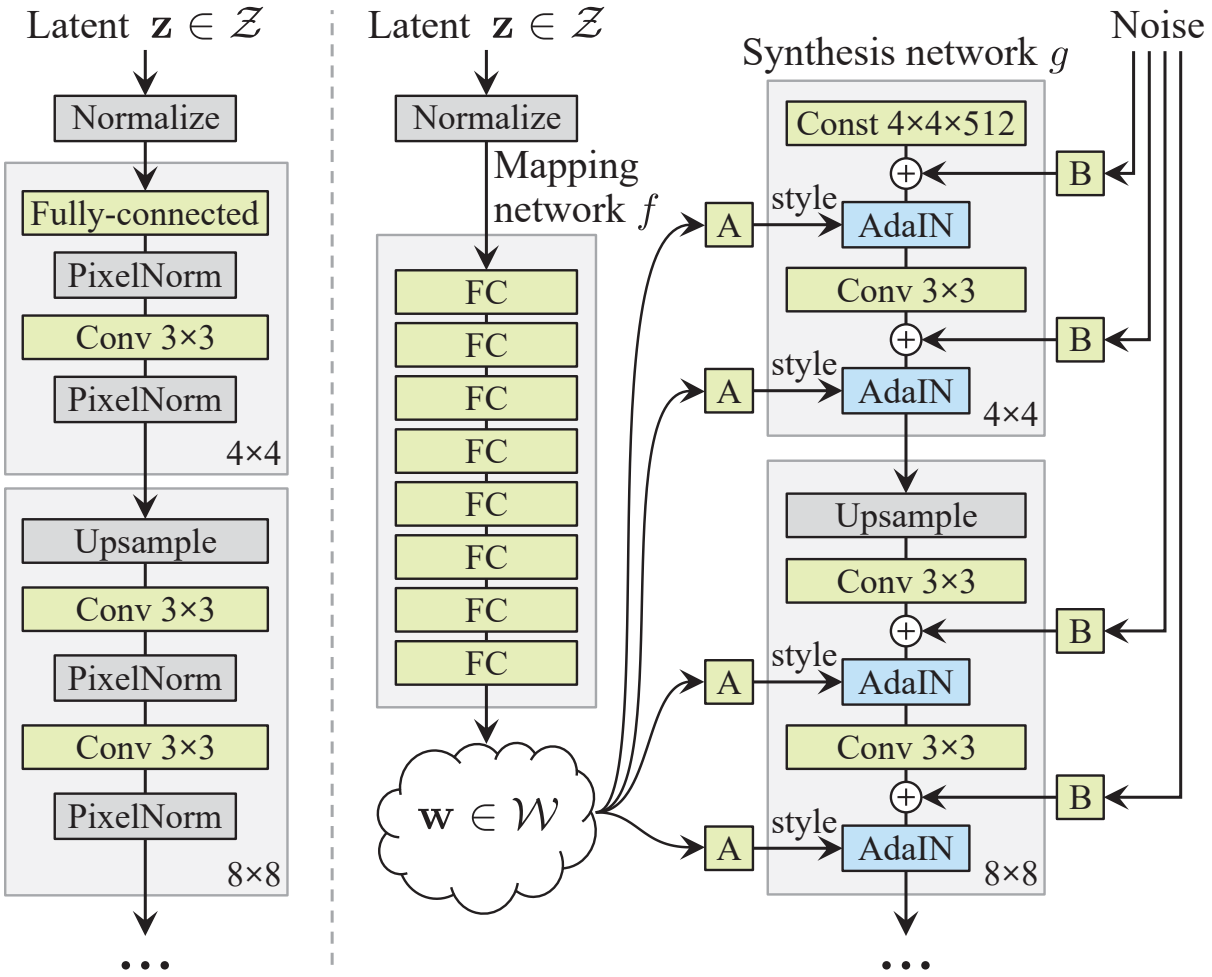


Figure 2.24: The conventional generator (presented in the left side) takes a latent space vector as input and produces images from it. However, the updated generator architecture of the StyleGAN (presented on the right side) inputs the latent space vector z in a mapping network f, generating an intermediate latent space W, which is used to control the generator through AdaIN at each convolution layer. Additionally, Gaussian noise is applied after each convolution and before the nonlinearity function. The "A" symbol represents a learned affine transform, and "B" applies learned per-channel scaling factors to the noise input. Source: (Karras et al., 2019).

2.7.2 StyleGANv2

The StyleGANv2 (Karras et al., 2020) is an evolution of the StyleGAN architecture and it is able to generate high-resolution images up to 1024×1024 pixels, in contrast with the limit of 512×512 of the original StyleGAN. The StyleGANv2 architecture uses a progressive growing technique that starts with a low resolution and gradually increases it as the model learns.

The StyleGAN synthesis network architecture is revamped, as presented in Figure 2.25. The first architecture presents the StyleGAN generator, where the letter "A" denotes a learned affine transform and "B" applies learned per-channel scaling factors to the noise input. The second architecture details the first one by showing the normalization process followed by modulation, which operates on the mean and standard deviation per feature map. The third and fourth architecture details the StarGANv2 generator, with the instance normalization replaced by a "demodulation" operation, applied to the weights associated with each convolution layer.

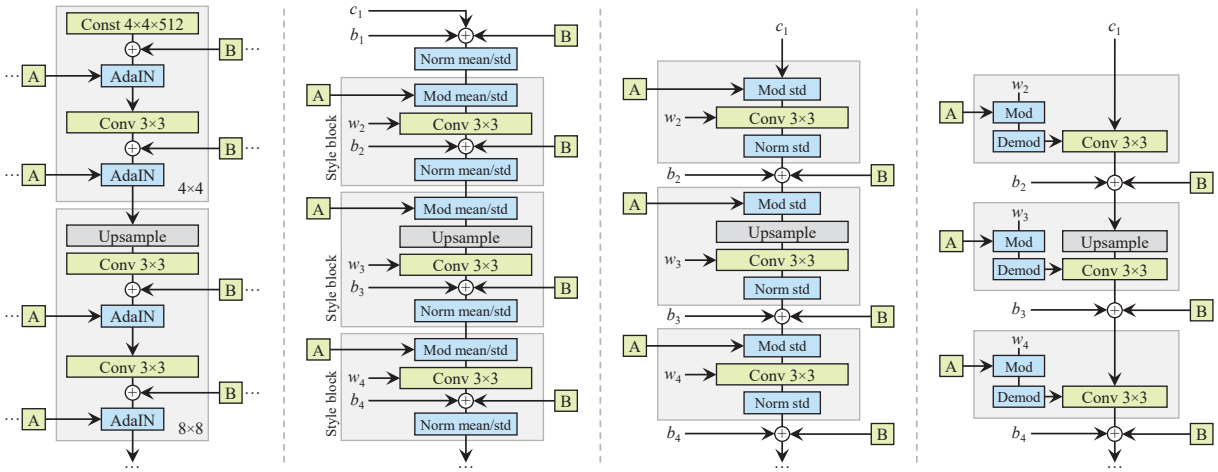


Figure 2.25: The first architecture presents the StyleGAN generator, where the letter "A" denotes a learned affine transform and "*B*" applies learned per-channel scaling factors to the noise input. The second architecture details the first one by showing the normalization process followed by modulation, which operates on the mean and standard deviation per feature map. The third and fourth architecture details the StarGANv2 generator, with the instance normalization replaced by a "demodulation" operation, applied to the weights associated with each convolution layer. Source: (Karras et al., 2020).

2.7.3 StarGAN

The StarGAN (Choi et al., 2018) is a generative architecture designed for image-to-image translation tasks. The image-to-image problem translation aims to learn a mapping that enables converting an image from domain A to an image from domain B (Isola et al., 2017). The main advantage of the StarGAN is that it can perform multiple tasks using a single pair generator and discriminator networks, unlike other image-to-image translation methods trained for a specific task. The core idea behind StarGAN is to learn a mapping between multiple domains. This is achieved by a single generator network that takes an input image and a target domain label as input and outputs an image in the specified domain. Then, a discriminator network is trained to distinguish between generated and real images. Additionally, it employs a novel technique called "adversarial loss with multiple discriminators," which enhances the discrimination ability of the network and improves the quality of the generated images.

The StarGAN architecture is presented in Figure 2.26. It comprises two main components: a discriminator network, denoted by D, and a generator network, denoted by G. Firstly, the discriminator network (presented in "a") is trained to classify real images according to their corresponding domain and distinguish between real and fake images. Then, the generator network (presented in "b") takes the input image and the target domain label as input and generates a fake image. The generator network is also trained to reconstruct the original image from the fake image (presented in "c"). Finally, the generator network is optimized to produce images that can be classified as the target domain by the discriminator network (presented in "d").

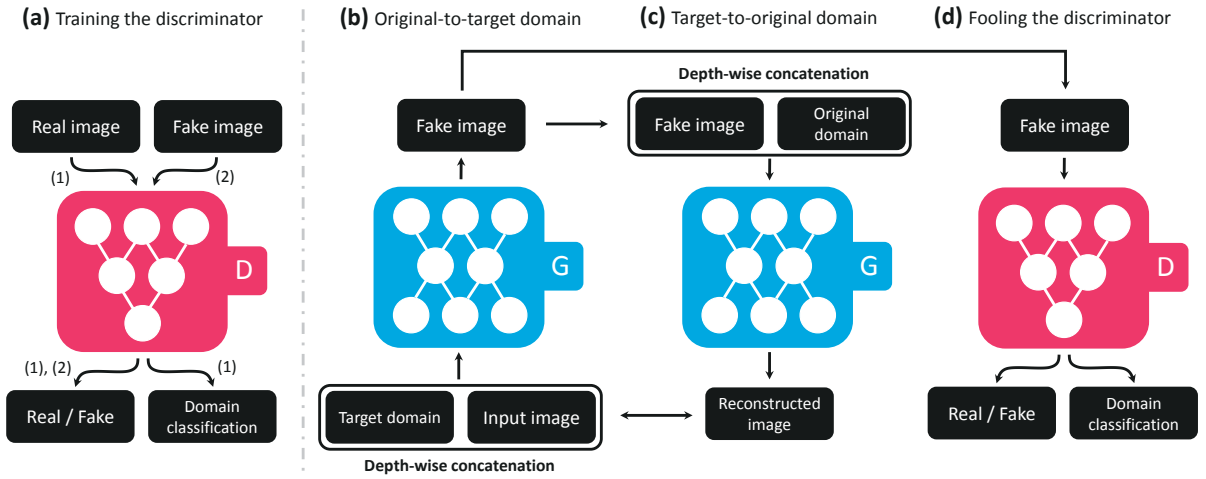


Figure 2.26: The StarGAN comprises two main components: a discriminator network, denoted by D, and a generator network, denoted by G. Firstly, the discriminator network (presented in "a") is trained to classify real images according to their corresponding domain and distinguish between real and fake images. Then, the generator network (presented in "b") takes the input image and the target domain label as input and generates a fake image. The generator network is also trained to reconstruct the original image from the fake image (presented in "c"). Finally, the generator network is optimized to produce images that can be classified as the target domain by the discriminator network (presented in "d"). Source: (Choi et al., 2018).

2.7.4 StarGANv2

The StarGANv2 (Choi et al., 2020) is an improved version of the original StarGAN architecture. One of the main differences between StarGANv2 and the original StarGAN is the introduction of an attention mechanism that allows the generator to focus selectively on specific regions of the input image. This attention mechanism has a spatial-semantic alignment module consisting of an attention mask and a feature descriptor that helps the generator selectively attend to certain regions of the input image based on the target domain. Furthermore, in contrast to the original StarGAN, which used a U-Net generator architecture, the StarGANv2 generator uses a progressive growing architecture with skip connections, similar to the StyleGAN architecture, allowing more flexibility in generating high-quality images. The StarGANv2 also introduces a new technique called "adaptive discriminator augmentation", which helps alleviate mode collapse, a common issue in GANs. This technique involves augmenting the discriminator network with randomly selected image transformations during training, encouraging the generator to produce diverse images.

The StarGANv2 is projected with four networks: the Mapping network, the Style encoder, the Generator, and the Discriminator. The Mapping network takes a latent code randomly sampled from the standard Gaussian distribution, the target domain, and outputs a style code, which maps the latent code to the target domain. The Style encoder takes the input image, its corresponding domain and outputs a style code that maps the input image into its corresponding domain. The Generator takes the input image, the style codes from the Mapping network, and the Style encoder and generates the output image in the target domain. Finally, the Discriminator evaluates the image to distinguish if it is real. Figure 2.27 presents the four networks of the StarGANv2.

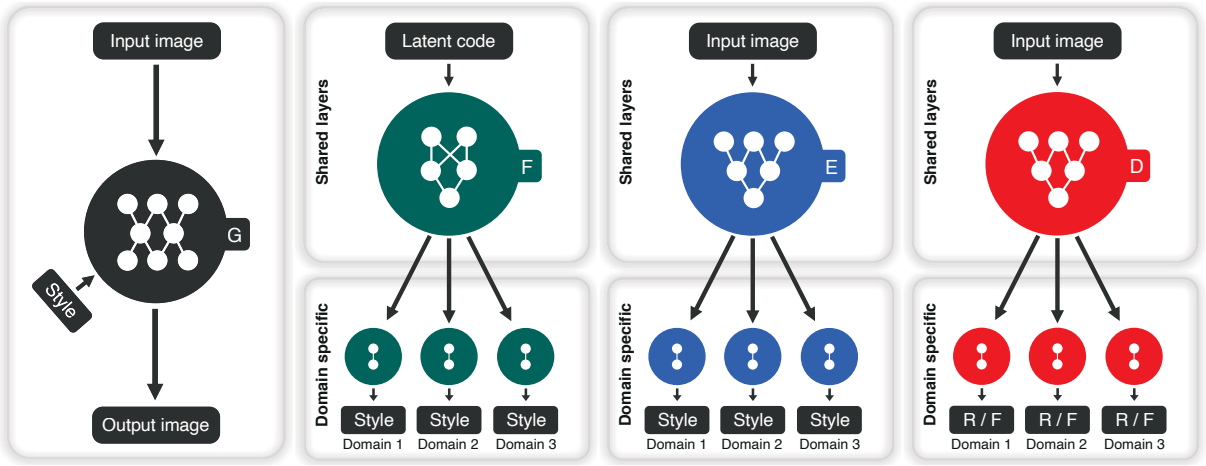


Figure 2.27: The four networks that composed the StarGANv2. The Mapping network, presented in green, generates the style code for the target domain. The Style encoder, presented in blue, generates the style code for the input domain. The Generator, presented in black, used the style codes from the Mapping network and Style encoder to translate the input image into another image with a different domain. The Discriminator, presented in red, evaluates the truthfulness of the generated image. Source: (Choi et al., 2020).

2.8 FINAL CONSIDERATIONS

This Chapter presented relevant concepts approached in this work as Deep Learning, CNN, Semantic Segmentation, Encoder-Decoder networks, and GANs. The CNNs reviewed in this Chapter are fundamental for classification and segmentation problems due to their generalization capabilities. Classification networks like VGG, ResNet, Res2Net, ResNeXt, DenseNet, SE-Net, and RegNet are designed to extract rich and hierarchical features from images, improving the accuracy and robustness of predictions. Specifically, architectures like ResNet introduced residual connections that facilitate the training of very deep networks, while DenseNet promotes feature reuse through dense connections. The segmentation networks like U-Net, U-Net++, FPN, PSPNet, LinkNet, and MA-Net are widely used in many segmentation problems, including medical segmentation problems. U-Net and its variants are highly effective due to their encoder-decoder architecture with skip connections that preserve spatial details, which is essential for detailed segmentation. FPN and PSPNet contribute advanced techniques for combining information at multiple scales, enhancing segmentation in varied contexts.

StarGAN and StyleGAN are generative models with advanced architectures and capabilities in producing high-quality and realistic images. StarGAN is particularly notable for its ability to perform multi-domain image-to-image translation and introduced an innovative approach using a unified framework for diverse image translation tasks, thereby reducing the need for separate models for each domain. StyleGAN is also renowned for its ability to generate high-resolution and photorealistic images. It has introduced style-based generator architecture, allowing precise control over the image generation process and enabling detailed manipulation of features such as facial expressions, textures, and lighting conditions. The next Chapter presents a literature review of segmentation studies and data augmentation techniques proposed for the COVID-19 CT segmentation problem.

3 RELATED WORKS

The basis for the development of this work is drawn from a compilation of pertinent studies, which are succinctly summarized in this chapter. Section 3.1 presents studies proposed for the segmentation of COVID-19 Computed Tomographys (CTs), and Section 3.2 presents studies with generic data augmentation techniques proposed for learning-based problems, and Section 3.3 discuss studies about data augmentation techniques for COVID-19 CT segmentation. Section 3.4 is the conclusion of this chapter.

3.1 COVID-19 CT-SCAN SEGMENTATION

The study proposed by (Saood and Hatem, 2021), conducted a comparative analysis between the U-net (Ronneberger et al., 2015) and SegNet (Badrinarayanan et al., 2017) architectures, presented on Figure 3.1, for the COVID-19 CT-Scan segmentation problem. According to the results obtained from the comparison, the SegNet outperformed the U-net in binary segmentation, whereas the U-net outperformed the SegNet in multi-class segmentation.

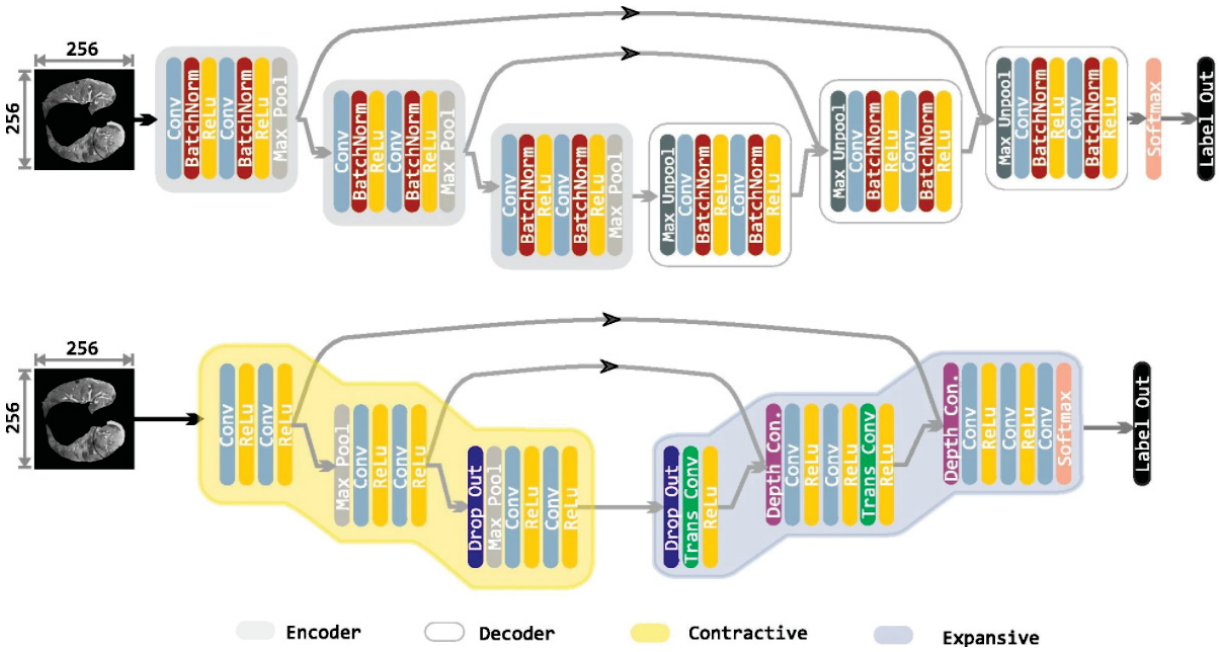


Figure 3.1: The top architecture is the SegNet with encoder blocks in grey and decoder blocks in white, and the bottom architecture is the U-net with encoder side in yellow and decoder side in blue. Source: (Saood and Hatem, 2021).

In their study, (Amyar et al., 2020) proposed a Multi-task Learning (MTL) architecture that combines classification, segmentation, and reconstruction tasks. The proposed architecture, illustrated in Figure 3.2, follows an encoder-decoder structure, with one encoder shared across three decoders. The first decoder is responsible for image reconstruction from the latent space, the second decoder generates a segmentation mask, and the third decoder produces a classification result with three classes: COVID, normal, and others. For the segmentation and reconstruction tasks, a U-net architecture is employed. At the same time, a sequence of fully connected layers is attached at the end of the architecture for the classification task. According to (Zhang and

Yang, 2022), incorporating multiple tasks into a single architecture can enhance the model's generalization ability.

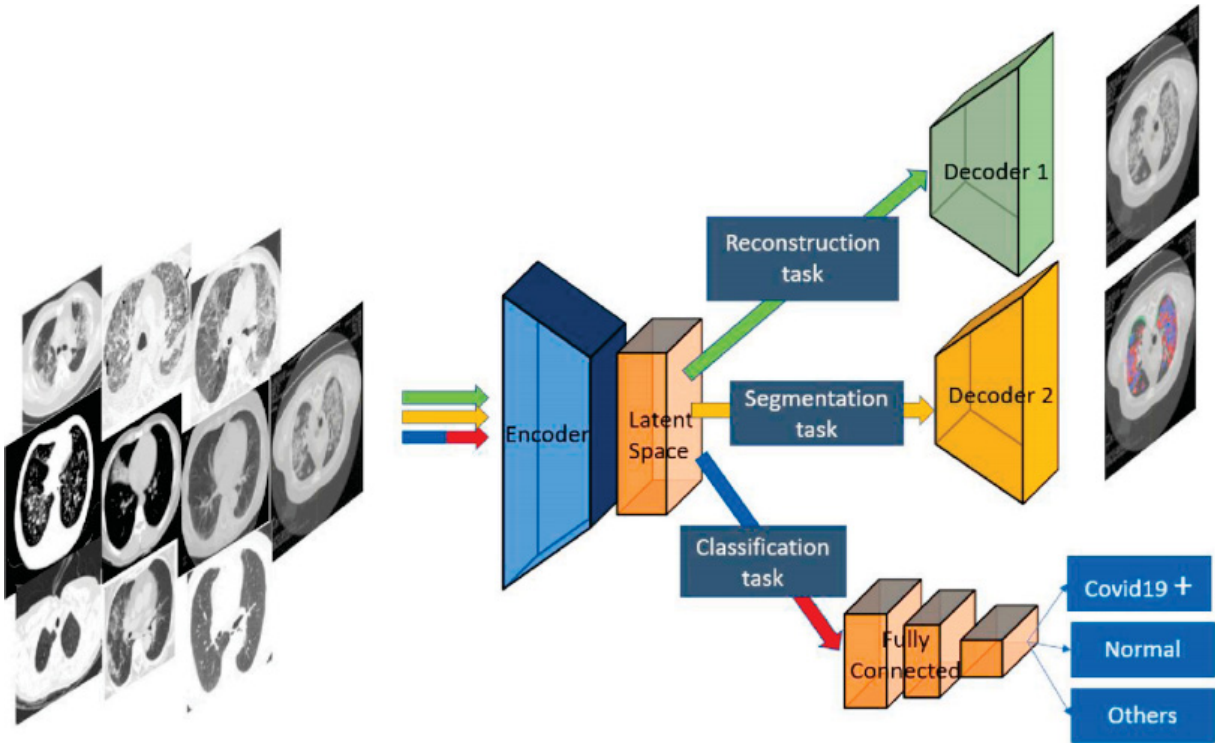


Figure 3.2: The proposed architecture follows an encoder-decoder structure, with three decoders sharing one encoder. The first decoder reconstructs the image from latent space, the second decoder generates a segmentation mask, and the third decoder generates a classification with three classes: COVID, normal, and others. Source: (Amyar et al., 2020).

In (Zhao et al., 2021), the authors modified the U-net model by substituting the convolution layers on the decoder side with dilated convolutions. Additionally, they proposed a dual attention module connected to the skip connections within the decoder module. The dual attention module comprises two components, namely the Gate Attention Module (GAM) and Decoder Attention Module (DAM), which refine the features extracted by the encoder and reduce the semantic gap by combining high and low-level feature maps. Furthermore, a module known as the Residual Attention Block (RAB) is attached to each decoder side block. The RAB incorporates a hybrid dilated convolution module and a DAM to refine post-upsample features. Figure 3.3 provides an overview of the proposed model.

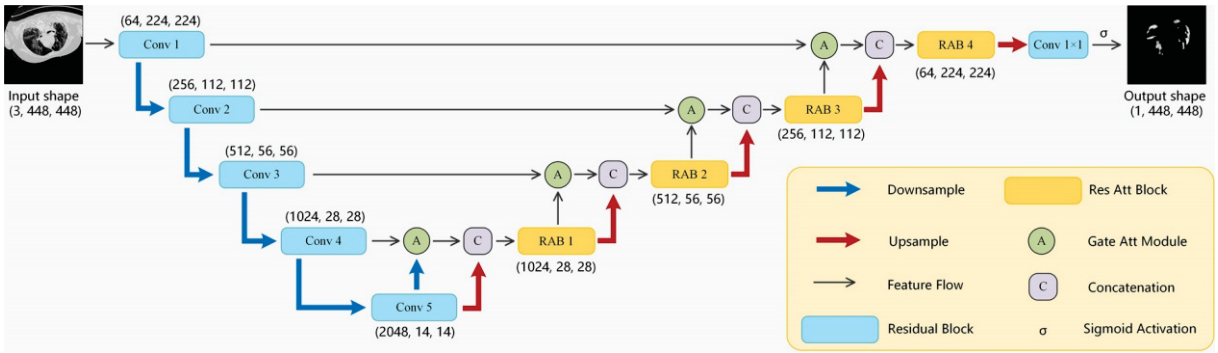


Figure 3.3: A module called RAB is attached to each block of the decoder side. The decoder has four GAMs and four RABs. Each RAB has a hybrid dilated convolution module and a DAM. Source: (Zhao et al., 2021).

A U-net-based architecture with attention modules for segmenting COVID-19 lesions is proposed in (Zhou et al., 2020). Each block in the encoder and decoder comprises a Res_dil block that follows a series of convolutional layers. The Res_dil blocks incorporate dilated convolutions, which aid the network in learning more detailed regions of the image. Each skip connection in the U-net, which connects a convolution block to its corresponding deconvolution block, incorporates two attention mechanisms constructed with dilated convolutions. The first attention mechanism is connected at the beginning of the skip connection, while the other is connected at the end. Figure 3.4 illustrates the proposed network, where the Res_dil blocks are depicted in yellow, and the attention mechanisms are presented in a light and dark grey rectangles.

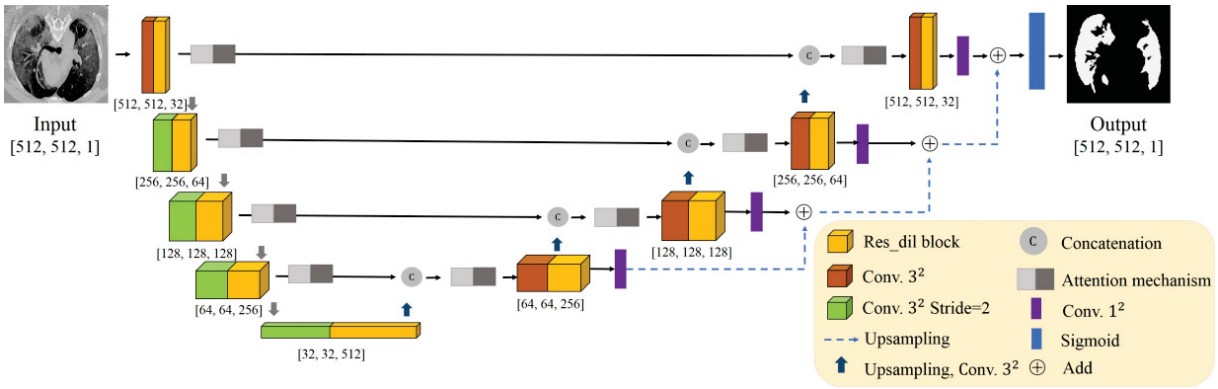


Figure 3.4: The Res_dil blocks, represented in yellow, use dilated convolutions to help the network learn more detailed image regions. The attention mechanisms are presented in light and dark grey rectangles. Source: (Zhou et al., 2020).

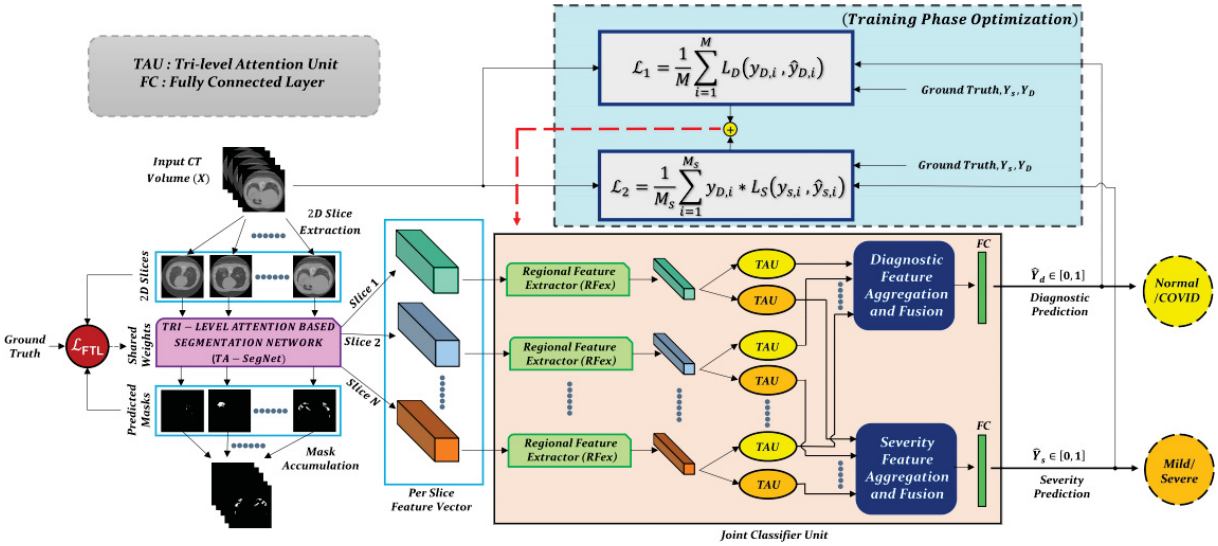


Figure 3.5: The Tri-level Attention-based Segmentation Network (TA-SegNet) is presented on the left side of the image. The output of the TA-SegNet inputs the Joint Classification Unit on the right side of the image. The Joint Classification Unit outputs the diagnostic and severity predictions. Source: (Mahmud et al., 2021a).

The work presented in (Mahmud et al., 2021a) proposes a complex framework based on attention to segment COVID-19 CT-Volumes, as shown in Figure 3.5. The first stage of the framework introduces a network named TA-SegNet. Based on SegNet (Badrinarayanan et al., 2017), this network performs image segmentation and extracts diagnostic feature vectors from COVID CT-Volumes. The diagnostic feature vectors generated by TA-SegNet are fed into the Joint Classification Unit. A proposed Tri-level Attention Unite (TAU) is employed to enhance

the diagnostic features and make diagnostic and severity predictions. In positive cases, the Joint Classification Unit outputs whether the image shows COVID and its severity level.

In (Qiblawey et al., 2021), a pipeline comprising two segmentation models is proposed. The first model takes the entire dataset's image as input and performs segmentation of lung regions, generating a new image containing only the lung regions. The second model uses this image as input to segment the COVID lesions, and the severity of the lesions is evaluated at the end of the pipeline. Several segmentation models are evaluated in the proposed pipeline. Figure 3.6 illustrates the proposed pipeline, where the first segmentation model, called Lung Segmentation, generates an image containing only lung regions, and the second model, called COVID-19 Pneumonia Segmentation, receives this image and produces a segmentation mask for COVID infections.

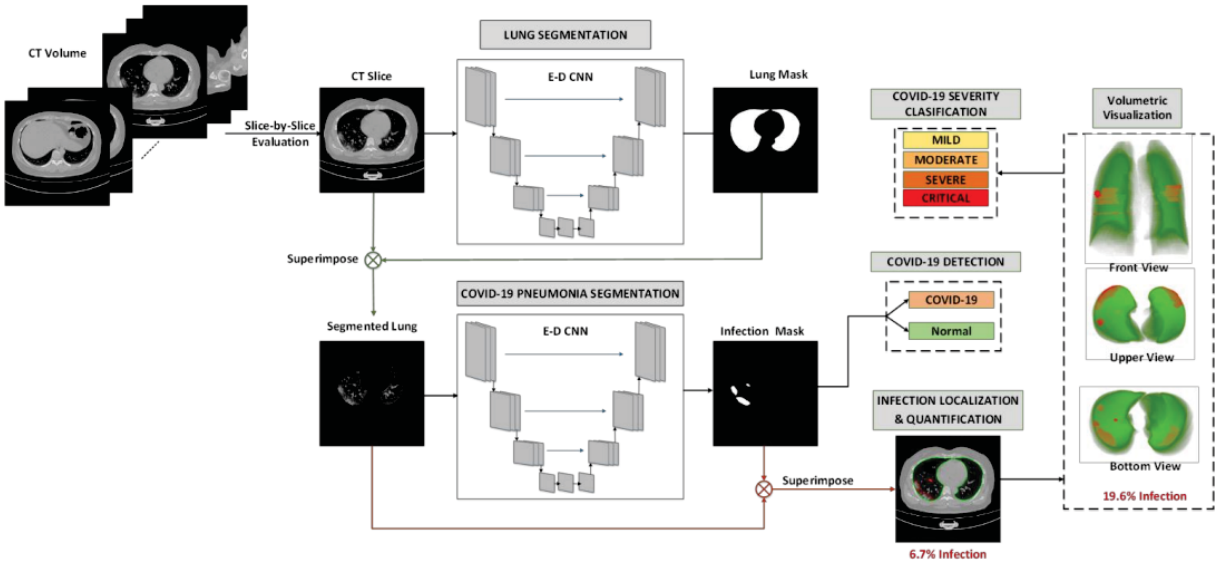


Figure 3.6: The first segmentation model, called Lung Segmentation, generates an image only with lung regions. The second segmentation model, called COVID-19 pneumonia segmentation, receives the image only with the lung regions and generates a segmentation mask with COVID infections. Source: (Qiblawey et al., 2021).

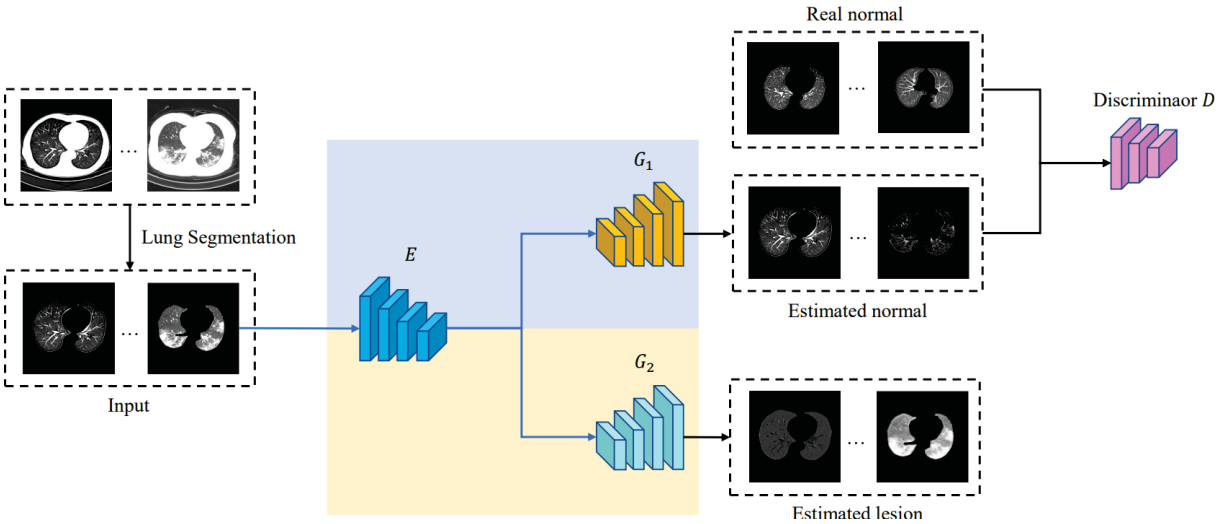


Figure 3.7: The encoder E is represented with dark blue, and the decoders G1 and G2 are presented in yellow and light blue. The discriminator D is presented in pink. Source: (Yang et al., 2021).

The authors of (Yang et al., 2021) introduced a weakly-supervised COVID lesion localization and segmentation framework. The framework includes an encoder and two decoders, namely G1 and G2. The G1 decoder generates an image without lesions, followed by a discriminator to determine if the image is genuine. The G2 decoder is a segmentation network used to estimate the lesion region. The framework aims to divide the image into two images, one with COVID lesions and the other without, to improve the detection process. Figure 3.7 presents an overview of the proposed network, where the encoder E is shown in dark blue, and the decoders G1 and G2 are in yellow and light blue. The discriminator D is presented in pink. In contrast,(Müller et al., 2021) proposed a 3D-U-net to perform COVID-19 lesion segmentation on 3D volumes. As shown in Figure3.8, the architecture takes 3D patches as input and outputs the segmentation of COVID-19 3D volumes.

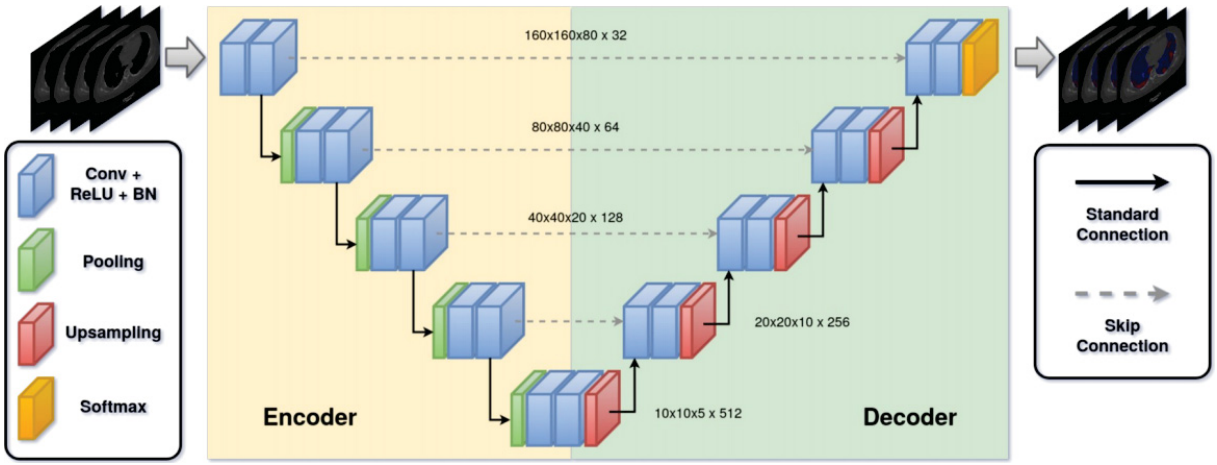


Figure 3.8: The architecture receives 3D patches as input and outputs the segmentation of COVID-19 3D-Volumes. Source: (Müller et al., 2021).

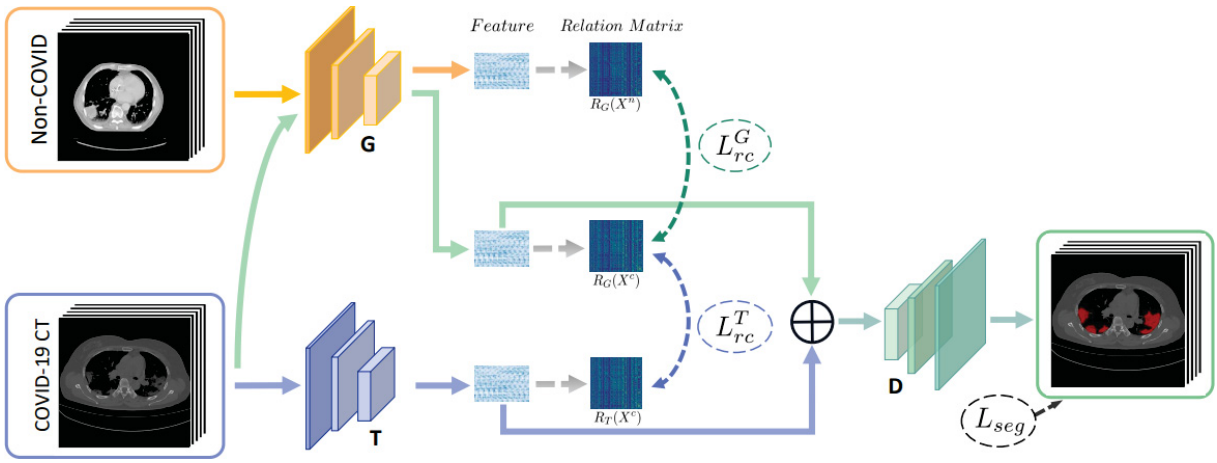


Figure 3.9: The proposed framework has two encoders. The first one in yellow, called G, extracts features from general examples, and the second one in blue, called T, extracts features from target examples, i.e., COVID. The features extracted from both encoders are concatenated to input the decoder D, represented in green, to generate the segmentation of COVID CT-Scans. Source: (Zhang et al., 2021a).

In their study, (Zhang et al., 2021a) investigated the combination of feature sets extracted from CT-Scans with COVID lesions and those with lesions from non-COVID cases. The proposed framework employs two encoders, as illustrated in Figure 3.9: an encoder (G) that extracts features from general examples and an encoder (T) that extracts features from target

examples, specifically COVID cases. The features extracted from both encoders are merged and inputted into the decoder (D), which generates the segmentation of COVID CT-Scans.

(Mahmud et al., 2021b) introduced a network named CovSegNet, which follows a two-phase training pipeline. Firstly, a 2D version of CovSegNet, CovSegNet2D, is trained to segment 2D images of COVID lesions. Secondly, the pre-trained CovSegNet2D is applied to remove redundant regions from a 3D volume. Given that CovSegNet2D is designed for 2D images, the 3D volume is divided into 2D slices and merged at the end of CovSegNet2D. In this phase, the output of CovSegNet2D is a Region of Interest (ROI)-enhanced segmentation volume, which serves as the input for a 3D version of CovSegNet called CovSegNet3D. The CovSegNet3D generates the final segmentation of the 3D volume, considering intra-slice and inter-slice contextual features. The proposed pipeline is illustrated in Figure 3.10, where the phase one represents the training of CovSegNet2D for 2D segmentation, and the phase two represents the utilization of CovSegNet2D to generate an ROI-enhanced volume, which is then inputted to CovSegNet3D for the final 3D segmentation.

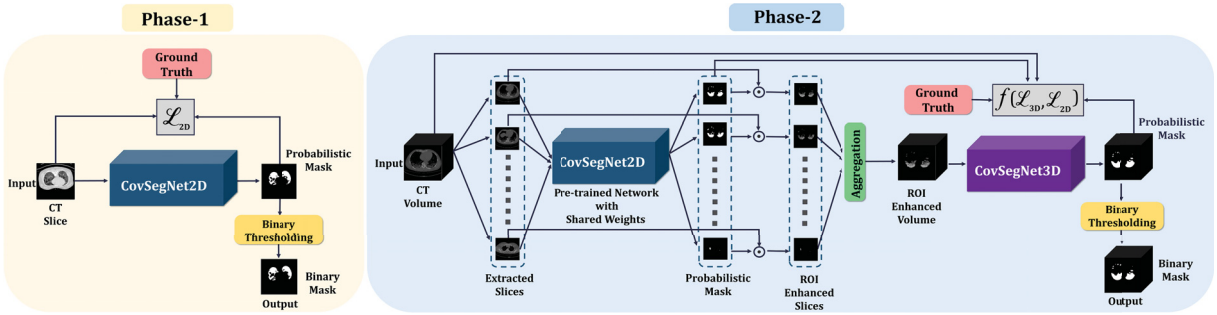


Figure 3.10: In Phase-1, represented in yellow, the CovSegNet2D is trained to perform segmentation of 2D slices and in Phase-2, represented in blue, the CovSegNet2D is used to generate the ROI-enhanced volume used as input of the CovSegNet3D. Source: (Mahmud et al., 2021b).

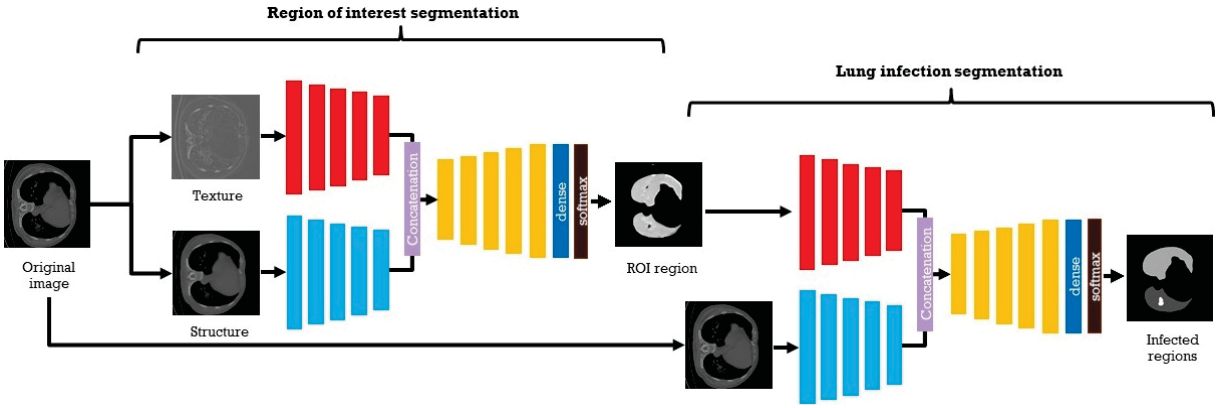


Figure 3.11: The input image is divided into texture and structure images and fed into the first network to generate the ROI region image. The ROI region image and the original image are then fed into the second network, which produces the final segmentation of lesion regions. Source: (Elharrouss et al., 2021).

(Elharrouss et al., 2021) employed two encoder-decoder networks for COVID-19 CT segmentation. The first network is designed to segment only the ROI with the lung, while the second network is responsible for segmenting the lesions inside the lung region. Both networks consist of two encoders that are concatenated before the decoder. The input image is divided into texture and structure images and fed into the first network to generate the ROI region image. The ROI region image and the original image are then fed into the second network, which

produces the final segmentation of lesion regions. The flowchart of the proposed networks is illustrated in Figure 3.11.

(Das et al., 2022) enhanced a U-Net architecture by incorporating two modules: a Attention-modulated Multi-Scalar (MS)-block and an Attention Gate (AG). The MS-block is added to the network's encoder and decoder steps and merges the features of dilated convolutions using four distinct dilation rates to extract multi-scale features. The Attention Gate (AG) module integrates the upsampled images with the encoded features to boost the importance of lesion regions in the feature maps. The proposed architecture, called Ensembled Attention-based Multi-scaled Convolution Network (EAMC), is illustrated in Figure 3.12.

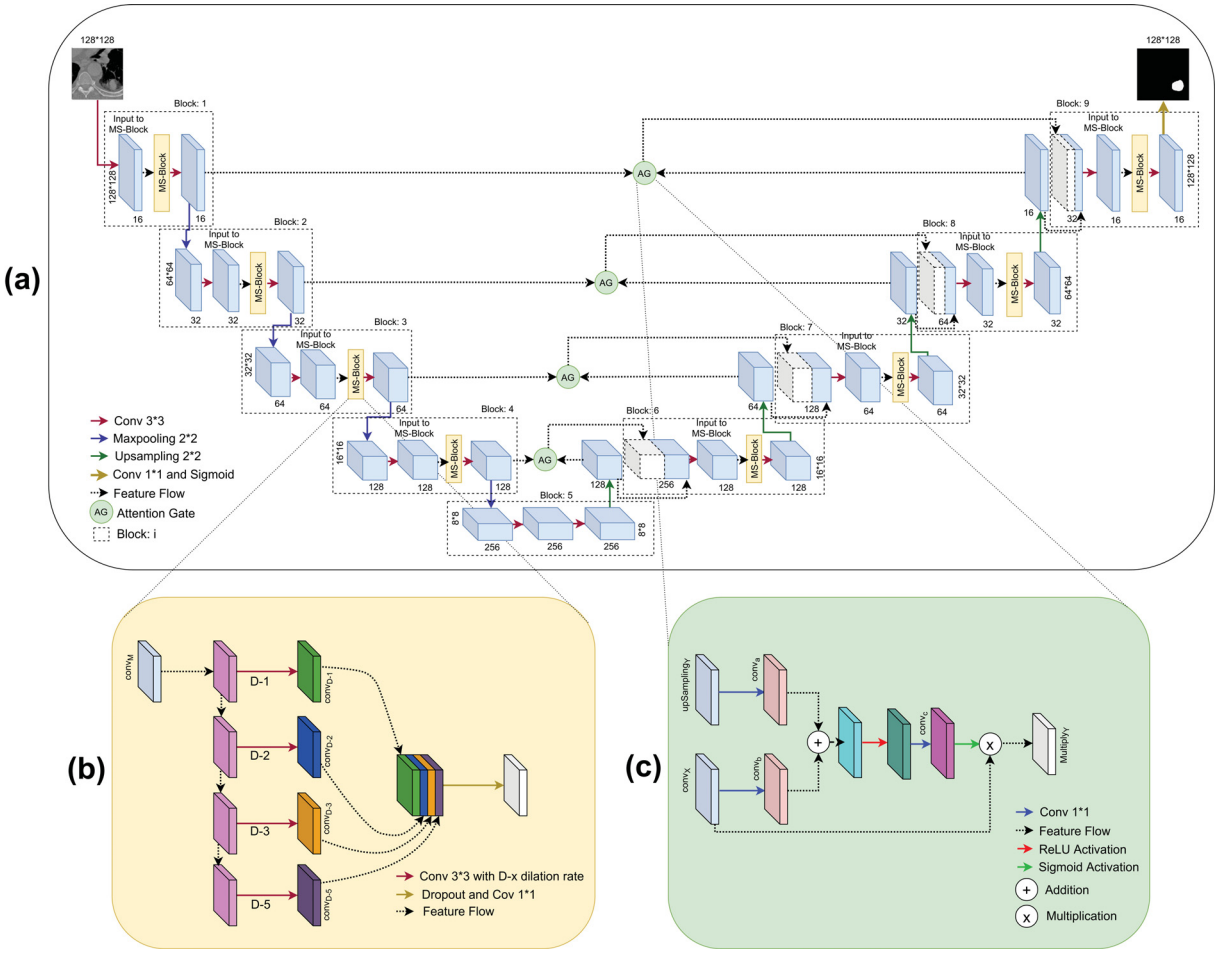


Figure 3.12: The MS-block is added to the network's encoder and decoder steps and merges the features of dilated convolutions using four distinct dilation rates to extract multi-scale features. The Attention Gate (AG) module integrates the upsampled images with the encoded features to boost the importance of lesion regions in the feature maps. The Attention-modulated Multi-Scalar (MS)-block is depicted in "b" and the AG module is shown in "c". The complete network is presented in "a". Source: (Das et al., 2022).

In their work, (Xu et al., 2020) proposed a Weakly-supervised Framework for COVID-19 Lesion Segmentation called GASNet, which incorporates a Generative Adversarial Network (GAN) training process into a segmentation network. The proposed framework, illustrated in Figure 3.13, comprises a generator, a segmenter, and a discriminator. The generator (in green) generates healthy images without COVID-19 lesions. The segmenter (in blue) generates a segmentation mask based on the input image. The output of the generator and the segmenter are combined to create a healthy synthetic image. The discriminator (in orange) then receives a real

healthy image and a synthetic one and tries to determine which is real. This training process improves the segmentation by allowing the segmenter and generator to deceive the discriminator.

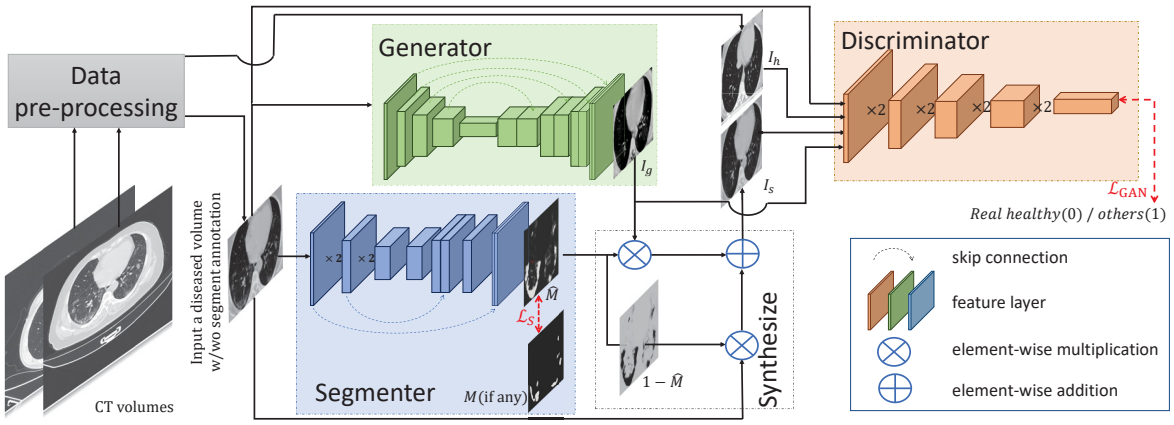


Figure 3.13: The GASNet is composed by a generator, a segmenter, and a discriminator. The segmenter (in blue) is responsible for taking the input image and generating a segmentation mask. The output of the generator and the segmenter's output are fused to generate a healthy synthetic image. Then, the discriminator (in orange) will receive a real healthy and the synthetic one and try to find out which one is real. Source: (Xu et al., 2020).

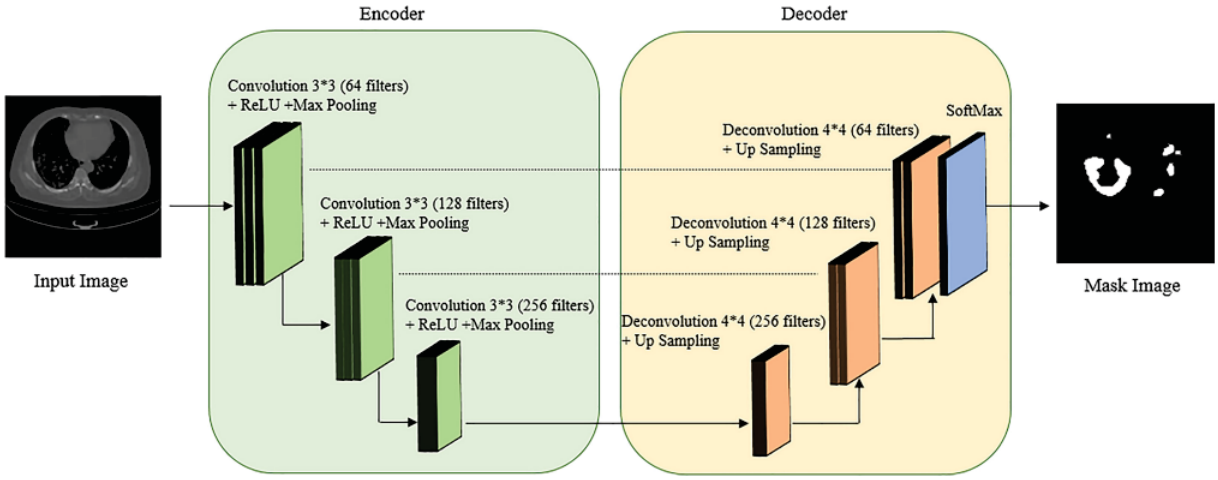


Figure 3.14: The proposed approach involves a three-stage encoder, where each stage consists of a single convolutional layer, followed by a Rectified Linear Units (ReLu) activation function and a max-pooling layer. The decoder also comprises three stages, each incorporating deconvolutional and upsampling layers. A softmax layer is appended to the decoder's end to produce the ultimate segmentation mask. Source: (Khalifa et al., 2021).

The authors of (Khalifa et al., 2021) put forward a U-Net-style architecture to address the COVID CT segmentation problem. The proposed approach involves a three-stage encoder, where each stage consists of a single convolutional layer, followed by a ReLu activation function and a max-pooling layer. The decoder also comprises three stages, each incorporating a deconvolutional and upsampling layers. A softmax layer is appended to the decoder's end to produce the ultimate segmentation mask. Figure 3.14 illustrates the proposed network architecture.

In (Chen et al., 2020c), the convolutional blocks of the U-net were replaced by ResNeXt blocks, and an attention mechanism was proposed to capture complex features from the feature maps. Furthermore, the output of each block in the decoder receives the concatenation of the output of the encoder block with the predecessor decoder block. The concatenation of the encoder and decoder outputs passes through the attention mechanism before being fed into the following decoder block. The evaluated model is presented in Figure 3.15.

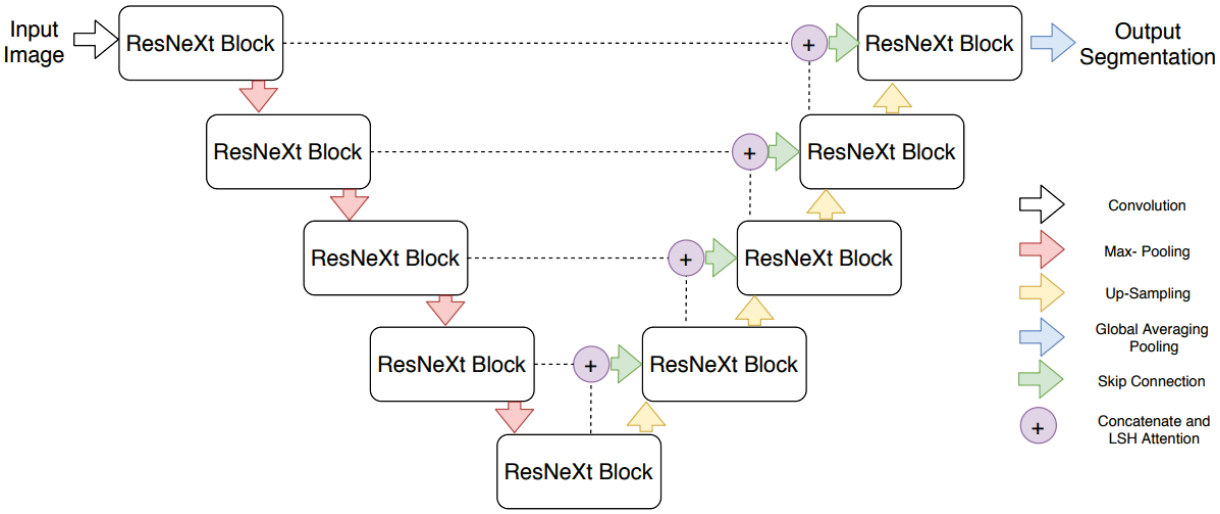


Figure 3.15: A U-Net architecture where all blocks are ResNeXts blocks. Furthermore, the output of each block in the decoder receives the concatenation of the output of the encoder block with the predecessor decoder block. The concatenation of the encoder and decoder outputs passes through the attention mechanism before being fed into the following decoder block. Source: (Chen et al., 2020c).

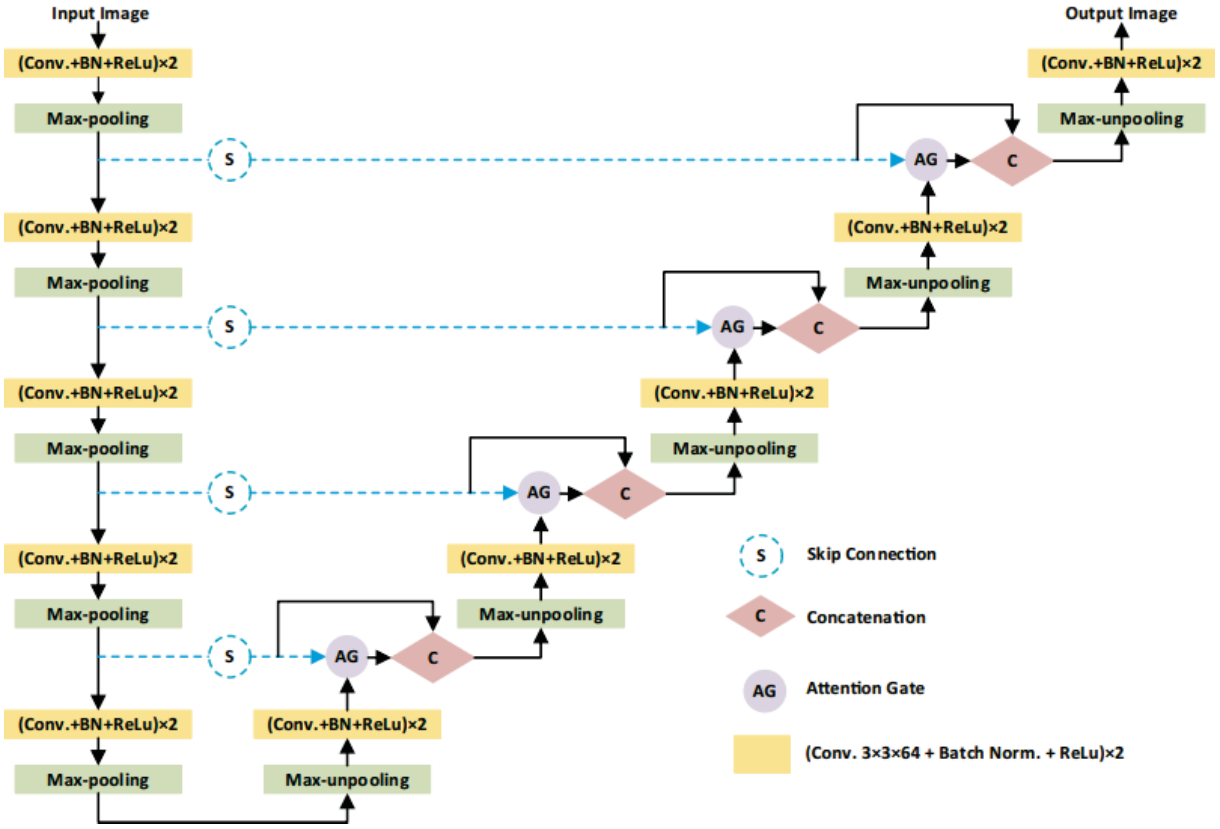
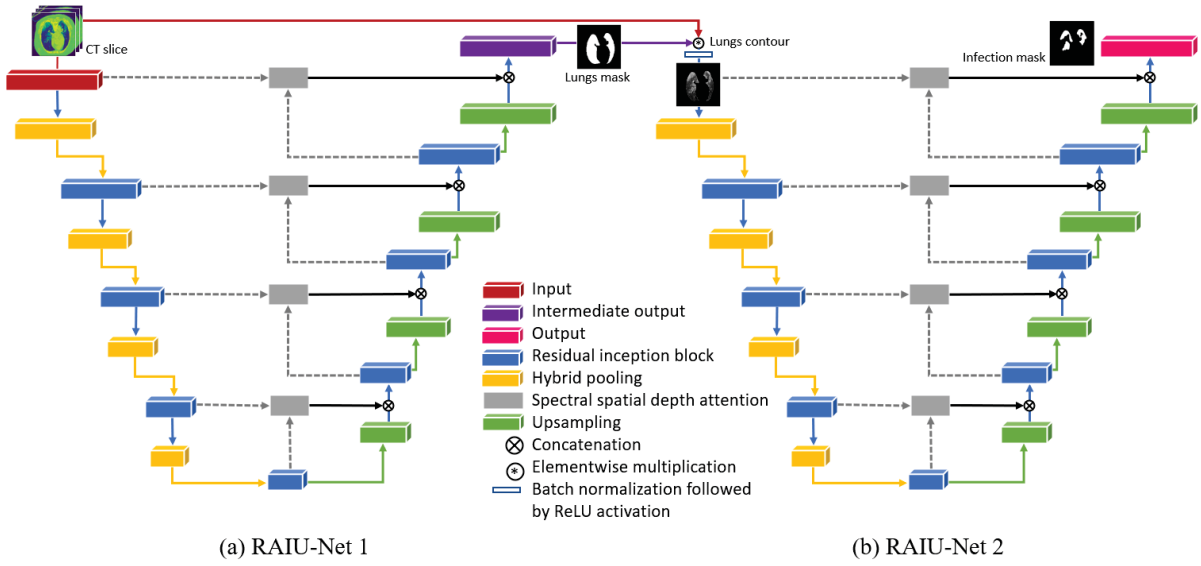


Figure 3.16: Each block of the encoder and the decoder are composed by a convolutional layer (the yellow rectangles) and Max-pooling layers (the green rectangles). Each AG mechanism receives the output of a decoder block and the output of an encoder block through a skip connection (in blue), and then the output of the AG mechanism is concatenated with the output of the encoder block past through the skip connection. Source: (Ümit Budak et al., 2021).

In their publication, (Ümit Budak et al., 2021) introduced a modified SegNet architecture that utilizes AG to enhance segmentation accuracy. Specifically, AG mechanisms are integrated into the skip connections of the SegNet, highlighting the salient features in these connections and

increasing their resolution, which improves the segmentation results. The proposed architecture, illustrated in Figure 3.16, comprises an encoder and decoder steps for feature extraction and mask generation. Each encoder and decoder block includes a convolutional layer (depicted as yellow rectangles) and Max-pooling layers (shown as green rectangles). The AG mechanism takes the output of a decoder block and an encoder block through a skip connection (denoted in blue). Then it concatenates the output of the AG mechanism with the output of the encoder block passed through the skip connection. This concatenated output serves as input for the subsequent decoder block.

The segmentation network proposed in (Punn and Agarwal, 2022) is named Covid-19 Hierarchical Segmentation Network (CHS-Net). The CHS-Net consists of two Residual Attention Inception U-Nets (RAIU-Nets) that are cascaded in sequence. An RAIU-Net is a segmentation network based on the U-net architecture that uses a spectral-spatial and depth attention network in the skip connections to improve the segmentation of the different feature maps resolutions. The first RAIU-Net is employed for lung segmentation, producing an image containing only the region inside the lungs. This image is fed into the second RAIU-Net to segment COVID-19 lesions. The architecture of the CHS-Net with the two RAIU-Nets is shown in Figure 3.17.



(a) RAIU-Net 1 (b) RAIU-Net 2

Figure 3.17: The first RAIU-Net performs lung segmentation, generating an image only with the region inside the lung. This image inputs the second RAIU-Net to perform the segmentation of COVID lesions. Source: (Punn and Agarwal, 2022).

(Asad et al., 2023) proposed the Adaptive Multiscale Online Likelihood Network (MONet), which utilizes a U-Net architecture during the training phase to produce a segmentation image with a corresponding probability of lesion and background regions. Subsequently, a human specialist provides feedback on the initial segmentation result by selecting foreground and background seeds and rejecting incorrect segmentation results. The likelihood network uses 3D convolutions with varying kernel sizes to extract multi-scale features from the generated feature maps. These are then concatenated to input fully connected layers responsible for producing the final segmentation image. The user can iteratively refine the segmentation result by adding or removing seeds, and the likelihood network is dynamically updated in real-time to incorporate the user's feedback. Figure 3.18 presents the proposed MONet.

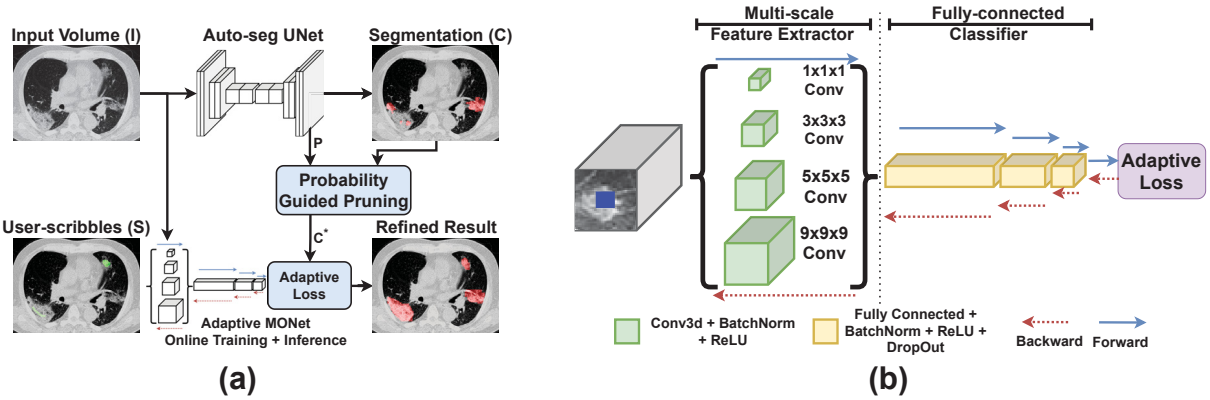


Figure 3.18: The MONet utilizes a U-Net architecture during the training phase to produce a segmentation image with a corresponding probability of lesion and background regions. Subsequently, a human specialist provides feedback on the initial segmentation result by selecting foreground and background seeds and rejecting incorrect segmentation results. The likelihood network uses 3D convolutions with varying kernel sizes to extract multi-scale features from the generated feature maps. These are then concatenated to input fully connected layers responsible for producing the final segmentation image. Source: (Asad et al., 2023).

(Hu et al., 2022) made modifications to an encoder-decoder network by incorporating three modules: Edge Supervised Module (ESM), Auxiliary Semantic Supervised Module (ASSM), and Attention Fusion Module (AFM). The ESM is added to the first convolutional layers of the encoder to accentuate edge information and object boundaries. Then, the ASSM is applied to the deeper layers in the encoder to extract semantic information from the feature maps in these layers. Finally, the AFM fuses multi-scale feature maps of different levels in the decoder to produce the final segmentation map. The objective of the AFM is to integrate high-level and low-level features to improve the network's capacity to segment multi-scale objects. Figure 3.19 depicts the proposed network architecture.

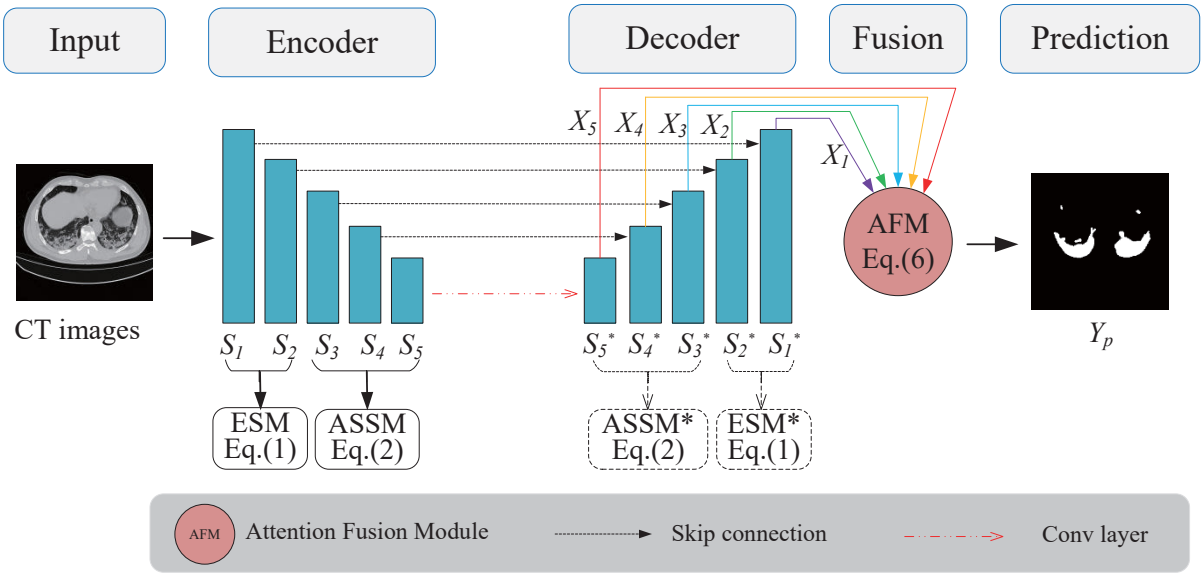


Figure 3.19: The ESM is added to the first convolutional layers of the encoder to accentuate edge information and object boundaries. Then, the ASSM is applied to the deeper layers in the encoder to extract semantic information from the feature maps in these layers. Finally, the AFM fuses multi-scale feature maps of different levels in the decoder to produce the final segmentation map. Source: (Hu et al., 2022).

The authors in (Zhang et al., 2021b) also presented a U-Net architecture with an attention module between the encoder and decoder steps. Additionally, they proposed a modification in the decoder by employing super-pixel blocks to restore the resolution of the feature maps and generate the output segmentation. The encoder in their proposed model uses Residual Network (ResNet) blocks to extract features from the input images. The modifications made to the U-Net architecture are illustrated in Figure 3.20.

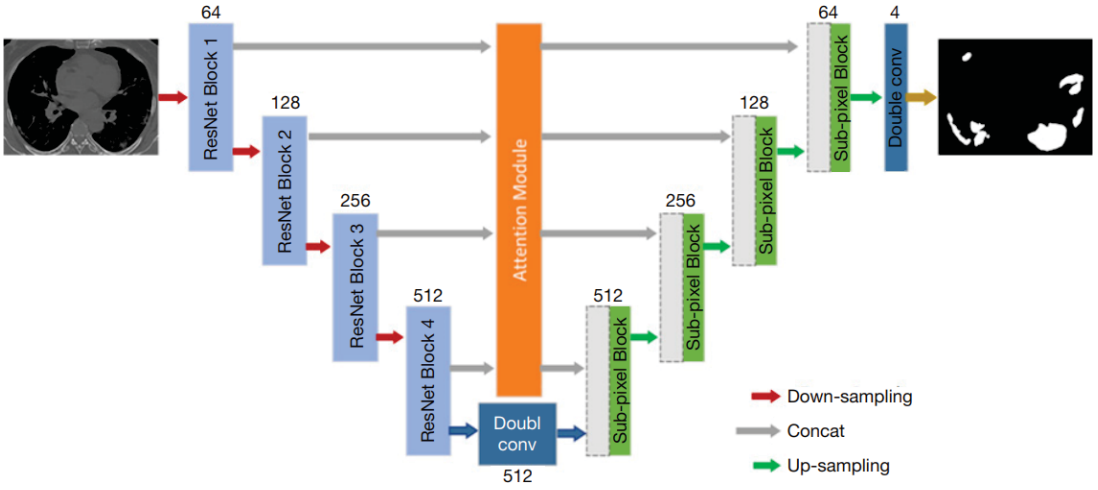


Figure 3.20: The proposed U-Net architecture has an attention module between the encoder and decoder steps. Additionally, the decoder has super-pixel blocks to restore the resolution of the feature maps and generate the output segmentation. The encoder in the proposed model uses ResNet blocks to extract features from the input images. Source: (Zhang et al., 2021b).

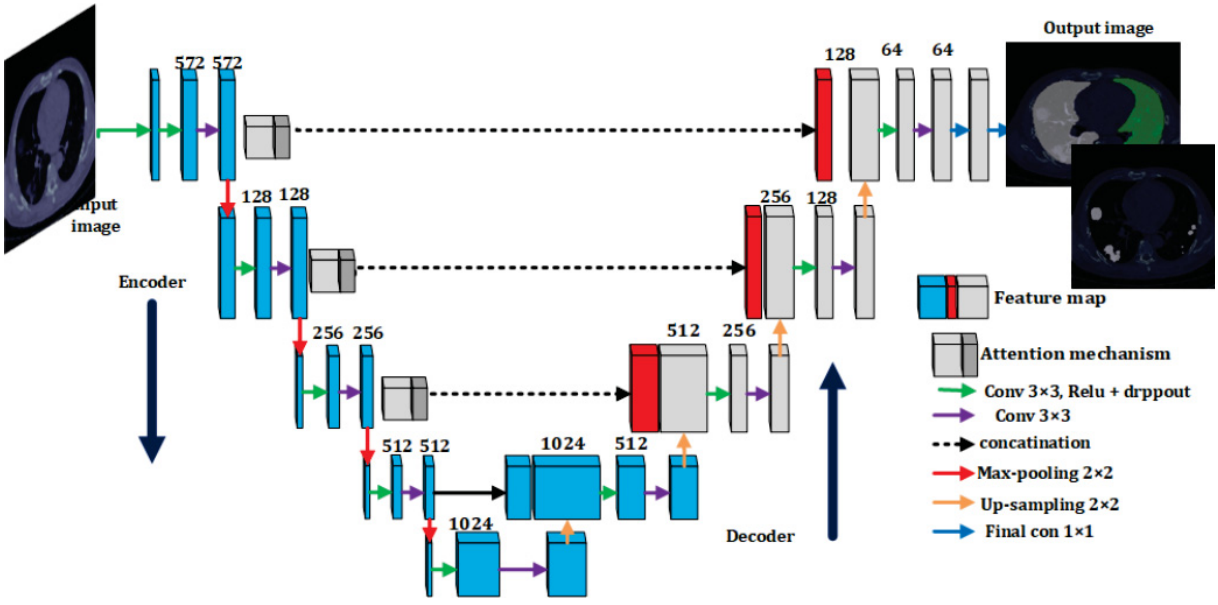


Figure 3.21: The proposed network incorporates attention mechanisms to enhance the learning of representations from the infected regions. These attention mechanisms are attached at the end of each convolutional block in the encoder, and their output is concatenated with the input of each deconvolutional block in the decoder. Source: (Ahmed et al., 2022).

The COVID-19 CT segmentation problem was addressed by (Ahmed et al., 2022) using a U-Net model. The proposed network incorporates attention mechanisms (Oktay et al., 2018) to enhance the learning of representations from the infected regions. These attention mechanisms

are attached at the end of each convolutional block in the encoder, and their output is concatenated with the input of each deconvolutional block in the decoder. Figure 3.21 provides a detailed view of the proposed network.

The network proposed in (Zhang et al., 2022) replaces the conventional convolutional and deconvolutional blocks with the Multiscale Feature Capture Block (MSFCB), which utilizes dilated convolutions to gather context information from various scales. The initial three encoder blocks capture low-level features, while the following two capture high-level features. Additionally, a Multiscale Feature Fusion (MSFF) module is incorporated between the encoder and decoder steps to combine multilevel features extracted from the preceding three MSFCB blocks.

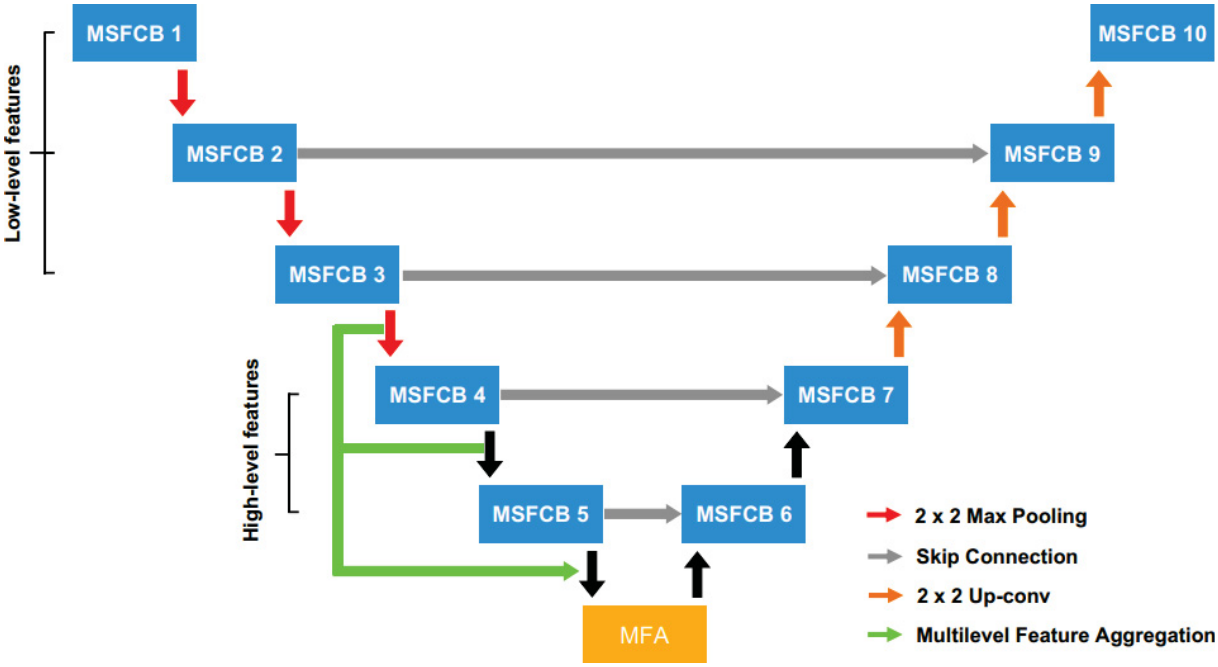


Figure 3.22: The proposed network replaces the conventional convolutional and deconvolutional blocks with the MSFCB, which utilizes dilated convolutions to gather context information from various scales. The initial three encoder blocks capture low-level features, while the following two capture high-level features. Additionally, a MSFF module is incorporated between the encoder and decoder steps to combine multilevel features extracted from the preceding three MSFCB blocks. Source: (Zhang et al., 2022).

In (Joseph Raj et al., 2021), the authors proposed the integration of an Improved Dilation Convolution (IDC) module between the encoder and decoder to enhance the receptive field and gather more precise edge information, which assisting with characteristic extraction. Additionally, the authors utilized an AG module in every skip connection between the encoder and decoder mirror blocks to minimize spatial information loss in the feature mapping toward the end of the encoder. Figure 3.23 represents the proposed architecture. The grey dashed square between the last encoder block and the first decoder block is the IDC module, and the triangle in each skip connection is the AG module.

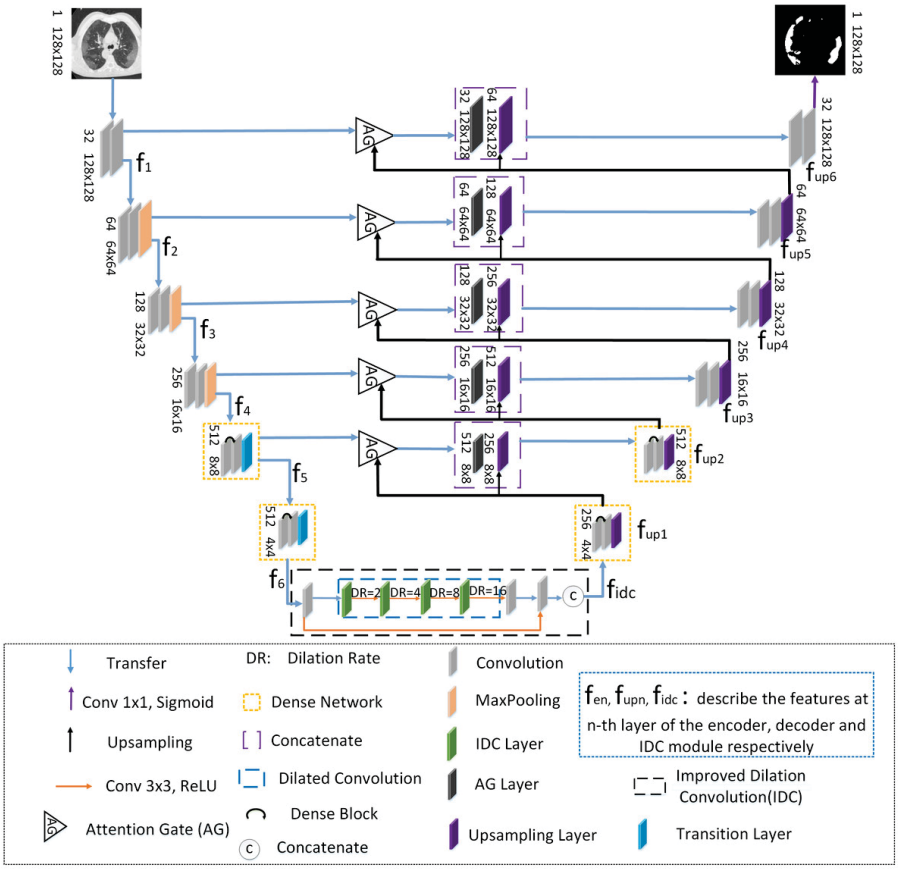


Figure 3.23: The grey dashed square between the last encoder block and the first decoder block is the IDC module, and the triangle in each skip connection is the AG module. Source: (Joseph Raj et al., 2021).

Table 3.1: List of papers reviewed in this work with methods proposed for COVID-19 CT segmentation.

Paper	Author/Year
Multi-task deep learning based CT imaging analysis for COVID-19 pneumonia: Classification and segmentation	(Amyar et al., 2020)
Residual Attention U-Net for Automated Multi-Class Segmentation of COVID-19 Chest CT Images	(Chen et al., 2020c)
GASNet: Weakly-supervised Framework for COVID-19 Lesion Segmentation	(Xu et al., 2020)
Automatic COVID-19 CT segmentation using U-Net integrated spatial and channel attention mechanism	(Zhou et al., 2020)
COVID-19 lung CT image segmentation using deep learning methods: U-Net versus SegNet	(Saood and Hatem, 2021)
D2A U-Net: Automatic segmentation of COVID-19 CT slices based on dual attention and hybrid dilated convolution	(Zhao et al., 2021)
Detection and Severity Classification of COVID-19 in CT Images Using Deep Learning	(Qiblawey et al., 2021)
Towards Unbiased Covid-19 Lesion Localisation And Segmentation Via Weakly Supervised Learning	(Yang et al., 2021)
Robust chest CT image segmentation of COVID-19 lung infection based on limited data	(Müller et al., 2021)
ADID-UNET—a segmentation model for COVID-19 infection from lung CT scans	(Joseph Raj et al., 2021)
CovTANet: A Hybrid Tri-Level Attention-Based Network for Lesion Segmentation, Diagnosis, and Severity Prediction of COVID-19 Chest CT Scans	(Mahmud et al., 2021a)
CovSegNet: A Multi Encoder–Decoder Architecture for Improved Lesion Segmentation of COVID-19 Chest CT Scans	(Mahmud et al., 2021b)
An Encoder–Decoder-Based Method for Segmentation of COVID-19 Lung Infection in CT Images	(Elharrouss et al., 2021)
A deep learning semantic segmentation architecture for COVID-19 lesion discovery in limited chest CT datasets	(Khalifa et al., 2021)
Efficient COVID-19 Segmentation from CT Slices Exploiting Semantic Segmentation with Integrated Attention Mechanism	(Ümit Budak et al., 2021)
Exploiting Shared Knowledge From Non-COVID Lesions for Annotation-Efficient COVID-19 CT Lung Infection Segmentation	(Zhang et al., 2021a)
Novel coronavirus pneumonia detection and segmentation based on the deep-learning method	(Zhang et al., 2021b)
CHS-Net: A Deep Learning Approach for Hierarchical Segmentation of COVID-19 via CT Images	(Punn and Agarwal, 2022)
Deep co-supervision and attention fusion strategy for automatic COVID-19 lung infection segmentation on CT images	(Hu et al., 2022)
A Sustainable Deep Learning-Based Framework for Automated Segmentation of COVID-19 Infected Regions: Using U-Net with an Attention Mechanism and Boundary Loss Function	(Ahmed et al., 2022)
Collective Intelligent Strategy for Improved Segmentation of COVID-19 from CT	(Das et al., 2022)
Semantic segmentation of COVID-19 lesions with a multiscale dilated convolutional network	(Zhang et al., 2022)
Adaptive Multi-scale Online Likelihood Network for AI-assisted Interactive Segmentation	(Asad et al., 2023)

Table 3.1 presents a list of papers reviewed in this work with methods proposed for COVID-19 CT segmentation.

3.2 DATA AUGMENTATION

Data augmentation is a technique used in deep learning to increase the size and diversity of a training datasets (Ruiz et al., 2019, 2020b,a). It involves applying transformations to the original data, generating new examples similar to the original but with some modifications. These transformations include rotations, translations, scaling, shearing, flipping, and adding noise or distortions (Krinski et al., 2022). It also aims to provide the model with more diverse examples, which can improve its ability to generalize and perform well on unseen data (Shorten and Khoshgoftaar, 2019). Data augmentation is particularly useful when dealing with limited training data or when the data is imbalanced, with some classes having significantly fewer examples than others. By applying transformations to the original data, data augmentation can balance the dataset and prevent the model from overfitting the training data. It can also improve the model's ability to handle variations in the input data, such as changes in lighting or viewpoint, and make it more robust to noise and errors (Shorten and Khoshgoftaar, 2019).

The Random Erasing (Zhong et al., 2020) is an example of data augmentation that adds new information to the image. This approach involves randomly selecting a rectangular area in an image and replacing the pixels in that region with random values or the average pixel value of the entire image. The rectangle's height, width, and center position are chosen randomly. This technique improves the model's robustness to object occlusions and reduce overfitting by introducing new information into the image. Figure 3.24 displays examples of the Random Erasing method applied to different images.

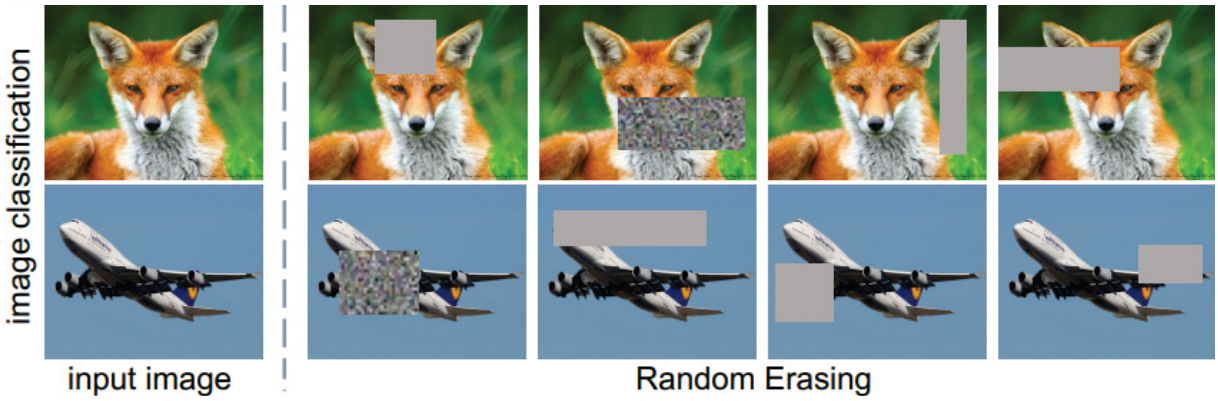


Figure 3.24: Some examples of the Random Erasing data augmentation applied in different images. A rectangle with random values is positioned on top of the image. This rectangle's height, width, and center points are also random values. Source: (Zhong et al., 2020).

The CutMix (Yun et al., 2019) technique combines two images to create a new one and shares a similar concepts with the Random Erasing (Zhong et al., 2020). The idea is to select a rectangular region from one image and replace it with the corresponding region from another image while adjusting the pixel values in the overlapping region to reflect the combined contribution of both images. This process creates a new image with a mix of objects and backgrounds from both images. The size and position of the rectangular region are chosen randomly. The CutMix helps improve the model's generalization ability by encouraging it to learn features from multiple classes simultaneously and reducing overfitting. Figure 3.25 provides examples of this data augmentation method, where Sub-figure 3.25(b) shows an image

containing a dog, and Sub-figure 3.25(a) displays the same image mixed with another image of a cat. Compared to methods that fill a rectangular image area with zero or random noise values, CutMix has the advantage of minimizing information loss, thereby improving training efficiency.

(a) Example of image with a dog.

(b) The same image of the dog but mixed with an image of a cat.

Figure 3.25: CutMix combines two images by adding a selected region of Image A to a selected region of Image B. Source: (Yun et al., 2019).

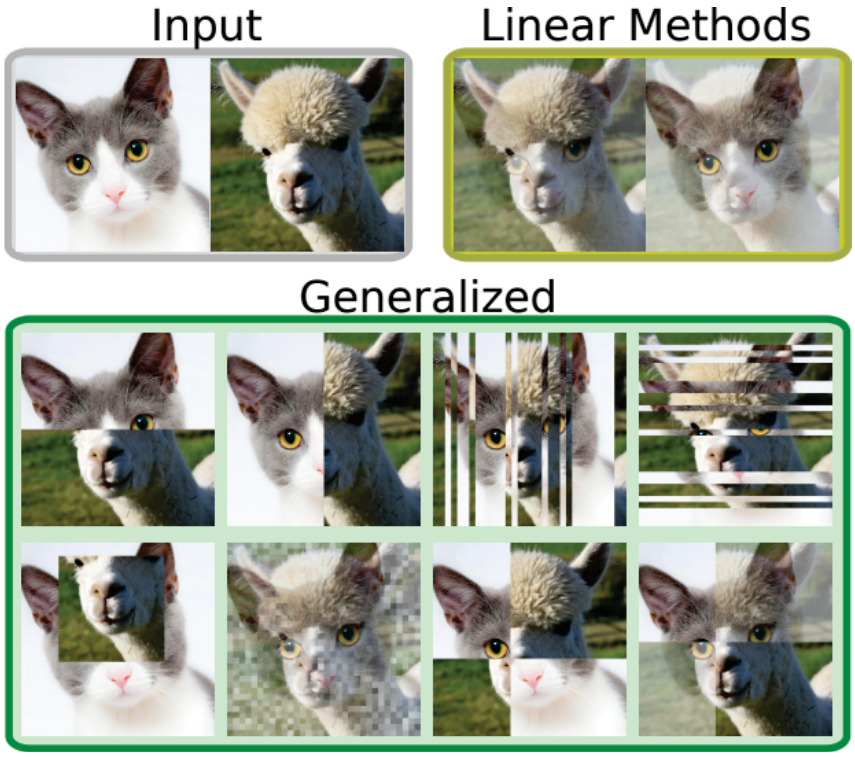


Figure 3.26: Some examples of the different mixing techniques evaluated by (Summers and Dinneen, 2019). First, an example of linear mixing data augmentation is presented, where the input images are added to each other. Then, eight examples of non-linear mixing algorithms evaluated by the authors are presented. Source: (Summers and Dinneen, 2019).

In a similar way of CutMix (Yun et al., 2019), which involves merging two distinct images, the study presented by (Summers and Dinneen, 2019) explored various non-linear mixing algorithms. Figure 3.26 displays some instances of the different mixing techniques analyzed

by the researchers. Initially, Figure 3.26 presents a linear mixing data augmentation technique, which involves adding input images to each other. Subsequently, Figure 3.26 presents eight non-linear mixing algorithms. The authors highlight that non-linear mixing algorithms can be as effective as linear mixing data augmentation techniques.

The AugMix (Hendrycks et al., 2020) is a data augmentation technique combining multiple image processing operations to create many augmented images. It consists of three main steps: augmentation chain creation, mixing of augmented images, and normalization. In the first step, image processing operations, such as rotation, shearing, and color distortion, are randomly selected to create an augmentation chain. The length and strength of the chain are also randomly determined. In the second step, multiple augmented images are generated by applying different augmentation chains to the same input image. These augmented images are mixed using a specific mixing parameter to create a final augmented image. Finally, in the third step, the normalized version of the augmented image is fed into the deep learning model for training. Figure 3.27 presents an example of the AugMix application with three augmentation sequences applied to the image. Then, the three sequences are combined in one image, with each sequence receiving a weight in the element-wise convex combination. In the end, the generated image and the original image are combined to generate the final image.

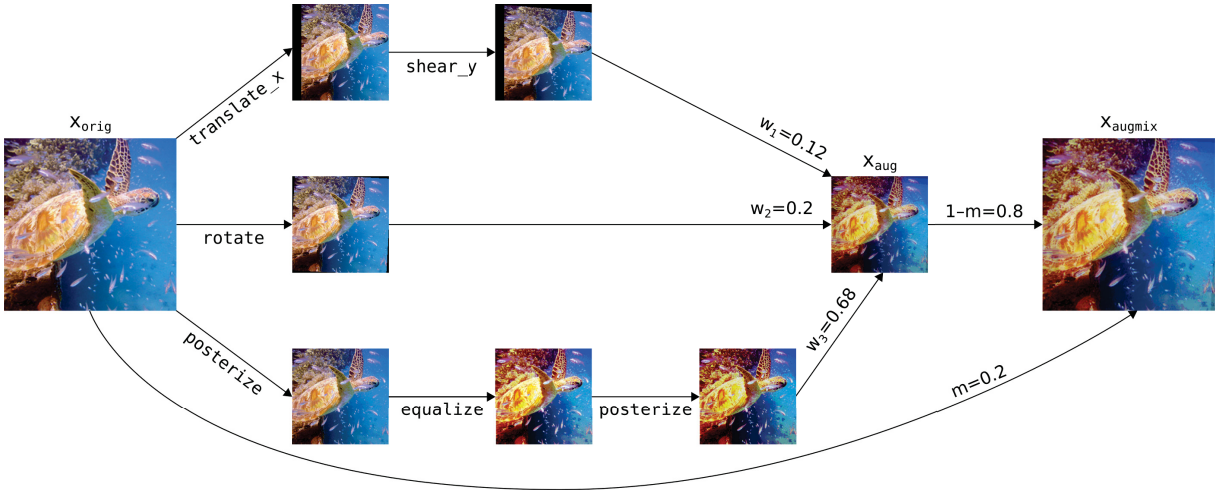


Figure 3.27: Example of the AugMix application with three augmentation sequences applied on the image. Then, the three sequences are combined in one image, with each sequence receiving a weight in the elementwise convex combination. In the end, the generated image and the original image are combined to generate the final image. Source: (Hendrycks et al., 2020).

The ClassMix data augmentation, proposed by (Olsson et al., 2021), is a data augmentation technique similar to CutMix (Yun et al., 2019). However, instead of replacing a portion of an image with a portion of another image, ClassMix replaces the class label of an image with the class label of another randomly selected image. Specifically, in ClassMix, a randomly selected image is chosen from the training dataset. Its class label is replaced with the class label of another randomly selected image in the same mini-batch. The resulting mixed image is then used to train the model. The intuition behind ClassMix is that it encourages the model to learn more robust and discriminative features by forcing it to generalize to images that belong to different classes. This is especially useful in cases where the training dataset is imbalanced or lacks diversity. In this algorithm, a segmentation network is applied to generate the segmentation masks S_A and S_B of the images A and B. Half of the classes from S_a are randomly selected and used to create a new segmentation mask M. Then, M creates a mixed image X_a and a mixed segmentation mask Y_a. The ClassMix pipeline is presented in Figure 3.28.

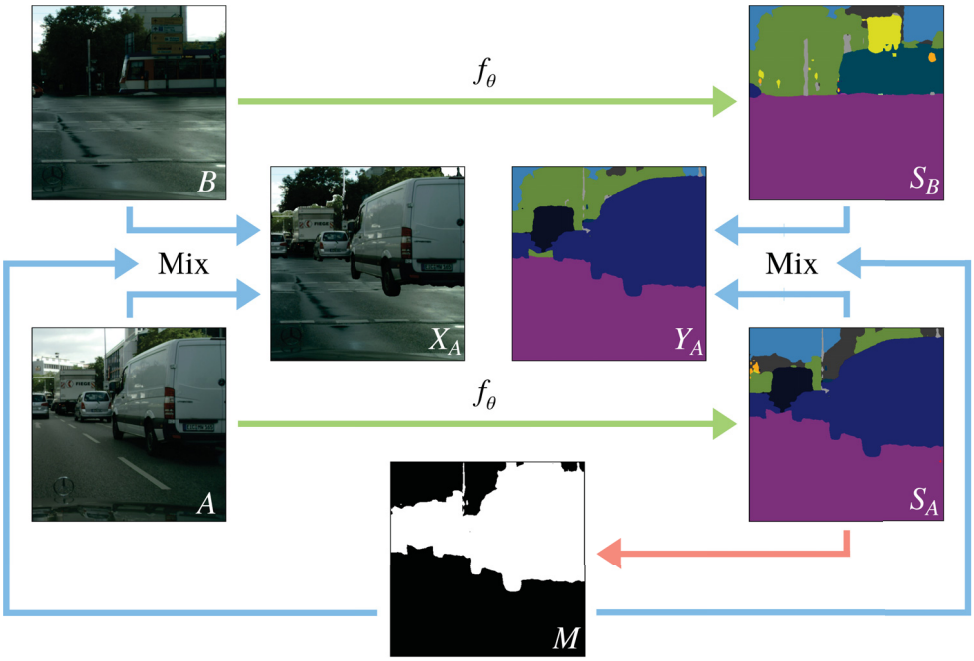


Figure 3.28: In the ClassMix, a segmentation network is applied to generate the segmentation masks S_A and S_B of the images A and B. Half of the classes from S_a are randomly selected and used to create a new segmentation mask M. Then, M creates a mixed image X_a and a mixed segmentation mask Y_a. Source: (Olsson et al., 2021).

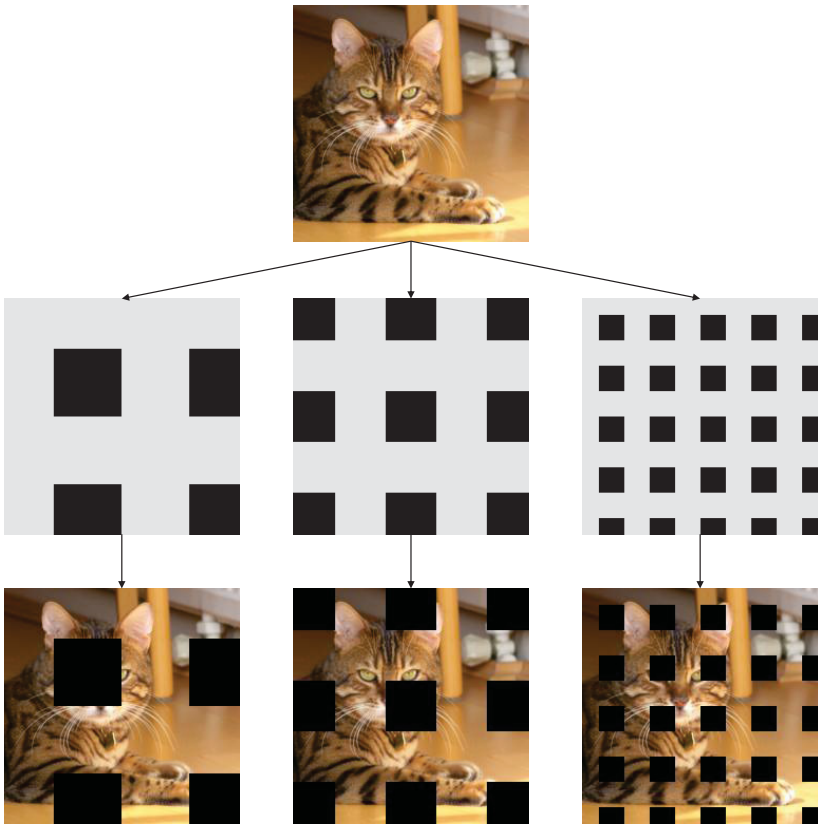


Figure 3.29: The GridMask overlaps a grid of rectangular patches on the input image, and the patches are randomly selected for masking. The grid can be uniform or random, and the size and shape of the patches can be varied. Source: (Chen et al., 2020b).

The GridMask (Chen et al., 2020b) overlaps a grid of rectangular patches on the input image, and the patches are randomly selected for masking. The grid can be uniform or random, and the size and shape of the patches can be varied. The main idea of this data augmentation is the same as Random Erasing proposed by (Zhong et al., 2020), which forces the network to learn from occluded regions in the image and reduces overfitting. This method's advantage over Random Erasing and others that randomly remove regions from the image is that these random algorithms can remove relevant regions from the image. Figure 3.29 presents an example of the Grid Mask application with three different square sizes on top of a cat image.

In (Kisantal et al., 2019), a data augmentation method was proposed to enhance the detection and segmentation of small objects. The authors utilized a "copy and paste" technique to generate multiple copies of the target objects. This approach increased the number of anchor boxes produced by Mask-RCNN (He et al., 2017), facilitating the network to identify and detect small objects. Figure 3.30 illustrates two instances of the proposed augmentation technique. The first image depicts a man playing tennis with several ball copies, and the second image shows three women playing with multiple copies of frisbees.

Figure 3.30: Examples of the data augmentation proposed by (Kisantal et al., 2019). The authors applied a "copy and pasting" strategies to create several copies of the interest objects. The first image presents a man playing tennis with several ball copies, while the second image presents three women playing several copies of frisbees. Source: (Kisantal et al., 2019).

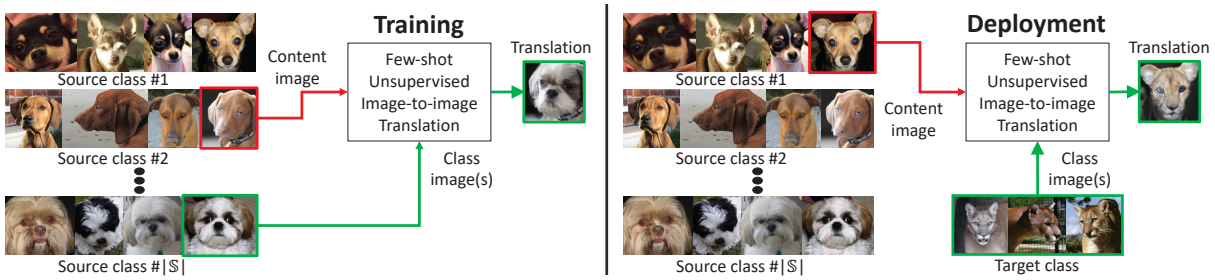
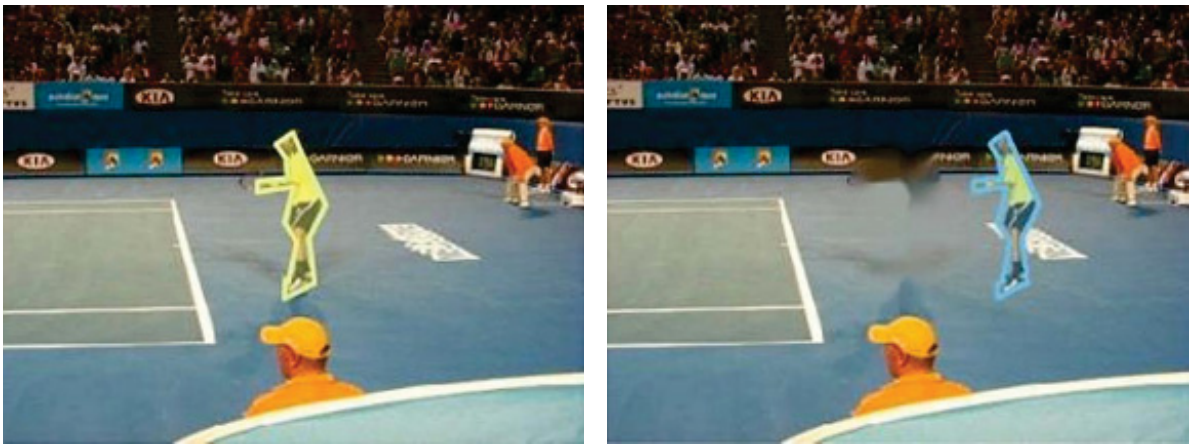


Figure 3.31: The steps of the Few-shot UNsupervised Image-to-image Translation (FUNIT) data augmentation proposed by (Liu et al., 2019). In the first step (called training), the network is trained with different classes of dogs (dog races), and in the second step (called deployment), the network can translate the images of dogs to images of felines. Source: (Liu et al., 2019).

The Few-shot UNsupervised Image-to-image Translation (FUNIT) (Liu et al., 2019) is an unsupervised image-to-image translation method that aims to learn a mapping between two domains of images without using paired data. It consists of two main steps: training and deployment. During training, the model learns to map images from one domain to another using an GAN with a content reconstruction loss. The adversarial loss ensures that the generated

images are indistinguishable from real images. In contrast, the content reconstruction loss ensures that the content of the input image is preserved in the output image. During deployment, a few input images are used to generate diverse output images by randomly sampling from the learned distribution of the target domain. This allows FUNIT to generate various images with different styles and appearances, which can be used for data augmentation in various image classification tasks. Figure 3.31 illustrates the steps of the proposed FUNIT data augmentation. In the first step (called training), the network is trained with different classes of dogs (dog races), and in the second step (called deployment), the network can translate the images of dogs to images of felines. The proposed method uses several images of different classes in the training step, called source images, and learns to translate the images between these classes. Finally, in the deployment step, FUNIT uses a small set of images from the target class to translate from the source classes to the target class.

The InstaBoost (Fang et al., 2019) is an augmentation technique specifically designed to improve instance segmentation. This algorithm uses an inpainting technique (Bertalmio et al., 2003) to remove the interest object from the image and place them in another region of the image. An appearance consistency heatmap (Field et al., 1993) is used to estimate the new region of the image where the object will be placed. The InstaBoost has three main steps. First, object proposals are generated for each image using a Region Proposal Network (RPN). Second, probability maps are computed for each object proposal, which estimates the probability of an object being present in a given image region. Third, the object proposals are augmented by copy-pasting objects from other images based on the probability maps. The probability maps guide the copy-pasting process, which ensures that objects are pasted in regions of the image where they are most likely to appear. This results in a diverse set of augmented images that contain variations of the objects' positions and scales, improving the robustness of the model to variations in object appearance. Figure 3.32 presents an example of InstaBoost data augmentation.



(a) Example of image with a tennis player highlighted in yellow.

(b) The same image but with the tennis player, highlighted in blue, placed in a different image region.

Figure 3.32: Example of InstaBoost data augmentation. Source: (Fang et al., 2019).

Table 3.2 presents a list of papers reviewed in this work with data augmentation techniques.

Table 3.2: List of papers reviewed in this work with data augmentation techniques.

Paper	Author/Year
CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features	(Yun et al., 2019)
Augmentation for small object detection	(Kisantal et al., 2019)
Improved Mixed-Example Data Augmentation	(Summers and Dinneen, 2019)
Few-Shot Unsupervised Image-to-Image Translation	(Liu et al., 2019)
InstaBoost: Boosting Instance Segmentation via Probability Map Guided Copy-Pasting	(Fang et al., 2019)
Random Erasing Data Augmentation	(Zhong et al., 2020)
AugMix: A Simple Method to Improve Robustness and Uncertainty under Data Shift	(Hendrycks et al., 2020)
GridMask Data Augmentation	(Chen et al., 2020b)
ClassMix: Segmentation-Based Data Augmentation for Semi-Supervised Learning	(Olsson et al., 2021)
Random Erasing Data Augmentation	(Zhong et al., 2020)

3.2.1 Salience Based Data Augmentation

The SuperMix (Dabouei et al., 2021) uses Visual Salience to mix images. For each image to be mixed, the algorithm generates a mixing binary mask with the salience information of the respective image. Then, a teacher model already trained in the problem is applied to mix the images, optimize the position of the salient region in the mixed image, and ensure that both salient regions are presented in the final mixed image. Figure 3.33 presents three examples of the SuperMix data augmentation. Input 0 and input 1 are the images being mixed, with the respective salience maps mask 0 and mask 1. The Mixed images are the final image generated.

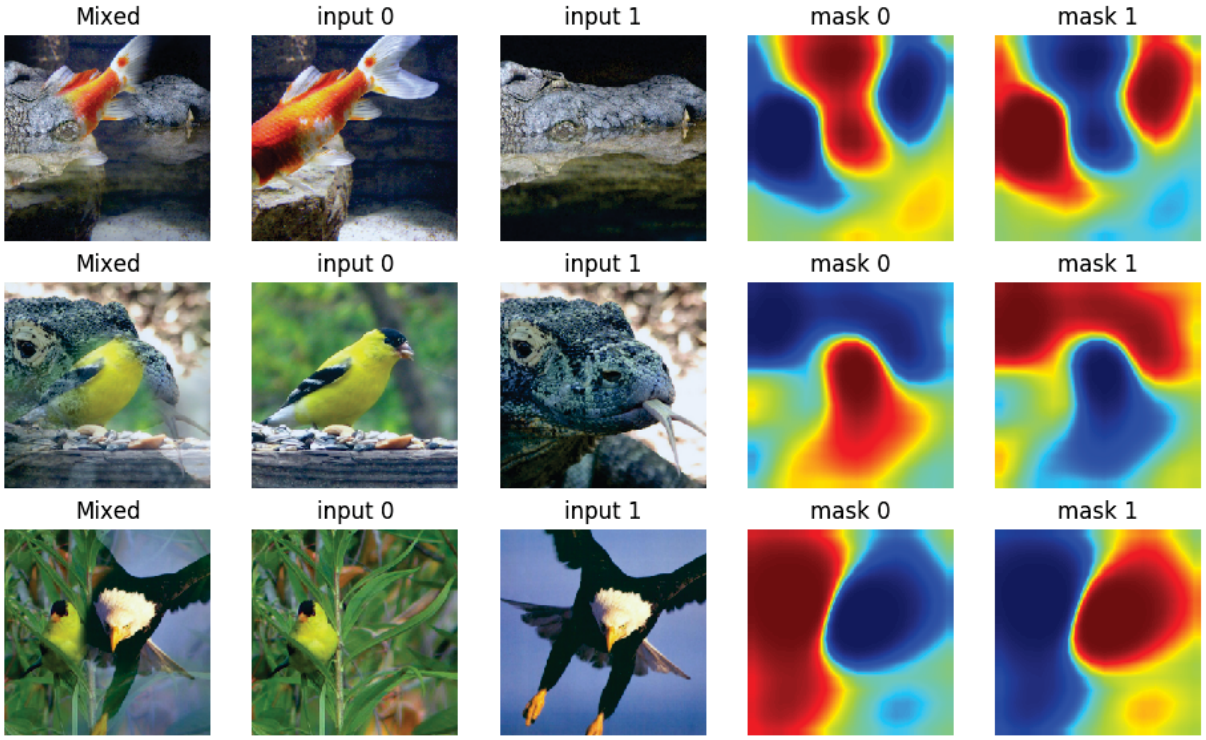


Figure 3.33: Three examples of the SuperMix data augmentation. Input 0 and input 1 are the images being mixed, with the respective salience maps mask 0 and mask 1. The Mixed images are the final image generated. Source: (Dabouei et al., 2021).

The method introduced by (Choi et al., 2021), known as SalfMix, is a single image-based augmentation. This technique involves the random selection of two images from the training dataset. Subsequently, a saliency map is computed for each image, followed by an average pooling operation on these saliency maps to emphasize salient regions instead of individual

salient pixels. The next step entails copying the most salient region from each image to the least salient region within the same image, creating two novel training samples. The SalfMix procedure's final phase entails applying a complementary mixing augmentation to blend the two new images. The complementary mixing methods utilized in this stage are Mixup (Zhang et al., 2017), SaliencyMix (Uddin et al., 2021), and CutMix (Yun et al., 2019). Figure 3.34 display the pipeline of the SalfMix algorithm.

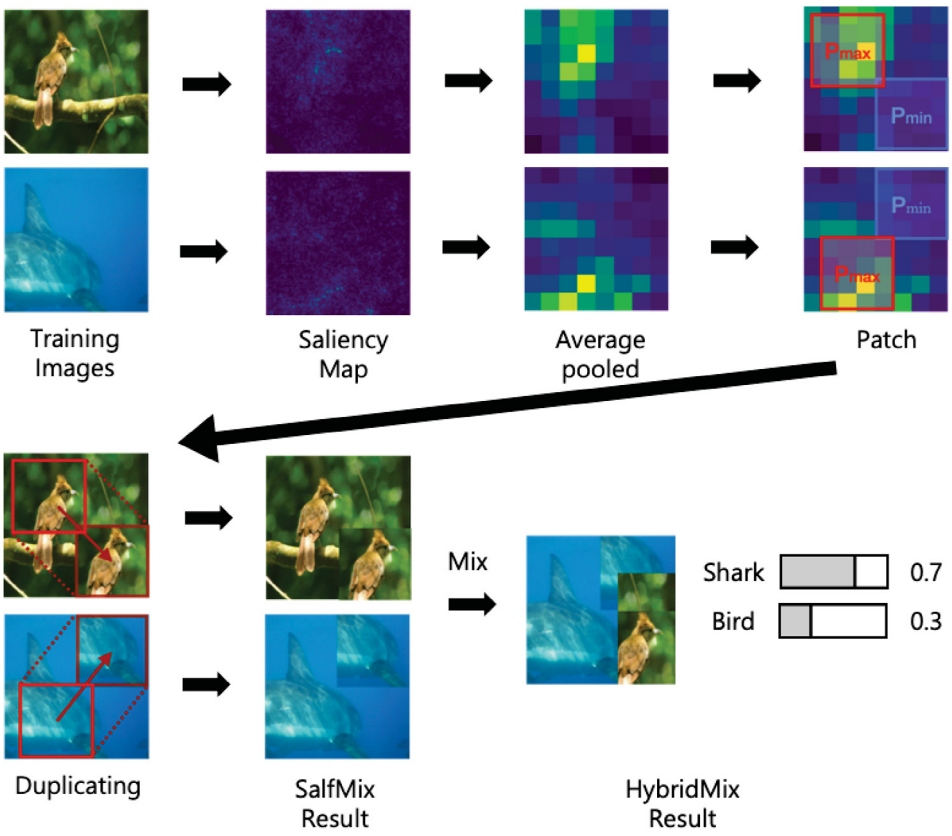


Figure 3.34: SalfMix technique involves the random selection of two images from the training dataset. Subsequently, a saliency map is computed for each image, followed by an average pooling operation on these saliency maps to emphasize salient regions instead of individual salient pixels. The next step entails copying the most salient region from each image to the least salient region within the same image, creating two novel training samples. The SalfMix procedure's final phase entails applying a complementary mixing augmentation to blend the two new images. Source: (Choi et al., 2021).

SnapMix, introduced by the authors in (Huang et al., 2021), represents an augmentation approach that asymmetrically combines two images by leveraging their Semantic Percentage Maps. The initial phase of the SnapMix algorithm involves employing a Convolutional Neural Network (CNN) to generate Semantic Percentage Maps for a given pair of images. Concurrently, a collection of random boxes is extracted from these two images. Subsequently, a series of transformations are applied to the region obtained from the second image, aligning it with the region extracted from the first image. In the concluding stage of SnapMix, the semantic proportions act as a guiding principle for merging the one-hot labels of the two images. Figure 3.35 presents the SnapMix algorithm pipeline for mixing two images.

PuzzleMix (Kim et al., 2020), a refinement of the SaliencyMix technique (Uddin et al., 2021), is an augmentation algorithm designed to mix two images. In pursuit of this objective, PuzzleMix leverages the saliency maps of the two images to identify their most salient regions. The authors then proposed an optimal transportation algorithm to mix these two salient areas,

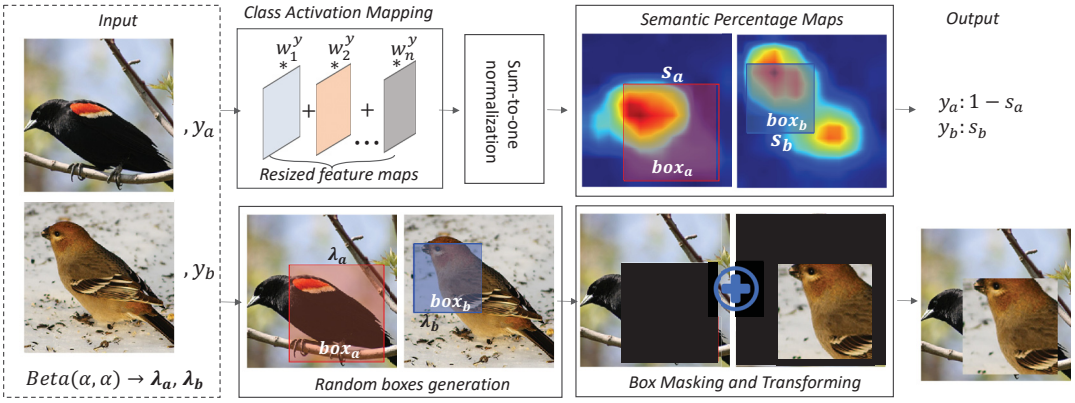


Figure 3.35: The initial phase of the SnapMix algorithm involves employing a CNN to generate Semantic Percentage Maps for a given pair of images. Concurrently, a collection of random boxes is extracted from these two images. Subsequently, a series of transformations are applied to the region obtained from the second image, aligning it with the region extracted from the first image. In the concluding stage of SnapMix, the semantic proportions act as a guiding principle for merging the one-hot labels of the two images. Source: (Huang et al., 2021).

resulting in a fresh image incorporating the most significant features from both original images. Figure 3.36 demonstrates an instance of the PuzzleMix algorithm combining two images.

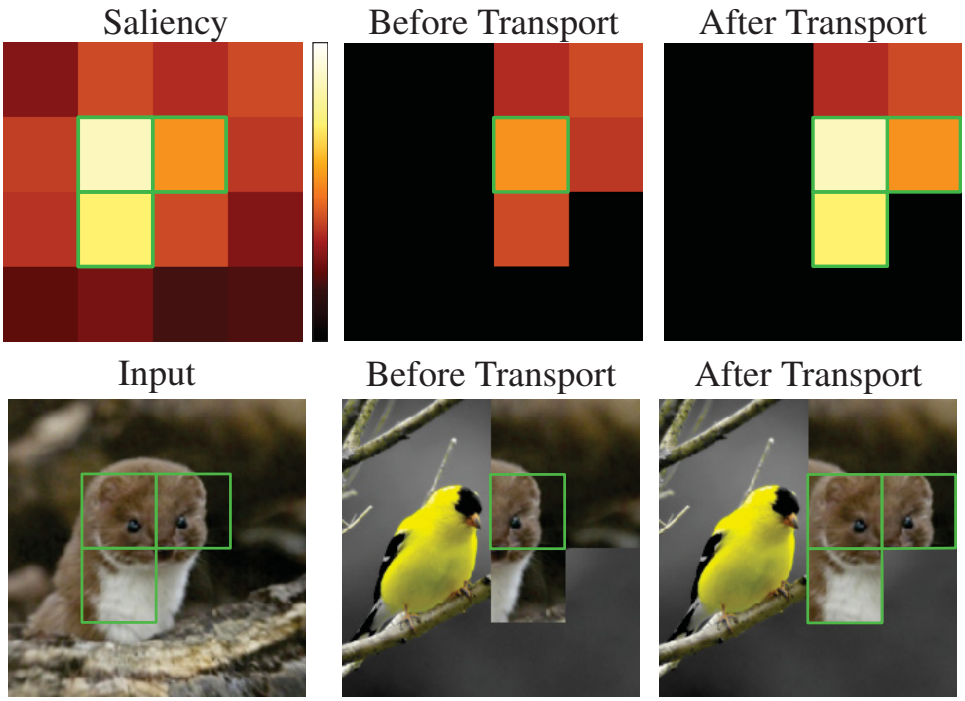


Figure 3.36: The PuzzleMix is an augmentation algorithm designed to mix two images. In pursuit of this objective, PuzzleMix leverages the saliency maps of the two images to identify their most salient regions. The authors then proposed an optimal transportation algorithm to mix these two salient areas, resulting in a fresh image incorporating the most significant features from both original images. Source: (Kim et al., 2020).

The SaliencyMix technique (Uddin et al., 2021) also employs visual saliency to mix images. This approach involves utilizing two images: a target and a source image. The algorithm generates a saliency map for the source image and subsequently chooses a patch containing the most salient region within that image. This selected patch is then combined with the target image. Figure 3.37 presents the sequential process of SaliencyMix augmentation.

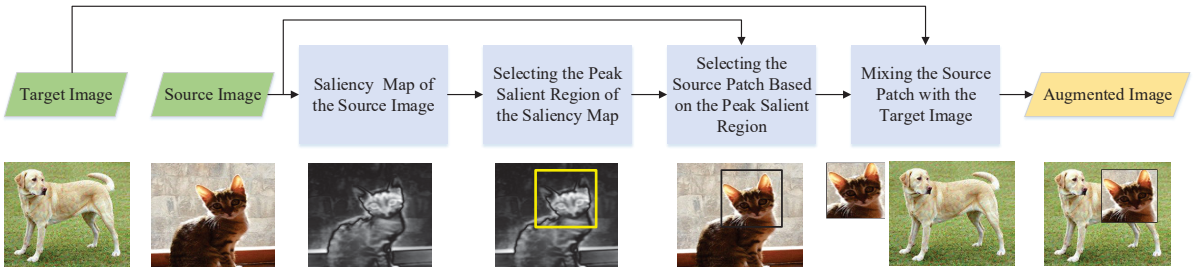


Figure 3.37: The SaliencyMix approach involves utilizing two images: a target and a source image. The algorithm generates a saliency map for the source image and subsequently chooses a patch containing the most salient region within that image. This selected patch is then combined with the target image. Source: (Uddin et al., 2021).

Table 3.3 presents a list of papers reviewed in this work with data augmentation techniques based on visual salience features.

Table 3.3: List of papers reviewed in this work with data augmentation techniques based on visual salience features.

Paper	Author/Year
Puzzle Mix: Exploiting Saliency and Local Statistics for Optimal Mixup	(Kim et al., 2020)
SuperMix: Supervising the Mixing Data Augmentation	(Dabouei et al., 2021)
SalfMix: A Novel Single Image-Based Data Augmentation Technique Using a Saliency Map	(Choi et al., 2021)
SnapMix: Semantically Proportional Mixing for Augmenting Fine-grained Data	(Huang et al., 2021)
SaliencyMix: A Saliency Guided Data Augmentation Strategy for Better Regularization	(Uddin et al., 2021)

3.3 DATA AUGMENTATIONS FOR COVID-19 CT SEGMENTATION PROBLEM

Due to the novelty of COVID-19, only a few studies have proposed data augmentation techniques for COVID-19 segmentation problems. In general, the studies in the COVID-19 CT segmentation field are limited to generic data augmentation applications (Narin et al., 2021; Diniz et al., 2021; Zhang et al., 2022; Salama and Aly, 2022; Yao et al., 2022). The most common operation are variations of an affine transformation, such as random flipping, translation, rotation, and scaling. Müller et al. (2020, 2021) evaluated eight generic data augmentation divided into three categories: spatial, color and noise transformations. An example of data augmentation proposed for COVID-19 CT segmentation is presented in the work of (Yazdekhasty et al., 2021). The authors employed a Conditional GAN (Mirza and Osindero, 2014) to generate CT images. Initially, they extracted the lesion regions from the images, resulting in an image with only the lesion regions. These images were mirrored and then fed into the Conditional GAN to generate new synthetic images of lesions. Subsequently, the generated lesions replaced the original lesions in the original image. The generated images were used to train a Fully Convolutional Network (FCN) architecture adapted with a Channel-wise attention mechanism (Woo et al., 2018) to extract contextual information from the images. Figure 3.38 illustrates the proposed data augmentation pipeline.

The approach proposed in (Mahapatra and Singh, 2021) aims to generate new samples for COVID-19 CT segmentation by using the geometric information of the lesions. To achieve this, the input image X is first processed by a Weakly-Supervised Segmentation (WSS) module, which generates a segmentation mask S_X. Then, S_X is fed into a Spatial Transformer Network (STN) to create a new segmentation mask by altering the shape, location, scale, and orientation attributes of the COVID-19 lesions. The output of the STN (Jaderberg et al., 2015) is used as input to a GAN. The generator of the GAN uses the STN mask to produce a new sample

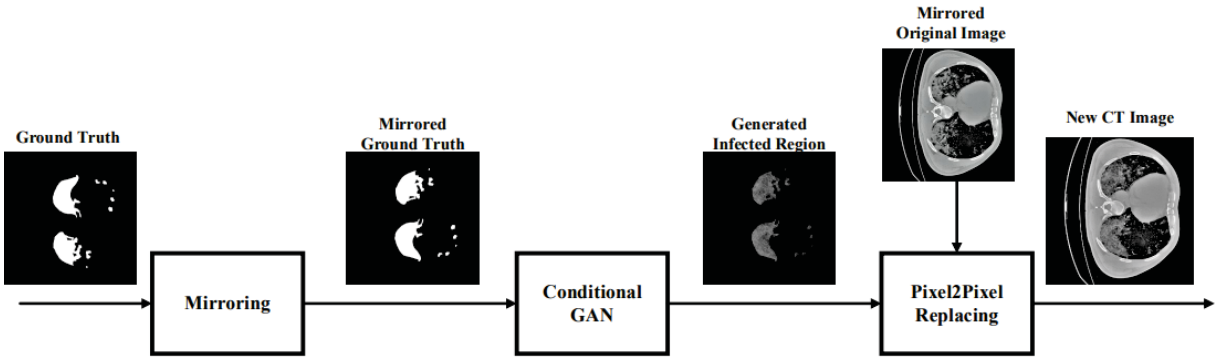


Figure 3.38: Initially, they extracted the lesion regions from the images, resulting in an image with only the lesion regions. These images were mirrored and then fed into the Conditional GAN to generate new synthetic images of lesions. Subsequently, the generated lesions replaced the original lesions in the original image. The generated images were used to train a FCN architecture adapted with a Channel-wise attention mechanism. Source: (Yazdekhasty et al., 2021).

image, which the discriminator evaluates. The discriminator includes two classifier networks that assess the accuracy of the class and shape of the new image. The pipeline of this proposed data augmentation approach is depicted in Figure3.39.

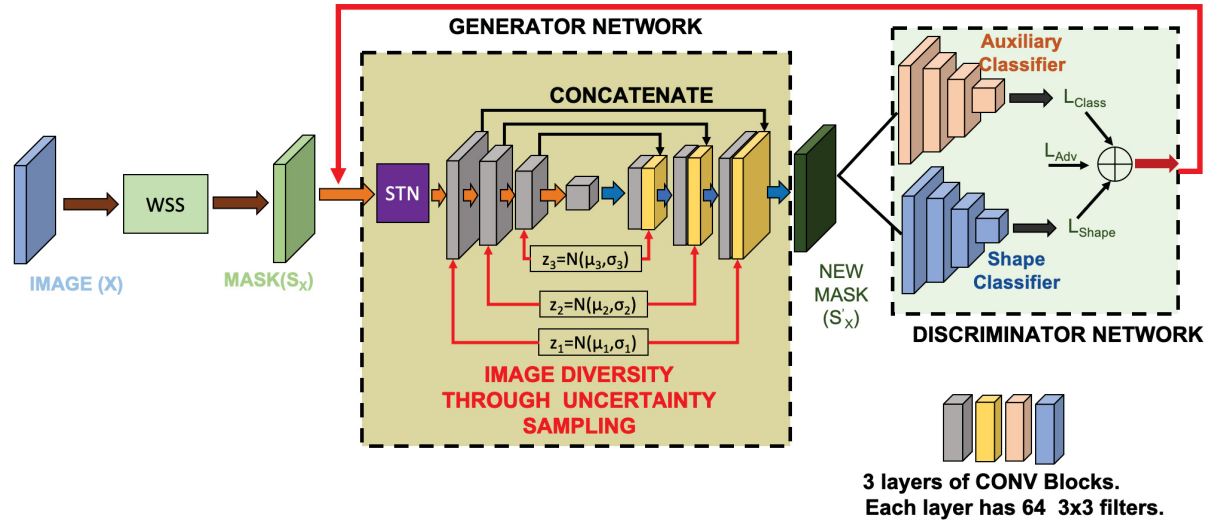


Figure 3.39: The input image X is fed into a WSS module to generate a segmentation mask S_X. Then, S_X inputs an STN to generate a new segmentation mask changing the shape, attributes of location, scale, and orientation of the COVID lesions. The output of the STN inputs a GAN. Source: (Mahapatra and Singh, 2021).

The authors in (Jiang et al., 2021) proposed a novel data augmentation technique for COVID-19 using a dual-network GAN. The generator and discriminator are designed to extract global and local features, respectively, and the generator consists of two sub-discriminators to differentiate multi-resolution images. The authors also propose the Dynamic Element-wise Sum (DESUM) to balance the information extracted during the generator step and the Dynamic Feature Matching (DFM) to weigh the loss of input with different resolutions dynamically. Figure 3.40 illustrates the proposed GAN, where the top image shows the generator training process that receives two segmentation masks of different resolutions and generates synthesized images in high and low resolutions with the lesions of the segmentation maps.

The authors of (Chen et al., 2023) also proposed a data augmentation technique for COVID-19 CT segmentation. They used the Fourier transformation to convert CT images from cancer patients to CT images with COVID-19 lesions. Then, they applied a teacher-student

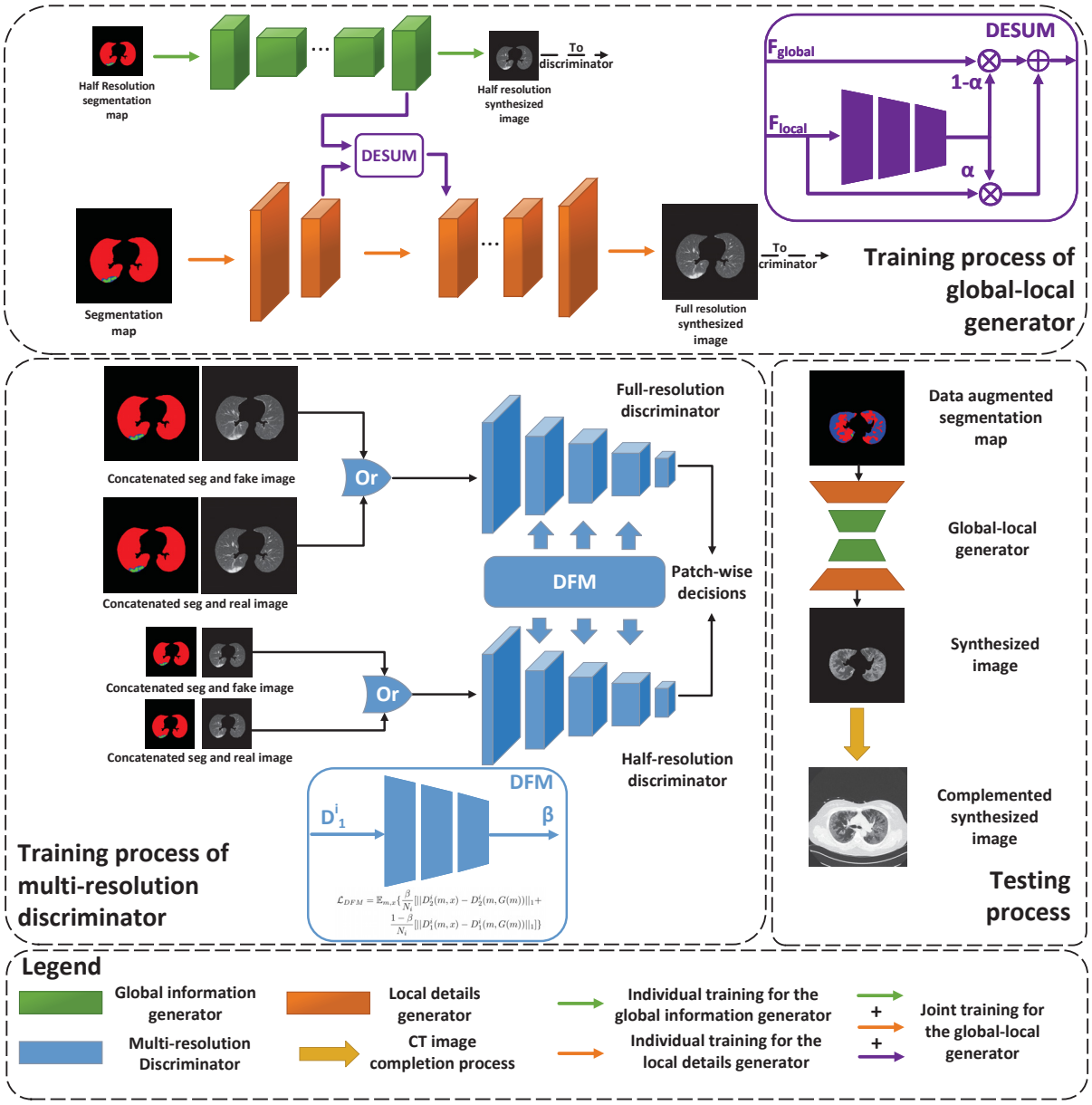


Figure 3.40: The top image presents the generator training process, the bottom left image presents the discriminator training process, and the bottom right image presents the generator test process. The orange generator extracts global information from the high-resolution image, and the green generator extracts local information from the low-resolution image. The bottom left image shows the discriminator training process with two sub-discriminators, one for high-resolution and the other for low-resolution images. The output of the discriminator is a new image with the lesions of the generated segmentation map. The bottom right image shows the generator test process, which receives a segmentation map and outputs the final image. Source: (Jiang et al., 2021).

architecture to generate segmentation masks for the new COVID-19 CT images. The student architecture received the cancer CT images converted to COVID-19 and unlabeled CT images with COVID-19 lesions to output the segmentation masks. Although the student network achieved good results for cancer CT segmentation, it did not perform well for COVID-19 CT segmentation. To address this issue, the authors introduced a teacher architecture trained with unlabeled COVID-19 CT images to assist the student architecture in extracting robust COVID-19 features. The proposed method is illustrated in Figure 3.41.

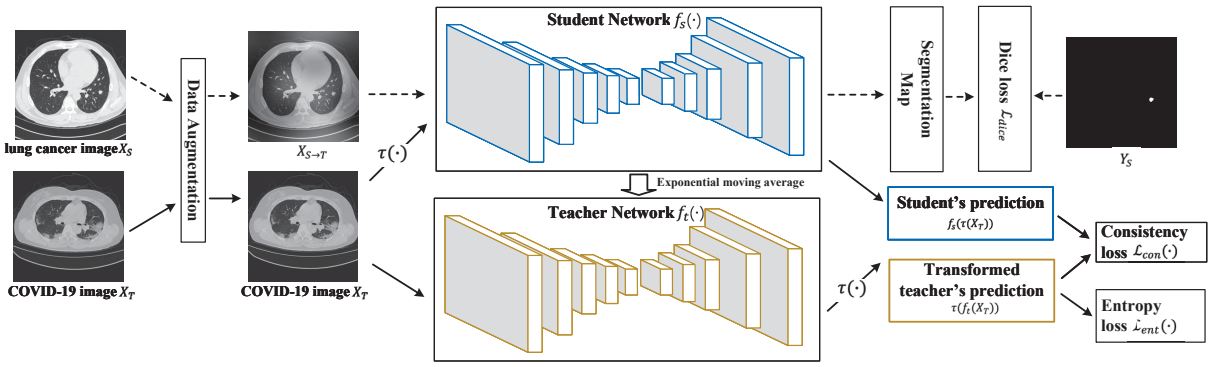


Figure 3.41: The proposed data augmentation uses the Fourier transformation to convert CTs from cancer patients to CTs with COVID-19 lesions. Then, a teacher-student architecture is applied to generate segmentation masks for the new COVID-19 CTs. The student architecture receives the cancer CTs converted in COVID-19 CTs and a set of unlabeled CTs with COVID-19 lesions and outputs the segmentation masks. The teacher architecture is trained with unlabeled COVID-19 CTs to help the student architecture to extract robust features of COVID-19. Source: (Chen et al., 2023).

Table 3.4 presents a list of papers reviewed in this work with data augmentation techniques proposed for COVID-19 CT segmentation.

Table 3.4: List of papers reviewed in this work with data augmentation techniques proposed for COVID-19 CT segmentation.

Paper	Author/Year
Segmentation of Lungs COVID Infected Regions by Attention Mechanism and Synthetic Data	(Yazdekhasty et al., 2021)
CT Image Synthesis Using Weakly Supervised Segmentation and Geometric Inter-Label Relations For COVID Image Analysis	(Mahapatra and Singh, 2021)
COVID-19 CT Image Synthesis With a Conditional Generative Adversarial Network	(Jiang et al., 2021)
A teacher–student framework with Fourier Transform augmentation for COVID-19 infection segmentation in CT images	(Chen et al., 2023)

3.4 FINAL CONSIDERATIONS

This chapter reviews various studies that address the COVID-19 segmentation problem and the data augmentation techniques proposed for COVID-19 CT segmentation datasets. Most of these studies utilized popular segmentation models in Semantic Segmentation, with U-Net being the most prevalent architecture. Additionally, this chapter examines studies on generic data augmentation techniques for classification and segmentation problems. Generally, these works applied transformations such as rotation, translation, scaling, shearing, flipping, and adding noise or distortions. Some studies also explored mixing image algorithms for data augmentation, while recent approaches proposed specific domain augmentation techniques. Furthermore, there are some works that explored visual salience features on data augmentation techniques and others that used GANs for data augmentation in the COVID-19 segmentation problem. This study aims to introduce a novel data augmentation technique for the COVID-19 segmentation problem. The next chapter will detail the proposed data augmentation approach and the steps to achieve the research objectives.

4 PROPOSED WORK

This chapter presents the steps in developing this work and provides a detailed overview of the proposed training pipeline and the novel Data Augmentation method.

4.1 PROPOSED WORK

Semantic segmentation of COVID-19 Computed Tomographys (CTs) scans has emerged as a crucial research domain in response to the global COVID-19 pandemic (Chen et al., 2023; Fung et al., 2021; Sun et al., 2022; Enshaei et al., 2022) and this field representing one of the numerous endeavors undertaken to combat the COVID-19 disease (Chen et al., 2020a; Ai et al., 2020; Shi et al., 2021). Chapter 3 explores various studies that have leveraged Deep Learning techniques and Deep Neural Networks to accurately segment COVID-19 CTs scans, yielding remarkable outcomes in this undertaking (Shi et al., 2021). However, Semantic Segmentation methods based on Deep Neural Networks suffer from a significant drawback, namely the substantial data requirements for training the networks effectively (Shorten and Khoshgoftaar, 2019; Ankile et al., 2020). In the context of COVID-19 CT segmentation, two additional factors further limit the utilization of Deep Neural Networks. Firstly, labeling Semantic Segmentation is arduous and time-consuming, necessitating accurate labeling for every pixel in the image to avoid erroneous network convergence (Shi et al., 2021; Cao and Bao, 2020). Moreover, the labeling of CT segmentation datasets must be carried out by highly specialized doctors capable of correctly annotating the lesion regions within the image (Shi et al., 2021).

Data augmentation techniques have been extensively employed in various Deep Learning domains to address these limitations (Shorten and Khoshgoftaar, 2019; Ruiz et al., 2019, 2020b). Nonetheless, in the context of COVID-19 CT segmentation, a thorough investigation of data augmentation methods is lacking, as existing approaches rely on generic techniques commonly utilized in diverse Deep Learning problems (Müller et al., 2021; Narin et al., 2021; Diniz et al., 2021; Zhang et al., 2022; Salama and Aly, 2022; Yao et al., 2022). Drawing upon these insights, the primary objective of this work is to present an innovative data augmentation approach designed explicitly for segmenting COVID-19 lesions in CT scans and create new CT scans with COVID-19 lesions.

The proposed technique involves harnessing Visual Salience, a subjective perceptual attribute found in objects that enables them to stand out from their surrounding areas and instantaneously capture the human brain's attention (Itti, 2007; Krinski et al., 2019). By leveraging Visual Salience, it becomes feasible to adjust the segmentation complexity of new CT examples, generating instances with COVID-19 lesions exhibiting varying levels of salience. This enables the generation of new examples with COVID-19 lesions that possess either a heightened or diminished salience level, thereby providing a broader range of segmentation difficulty. As presented in Figure 4.1, there are differences in size and position of COVID-19 lesions in a CT.

Upon visual examination of the images and their corresponding ground-truth data, it becomes evident that the lesions depicted in Sub-figure 4.1(a) exhibit a higher level of salience compared to the lesions in Sub-figure 4.1(b), while the lesions in Sub-figure 4.1(c) possess an even lower salience level. This disparity in salience levels arises from factors such as the proportion and spatial position of the lesions within the images. The varying levels of salience directly impact the training phase of a deep learning-based segmentation network, making examples like Sub-figure 4.1(a) and Sub-figure 4.1(b) relatively easier for the model to

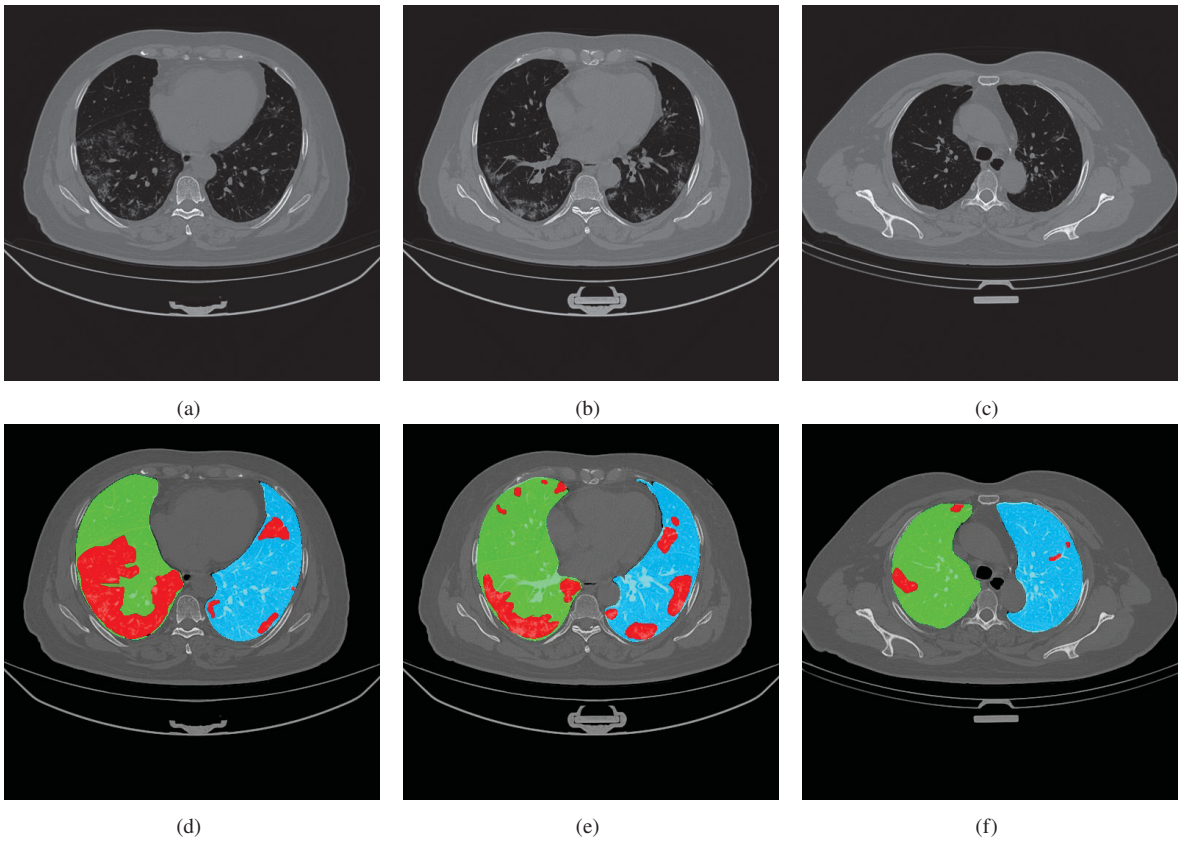


Figure 4.1: Examples of different visual salience level in COVID-19 a CT. Sub-figures 4.1(a), 4.1(b), and 4.1(c) present three examples of CTs from the Zenodo (Jun et al., 2020; Ma et al., 2021) dataset and its corresponding segmentation mask in Sub-figures 4.1(d), 4.1(e), and 4.1(f). Green is the label for the Left Lung, blue is the label for the Right Lung, and red is the label for the COVID-19 lesion. Source: (Jun et al., 2020; Ma et al., 2021).

segment compared to examples like Sub-figure 4.1(c). The concept behind data augmentation utilizing Visual Salience is to consider these differences while generating new examples and aligning the new data to create a balanced dataset with optimized examples in terms of Visual Salience. As a result, the newly generated examples will not be as easy as those illustrated in Sub-figure 4.1(a) nor as challenging as those depicted in Sub-figure 4.1(c). The proposed approach for this novel data augmentation based on Visual Salience involves three key steps: evaluation of segmentation networks, assessment of data augmentation techniques, and creation of a new data augmentation workflow.

The segmentation of COVID-19 CTs is an emerging field, as discussed in Chapter 3, where numerous studies have employed Deep Learning techniques and Deep Neural Networks to achieve remarkable results in COVID-19 CT segmentation (Shi et al., 2021). Nevertheless, due to the rapid introduction of novel approaches heightened by the global pandemic, there is a demand for comprehensive evaluations to be conducted. In order to establish a standardized basis for comparing various architectures on multiple recently introduced datasets, the initial phase of this study involves conducting a thorough benchmarking of multiple segmentation architectures. In this step, twenty encoder models where combined with six decoder models, totaling one hundred twenty segmentation models evaluated. Section 6.1 in Chapter 6 details the experiments conducted in this phase; the evaluated networks; the training, validation, and testing procedure; the datasets utilized; and the results obtained.

One notable finding from the comparison between the Encoder-Decoder architectures is that the datasets exhibit certain limitations, including a limited number of images and a

significant class imbalance. These factors can influence the learning process of deep neural networks. Aiming to overthrow these problems, the best network evaluated in the previous step is used to perform a comparison between twenty data augmentation techniques across five recently proposed datasets. Also, a deep investigation on how different probabilities of applying a data augmentation was performed and each data augmentation was evaluated with ten different probabilities. Section 6.2 in Chapter 6 details the experiments conducted in this phase; the augmentation techniques evaluated; the training, validation, and testing procedure; the datasets utilized; and the results obtained.

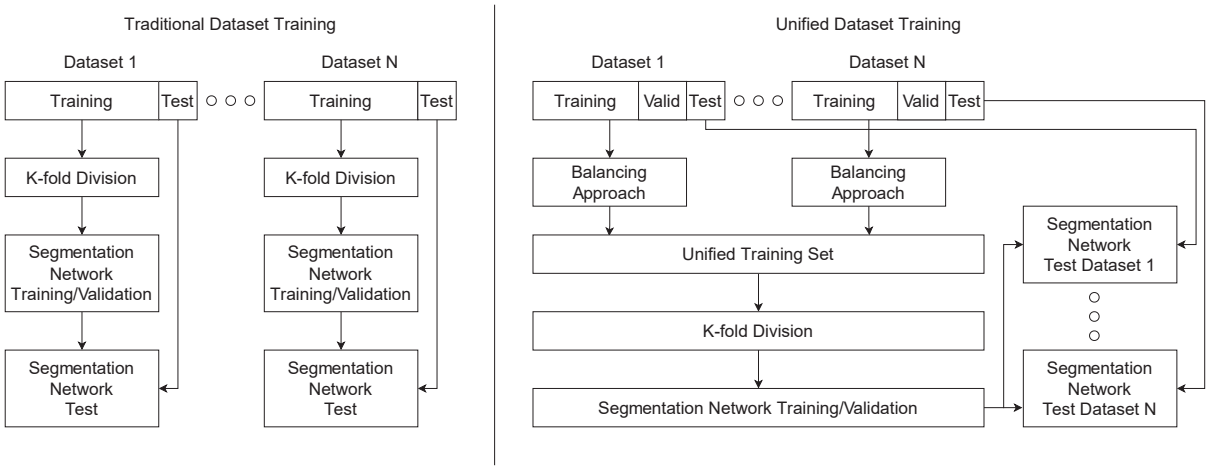


Figure 4.2: A comparison is made between the traditional training workflow and the unified training methodology. In the traditional approach (left), the datasets train the network individually. On the other hand, in the unified training strategy (right), the datasets are combined to train the network. Source: (Krinski et al., 2023).

Moreover, an alternative training methodology is examined apart from the conventional training approach, where the datasets are used individually to train the network. The proposed training methodology involves merging the training sets from the five datasets into a larger unified training set and the twenty data augmentation techniques were also evaluated with this new training strategy. As depicted in Figure 4.2 (right), the individual training subsets are consolidated into a single extensive set, while the testing procedure remains unchanged. The original class labels were adapted to **background** and **lesion** to ensure a fair comparison. Furthermore, a balancing procedure was implemented to ensure a fair representation of each original sample in the new combined training set. For smaller datasets, namely CC-CCII, MedSeg, MosMed, and Zenodo, their samples were repeated n times to match the number of samples in the largest dataset, Ricord1a (details about the datasets are provided in Chapter 5). The value of n_i for each dataset i was calculated as the ceiling of the number of images in the Ricord1a training set (calculated as the number of images in the Ricord1a dataset minus the number of images in the test and validation sets of the Ricord1a dataset) divided by the number of images in the corresponding training set of the dataset x_i (calculated as the number of images in the dataset x_i minus the number of images in the test and validation sets of the dataset x_i), as described by Equation 4.1. Section 6.2 in Chapter 6 details the experiments conducted in this new training strategy and the results obtained.

$$n_i = \left\lceil \frac{|Ricord1a|}{|x_i|} \right\rceil \tag{4.1}$$

The third phase of this work involves the development and evaluation of a novel technique for data augmentation. This technique utilizes a Generative Adversarial Network (GAN) model

to generate healthy CT scans of lung regions and combine them with existing labeled COVID-19 lesions to create additional samples and enhance the segmentation training process. This phase included two evaluations of the proposed workflow: one version without the salience algorithm, where lesions were randomly added to the healthy CT scans, and an improved version that uses visual salience distance to add lesions.

4.1.1 Proposed Augmentation - DACov

As presented in Figure 4.3, the overall workflow consists of three main steps: healthy CT generation with GANs, lung segmentation algorithm, and lesion addition algorithm. In the initial step of the data augmentation workflow (healthy CT generation), the GAN is trained using two images: CT scans with COVID-19 lesions and CT scans without lesions, or only CT scans without lesions, depending on the generative network. By learning from these images, the GAN can effectively generate healthy CT scans without COVID-19 lesions. In the subsequent step, a segmentation model is trained in the CC-CCII and Zenodo datasets to perform lung segmentation. The segmentation model is trained in these two datasets because they are the only two datasets that provided the lung field annotation in the ground-truth masks. After training, the segmentation model takes the images produced by the GAN and generates lung segmentation masks for these new images. These masks ensure that the newly added lesions are placed only within the lung regions of the images.

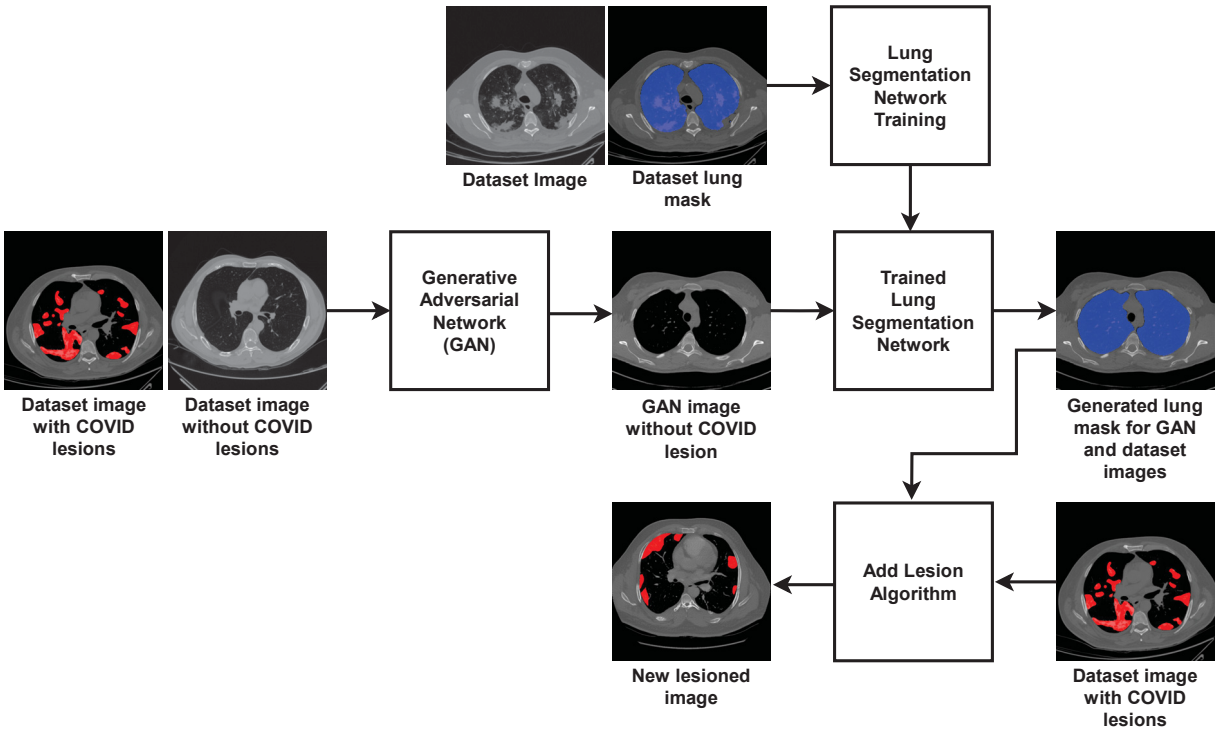


Figure 4.3: Workflow of the DACov version of the proposed approach, which is divided into three main steps. A GAN is trained using two classes, healthy and COVID-19-infected lungs, or only healthy lungs, depending on the network. Thus, the network generates new healthy samples that subsequently have the lung region labeled by a pre-trained segmentation model, ensuring that when the lesions are added later they are cropped by the lung mask. The last step randomly adds labeled lesions from real CT images into the generated samples. Labeled lesion regions were highlighted in red, labeled lung region was highlighted in blue. Source: (Krinski et al., 2023).

The final step of the proposed workflow is the algorithm that adds lesions, which uses the lung segmentation mask to place the new COVID-19 lesion inside the lung region, resulting

in a new CT scan with COVID-19 lesions. This step receives the newly generated image without COVID-19 lesions (healthy images) and the respective lung masks, a COVID-19 lesion image (dataset image), and the respective segmentation mask. The algorithm uses the lung masks from both images to generate the bounding boxes of each lung (right and left lungs). Then, for each healthy image generated by the GAN, the algorithm tries to find a corresponding image from the dataset in which the bounding boxes of the lungs are similar sizes. The size difference is limited to a range of 10% smaller or 10% bigger. This limitation is necessary due to the great difference between the size of the lungs in the dataset, which could lead to distortion in the lesions when added to another image. After finding an image in the dataset that matches the size, the algorithm uses the AddWeighted algorithm from OpenCV (Bradski, 2000) library to add the lesions in the new lung images generated by the GANs. Sub-Section 6.3.3 in Chapter 6 details the experiments conducted with the DACov version of the proposed augmentation.

4.1.2 Proposed Augmentation - SalDACov

The SalDACov version of the proposed augmentation has the same steps as the DACov version: healthy CT generation with GANs, lung segmentation algorithm, and lesion addition algorithm. Figure 4.4 presents the overall workflow. The healthy CT generation and lung segmentation algorithm are the same as the DACov version. However, in this version, the lesion addition algorithm is replaced by a visual salience algorithm, presented in Figure 4.5, to guide the choice of the lesions that will be added to each lung.

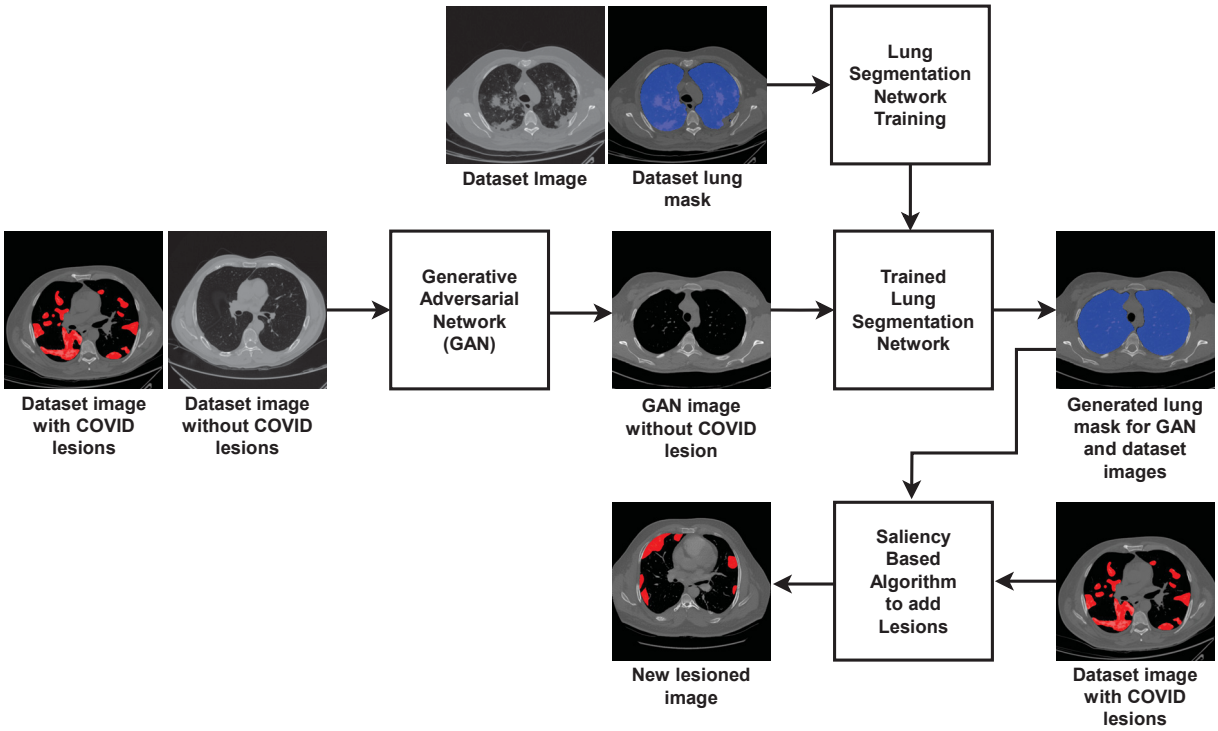


Figure 4.4: Workflow of the SalDACov version of the proposed approach, which is divided into three main steps. A GAN is trained using two classes, healthy and COVID-19-infected lungs, or only healthy lungs, depending on the network. Thus, the network generates new healthy samples that subsequently have the lung region labeled by a pre-trained segmentation model, ensuring that when the lesions are added later they are cropped by the lung mask. The last step uses visual salience features to add labeled lesions from real CT images into the generated samples. Labeled lesion regions were highlighted in red, labeled lung region was highlighted in blue.

The salience algorithm receives the images generated by the GANs (set A) and the dataset images (set B). The algorithm also receives the lung masks generated in the segmentation

algorithm step for both sets of images. Then, in the first step of the algorithm, the bounding boxes of each lung (left and right lung) are generated for both sets of images. Also, in this step, a salience feature vector is generated for each lung region (left and right lungs) of the images generated by the GAN and for each lesion region (left and right lesions) of the images from the dataset.

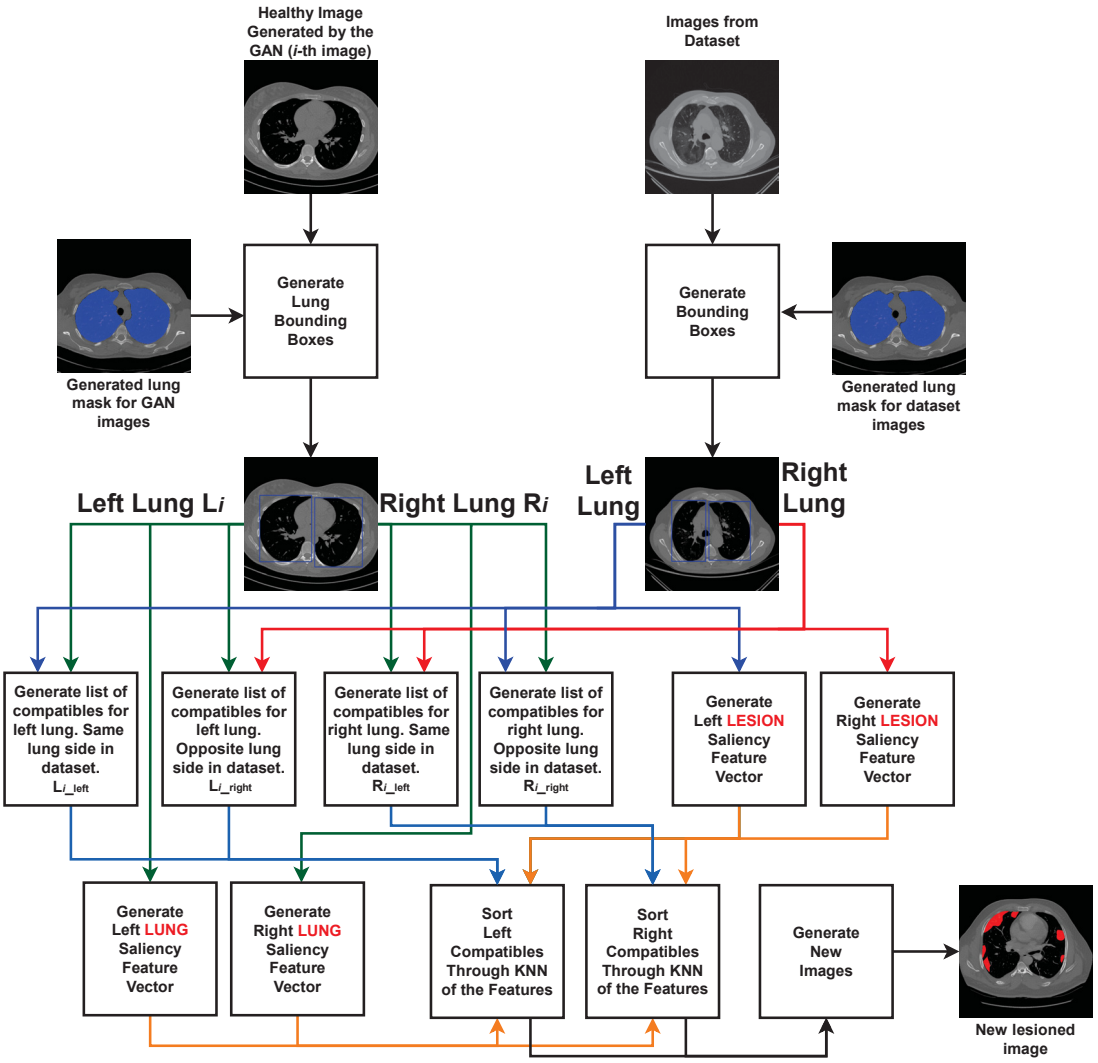


Figure 4.5: Workflow of the salience algorithm in the proposed data augmentation approach. In the first step of the algorithm, the bounding boxes of each lung (left and right lung) are generated (healhty and dataset images). Also, in this step a salience feature is generated for each lung region (left and right lungs) of the images generated by the GANs and for each lesion region (left and right lesions) of the images from the dataset. Then, for each lung, the algorithm generates two lists of compatibles lesions for each lung. The first list stores the compatible lesions in the same side lung, and the second list stores the compatible lesions in the opposite side lung (flipped). After that, the K-Nearest Neighbors (KNN) (Fix and Hodges, 1989) is applied to sort the compatibles lesions using the salience feature vector distance between the healthy image and the compatible lesions in the list. In the last step, the lesions are added to the healthy images.

Visual salience comprises three features: color, intensity, and orientation (Itti, 2007). In the proposed augmentation, the intensity was extracted with the grayscale histogram, and the orientation was extracted with the Local Binary Patterns (LBP) (He and Wang, 1990; Wang and He, 1990) histogram. The color feature was left out since the CT images are only grayscale. So, the salience feature vector comprises 128 features encoding the grayscale histogram and 128 features encoding the LBP histogram, totaling a vector of 256 features.

In the next step, the algorithm uses the size of the lung bounding box to find compatible lesions for each lung in the image generated by the GANs. More precisely, for each healthy image generated by the GAN, the algorithm searches in the images from the dataset to build a list of compatible lungs. The algorithm looks at each image in the dataset, attempting to find at least 25 compatible lungs. The search can be stopped earlier (i.e., early stop for the dataset) if there are at least 100 elements per list (easy case, with many compatibles). If the image i does not have at least 25 elements on its list after going through all lesions on the dataset (i.e., early stop for image i), the image is excluded from the following steps thus reducing the total number of artificial images (worst case, with the least compatible cases).

Similar to the DACov version of the proposed augmentation, the size of the lung bounding box is used to build the list of compatible lungs. The difference in size is limited to the range [0.9,1.1]. However, different from the DACov version, which enforced both lungs in the healthy image to match the size of the lungs in the image from the dataset, this version makes the match independent. Each lung in the image might receive a lesion from a different image.

Furthermore, the algorithm generates two lists of compatible lesions for each lung. That is, the left lung L of the healthy image i has a list of images from the dataset that the left lung is size compatible, list Li_left; and a list of images from the dataset that the right lung is size compatible, list Li_right, (the lesions in the right in the dataset will be flipped when added to the left lung in a healthy image). The right lung R of the healthy image i also has a list of images from the dataset that the right lung is size compatible, list Ri_right; and has a list of images from the dataset that the left lung Ri_left is size compatible (the lesions in the left in the dataset will be flipped when added to the right lung in a healthy image).

After that, the proposed algorithm uses the KNN (Fix and Hodges, 1989) with the Euclidean distance to sort the compatible lesions based on the distance between the salience feature vector of the healthy image and the salience feature vector of the compatible lesions in the list. That is, for each list of compatible lungs l, the KNN with a size of $|l|$ is applied to sort that list using the distance between the salience feature vector of the healthy image generated by the GAN and the compatible images from the dataset. Then, for each lung, the algorithm takes a compatible lesion from the compatible list based on the salience distance and uses the AddWeighted algorithm from OpenCV (Bradski, 2000) library to add the lesion. The lesion can be chosen considering the higher salience distance (randomly chosen between the top 10 last elements in the list) or the lower salience distance (randomly chosen between the top 10 first elements).

4.2 FINAL CONSIDERATIONS

This chapter outlined the proposed data augmentation method and the steps to develop this work. The data augmentation approach leverages GANs to generate healthy CT images without COVID-19 lesions. A segmentation model is then trained to produce segmentation masks for the lung regions in these newly generated images. These lung masks guide the augmentation algorithm in accurately placing new lesions within the image. Subsequently, a salience algorithm identifies the specific lesion region to be added to the healthy image produced by the GAN. The effectiveness of the proposed augmentation can be assessed based on salience distance (maximum, minimum, and random) and the lesion's position relative to the original dataset image (same side, opposite side, and random). The next chapter will discuss the datasets and evaluation metrics used to assess each stage of this work.

5 METHODOLOGY

This chapter is structured into three main sections. Firstly, Section 5.1 introduces the dataset employed for training both the segmentation models and Generative Adversarial Networks (GANs). Secondly, Section 5.2 details the evaluation metrics utilized to assess the performance of the segmentation models and GANs. Thirdly, Section 5.3 describes the statistical methods employed in this study. Lastly, Section 5.4 concludes the chapter.

5.1 DATASETS

In this work, a total of six datasets were used: CC-CCII (Zhang et al., 2020), MedSeg (MedSeg, 2021), MosMed (Morozov et al., 2020), Ricord1a (Tsai et al., 2020), Ricord1b (Tsai et al., 2020), and Zenodo (Jun et al., 2020; Ma et al., 2021). The following sub-sections present details about each dataset.

5.1.1 CC-CCII

The first dataset is a subset of the CC-CCII (Zhang et al., 2020) dataset comprising 750 images with segmentation masks with four labels: Background, Lung Field, Ground Glass Opacity (GGO), and Consolidation. Each image has dimensions of 512×512 pixels. Examples of images and their corresponding segmentation masks from the CC-CCII dataset are shown in Figure 5.1. The Lung Field is represented in blue, the GGO in green, and Consolidation in red. Regions without any color correspond to the Background regions. Additionally, the pixel distribution for each label in the images is unbalanced, with the classes occupying the following mean proportion: Background (0.87152), Lung Field (0.11691), GGO (0.00802), and Consolidation (0.00353).

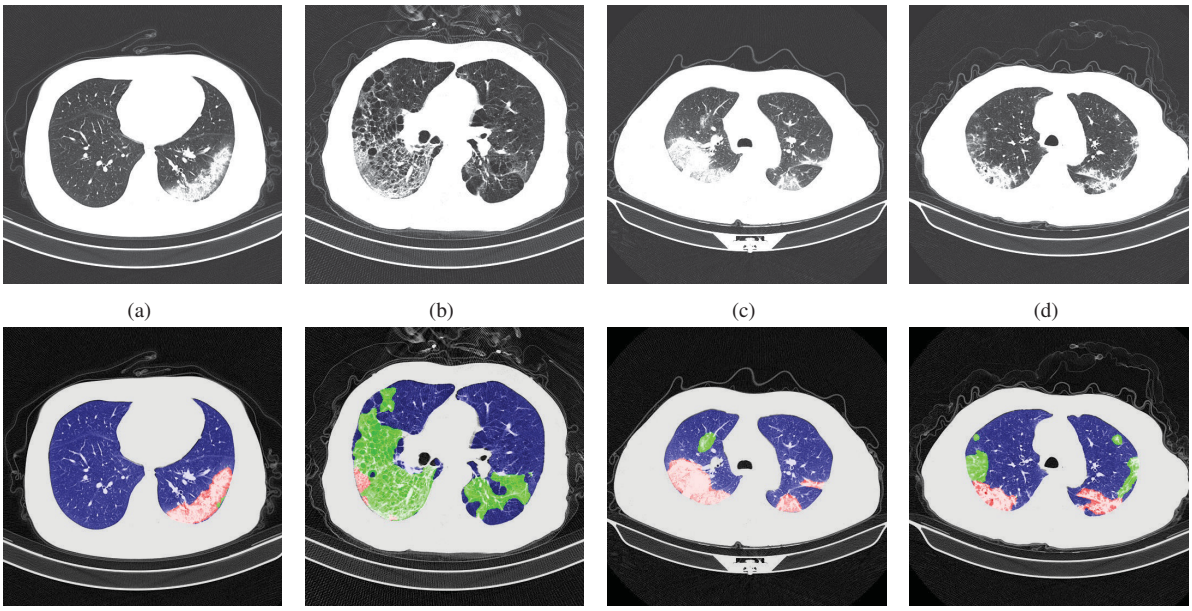
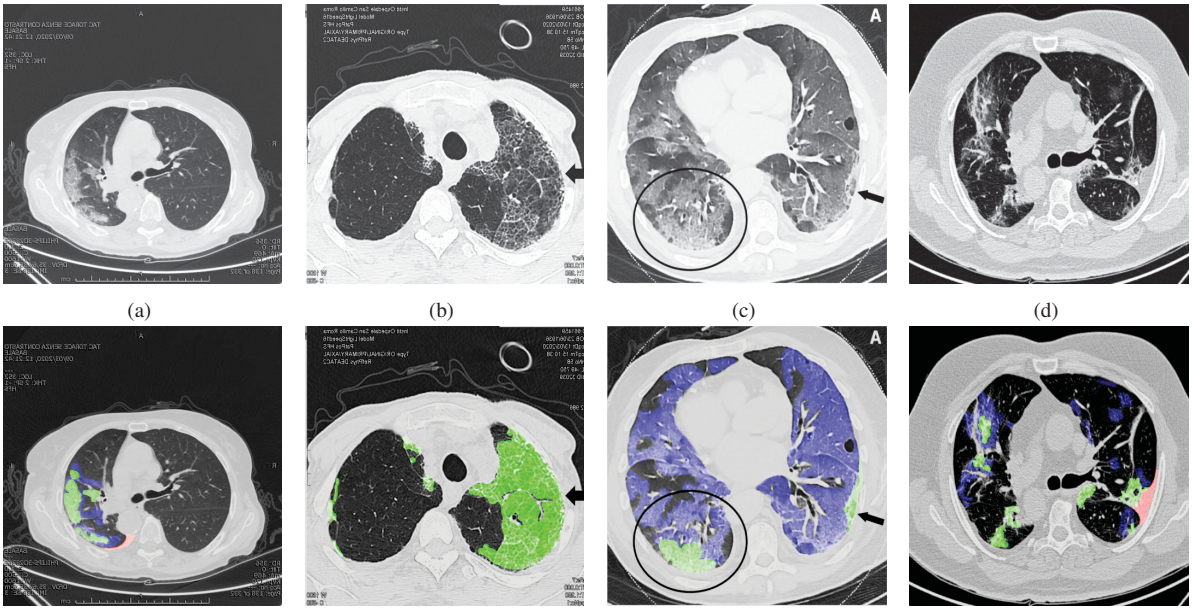


Figure 5.1: Examples of images and segmentation masks from CC-CCII datasets. The Lung Field is presented in blue, the GGO in green, and Consolidation in red. The regions without any color are the Background regions. Source: (Zhang et al., 2020).

5.1.2 MedSeg

The MedSeg (MedSeg, 2021) dataset includes 929 images and labels for four classes: Background, GGO, Consolidation, and Pleural Effusion, with each image having a size of 512×512 pixels. Examples of images and segmentation masks from the MedSeg dataset are presented in Figure 5.2, where the GGO is depicted in blue, Consolidation in green, and Pleural Effusion in red. The regions without any color represent the Background regions. This dataset presents some issues like coarse segmentation masks and trash markings, as shown in Sub-figures 5.2(f) and 5.2(c), that make the learning process of Deep Neural Networks difficult. The number of pixels for each label in the images is unbalanced, with the classes occupying the following mean proportion: Background (0.98563), GGO (0.01072), Consolidation (0.00351), and Pleural Effusion (0.0001).



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 5.2: Examples of images and segmentation masks from MedSeg dataset. The GGO is presented in blue, the Consolidation in green and Pleural Effusion in red. The regions without any color are the Background regions. Source: (MedSeg, 2021).

5.1.3 MosMed

The MosMed (Morozov et al., 2020) dataset is composed by 2,049 images, with labels for two classes: Background and GGO-Consolidation, with each image being of size of 512×512 pixels. Figure 5.3 presents examples of images and segmentation masks from the MosMed dataset. The GGO-Consolidation is presented in red and the regions without any color are the Background regions. The MosMed is the most unbalanced dataset, with the classes occupying the following mean proportion: Background (0.99810) and GGO-Consolidation (0.00189).

5.1.4 Zenodo

The Zenodo (Jun et al., 2020; Ma et al., 2021) dataset comprises 3,520 images, each of size 512×512 pixels, with labels for four classes: Background, Left Lung, Right Lung, and Infections. Examples of images and their corresponding segmentation masks from the Zenodo dataset are shown in Figure 5.4, where the Left Lung is presented in blue, the Right Lung in green, and the Infections in red. The regions without any color represent the Background regions. The Zenodo

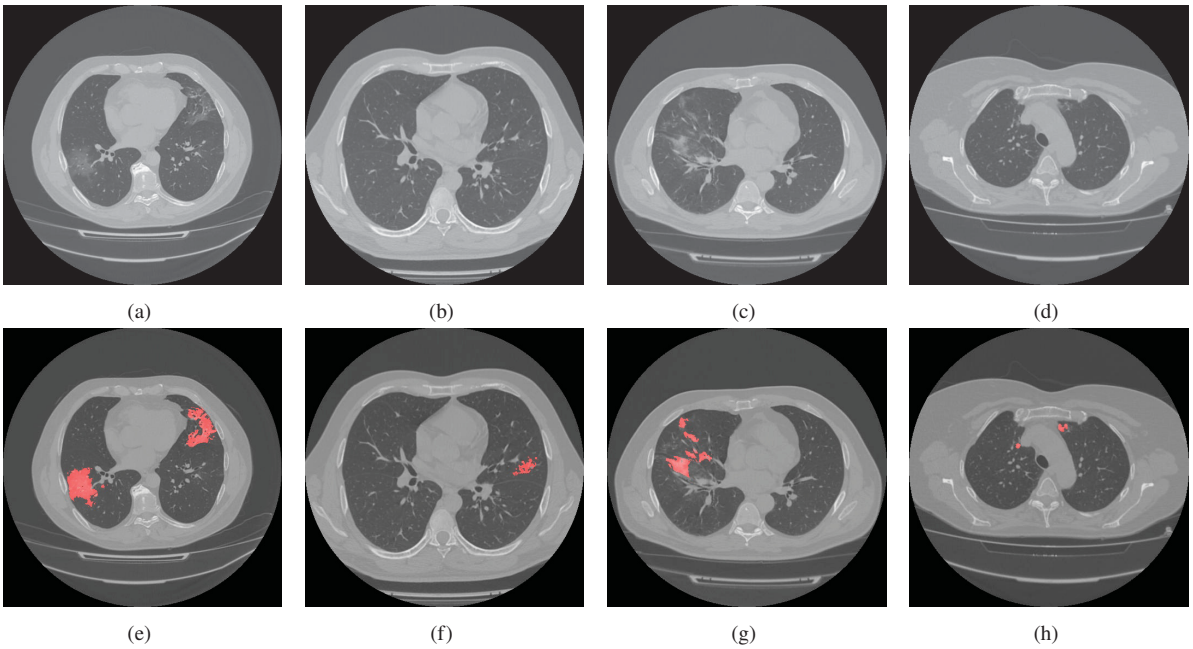


Figure 5.3: Examples of images and segmentation masks from MosMed dataset. The GGO-Consolidation is presented in red, and the regions without any color are the Background regions. Source: (Morozov et al., 2020).

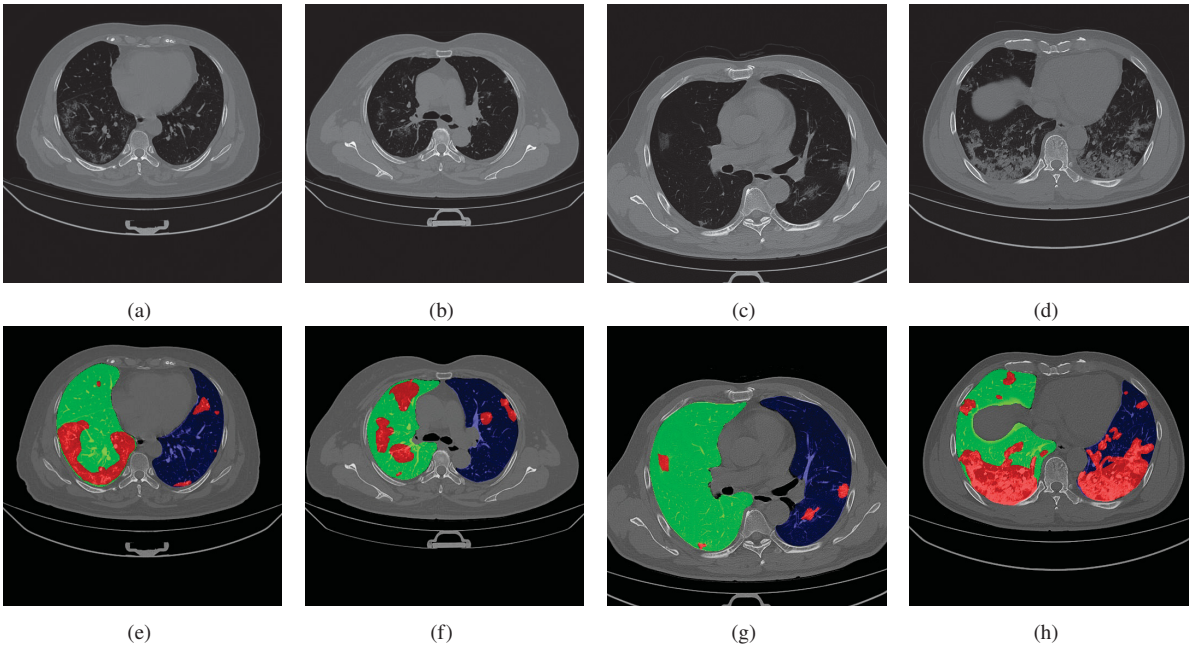


Figure 5.4: Examples of images and segmentation masks from Zenodo dataset. The Left Lung is presented in blue, the Right Lung is green, and the Infections are red. The regions without any color are the Background regions. Source: (Jun et al., 2020; Ma et al., 2021).

dataset presents the following mean proportion for each class: Background (0.89893), Left Lung (0.04331), Right Lung (0.04923), and Infections (0.00852).

5.1.5 Ricord1a

The Ricord (Tsai et al., 2020) dataset is divided into three sets: 1a, 1b, and 1c. The set 1a is the only one with segmentation masks, the set 1b has Computed Tomography (CT)-Scans with

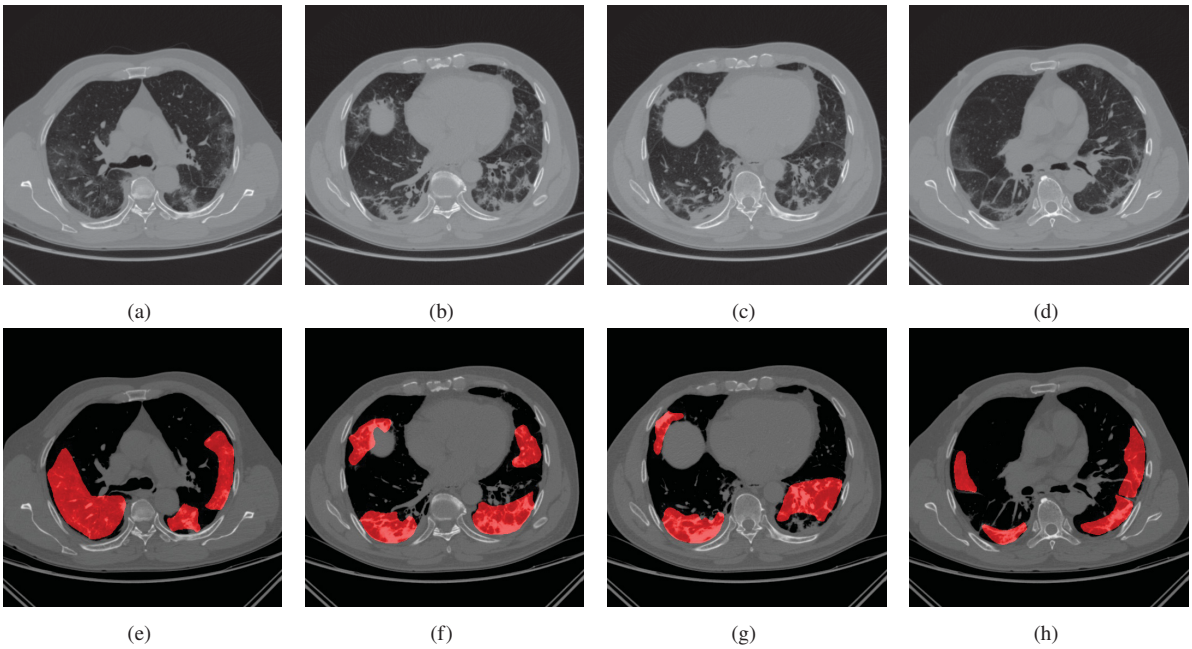


Figure 5.5: Examples of images and segmentation masks from Ricord1a dataset. The Infections are presented in red, and the regions without any color are the Background regions. Source: (Tsai et al., 2020).

negative diagnostics for COVID-19, and the set 1c has CT-Scans with positive diagnostics for COVID-19 . The Ricord1a is the largest segmentation dataset, composed of 9,166 images, with labels for two classes: Background and Infections. Each image has a size of 512×512 pixels. Figure 5.5 presents examples of images and segmentation masks from Ricord1a dataset. The Infections are presented in red, and the regions without any color are the Background regions. The Ricord1a has the following mean proportion: Background (0.95295) and Infections (0.04704).

5.1.6 Ricord1b

The Ricord1b is another subset of the Ricord (Tsai et al., 2020) dataset. This dataset has 21,220 CT-Scans with negative diagnostics for COVID-19. Each image has a size of 512×512 pixels. Figure 5.6 presents examples of images in this dataset.

5.2 EVALUATION METRICS

5.2.1 Semantic Segmentation Metrics

To evaluate the performance of Deep Neural Networks in COVID-19 CT-Scan semantic segmentation, commonly used metrics are the F-score (equation 5.2) and Intersection over Union (IoU) (equation 5.3) are employed. These metrics are widely used in Semantic Segmentation (Minaee et al., 2021). Additionally, during the training process, the segmentation models were optimized using a weighted mean loss function (equation 5.1), with weights w_1 and w_2 set to 1. The loss function was defined as the weighted average of both F-score and IoU to enhance the performance of both metrics. The F-score, Precision, and Recall are commonly used metrics for evaluating binary classification problems, particularly in cases where the number of samples of one category is significantly greater than the other (Pedregosa et al., 2011). Since the datasets employed in this study involved more than two classes, the F-score and IoU were computed instead. For each image, the F-score and IoU were calculated as the mean of the F-score and IoU across all classes. The F-measure is computed as the harmonic average of Precision and Recall, which are

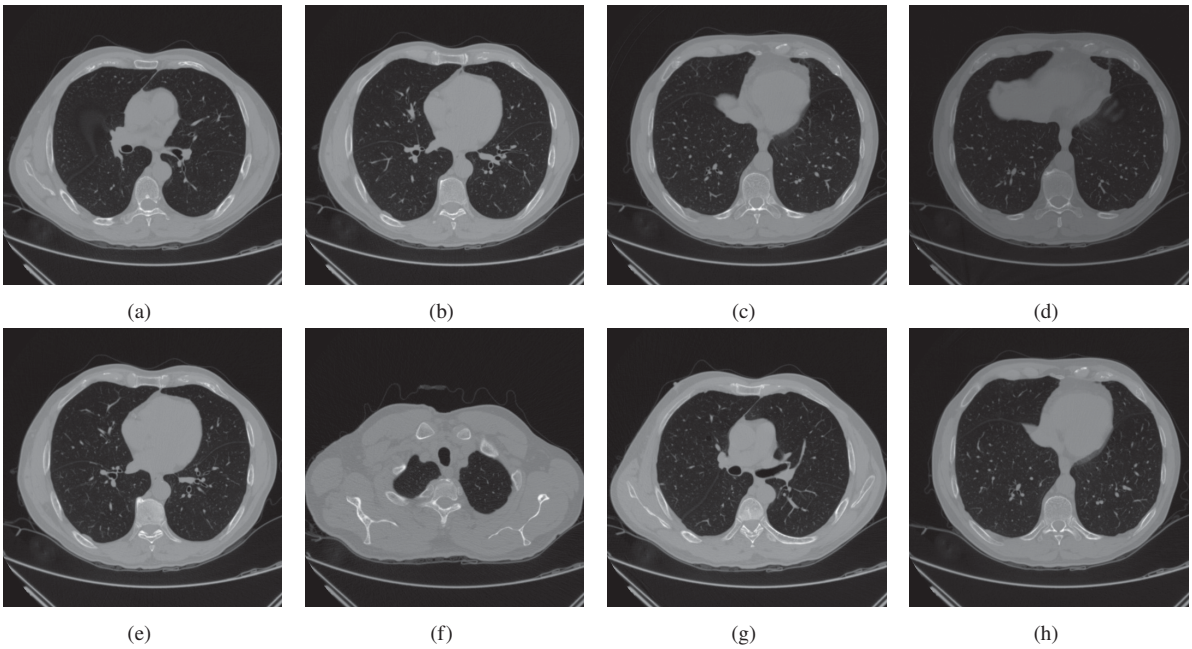


Figure 5.6: Examples of images from Ricord1b dataset. Source: (Tsai et al., 2020).

metrics used to evaluate the performance of binary classification models. Precision measures the proportion of positive predictions, while Recall measures the proportion of actual positive instances correctly classified as positive (Powers, 2008). True Positive refers to the number of positive instances correctly classified as positive, False Positive refers to the number of negative instances wrongly classified as positive, and False Negative refers to the number of positive instances wrongly classified as negative.

$$Loss = 1 - \frac{w_1 * F - score + w_2 * IoU}{2} \tag{5.1}$$

where:

$$F - score = \frac{TruePositive}{TruePositive + \frac{FalsePositive+FalseNegative}{2}} \tag{5.2}$$

and

$$IoU = \frac{Target \cap Prediction}{Target \cup Prediction} \tag{5.3}$$

The IoU is a widely used metric to evaluate Semantic Segmentation methods (Jaccard, 1912; Minaee et al., 2021). It involves the predicted object's area and the target object's area and two values are calculated: the intersection and the union of these areas. The intersection is the overlapping area between the predicted and target objects, while the union is the sum of both areas. The IoU is then calculated as the intersection value divided by the union value (Minaee et al., 2021). The segmentation methods aim to generate segmentation masks with predicted object areas as close to the target area as possible. The calculation of the IoU is illustrated in Figure 5.7, where the area of overlap is represented by the blue-colored square between the two white-colored squares, and the union area is all blue-colored areas. Therefore, the higher the intersection area between the predicted and target objects, the better the segmentation method.

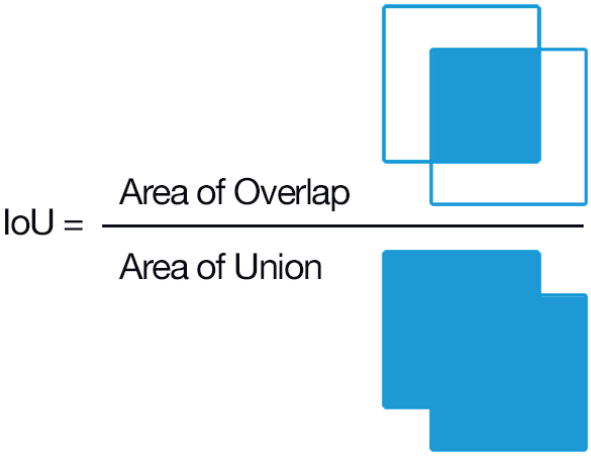


Figure 5.7: Illustration of the IoU calculation. The area of overlap is the blue-colored square between the two white-colored squares, and the area of union is all blue-colored areas. Source: (Jaccard index, 2021).

5.2.2 Generative Adversarial Networks Metrics

The Fréchet Inception Distance (FID) (Heusel et al., 2017) is employed to assess the performance of the GANs architectures. This metric measures the similarity between the images generated by the GAN and the real images used to train the discriminator. The FID uses one of the deeper layers of the Inception V3 (Szegedy et al., 2016) network to compare the mean and standard deviation of the distributions. The FID is calculated as presented in equation 5.4.

$$FID = ||\mu_X - \mu_Y||^2 - Tr\left(\sum_X + \sum_Y - 2\sqrt{\sum_X \sum_Y}\right) \tag{5.4}$$

The X and Y are the real and fake vectors extracted from the Inception model. μ_X and μ_Y are the magnitudes of X and Y. Tr is the trace matrix, and $\sum X$ and $\sum Y$ are the covariance matrix.

5.3 STATISTICAL METHODS

5.3.1 Wilcoxon Signed-Rank Test

The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test for comparing two distributions and assessing the performance difference between two classifiers (Demšar, 2006). One significant advantage of this test is its ability to handle distributions that are not necessarily Gaussian. The Wilcoxon test is computed as follows:

- Let $x = [x_0, x_1, ..., x_n]$ and $y = [y_0, y_1, ..., y_n]$ be two random distribution that will be compared;
- The null hypothesis of the Wilcoxon signed-rank says that the median of the differences between x and y is equal to zero;
- The alternative hypothesis of the Wilcoxon signed-rank says that the median of the differences between x and y is different of zero;
- Compute the absolute difference d between x and y, $d = [|x_0 - y_0|, |x_1 - y_1|, ..., |x_n - y_n|]$;

- Compute the sign s of the difference between x and y, $s = [sgn(x_0 - y_0), sgn(x_1 - y_1), ..., sgn(x_n - y_n)]$;
- Discard from d and s the differences equal to zero;
- Order d in ascending order;
- For each value of d, assign a rank $r = 1, 2, 3, ..., n$. Repeated absolute values receive the average of the ranks they cover;
- Each rank r_i of d_i receives a sign s_i;
- Calculate R^+ as the sum of the positive ranks and R^- as the sum of the negative ranks, both in absolute values;

$$R^+ = |\sum_{i=0}^{n} r_i|, r_i > 0 \tag{5.5}$$

$$R^- = |\sum_{i=0}^{n} r_i|, r_i < 0 \tag{5.6}$$

- Take the smaller value between R^+ and R^-, $T = min(R^+, R^-)$;
- Calculates a z-score as follows:

$$z = \frac{T - \frac{1}{4}n(n+1)}{\sqrt{\frac{1}{24}n(n+1)(2n+1)}} \tag{5.7}$$

The implementation of the Wilcoxon signed-rank test in this study utilized the Scipy library (Virtanen et al., 2020). Scipy is an open-source Python library for scientific, mathematical, and engineering computations. The Scipy implementation of the Wilcoxon test also accommodates various null hypotheses, enabling evaluation of whether the median differences between x and y are negative ($x < y$) or positive ($x > y$).

5.3.2 Friedman Test

Like the Wilcoxon signed-rank test, the Friedman test is a non-parametric statistical hypothesis test for comparing multiple distributions (Demšar, 2006) that does not assume that the compared distributions follow a Gaussian distribution. In this study, the Friedman test was applied to the experiments using the implementation provided by the Scipy library (Virtanen et al., 2020). The calculation of the Friedman test is as follows:

- The measurements must be organized in a matrix of n rows (the size of the distributions) and k the columns (the number of distributions);
- Rank each row (the best value receives rank 1, the second-best value receive rank 2, ...) and replace the values in the matrix with the ranks (repeated ranks are replaced by the average rank);
- Find the average rank of each column as follows:

$$R_j = \frac{1}{n}\sum_{i}^{n} r_{ij} \tag{5.8}$$

where i is the i-th row and j is the j-th column;

- Calculate the test statistic as follows:

$$Q = \frac{12n}{k(k+1)} \sum_{j=1}^{k} (R_j - \frac{k+1}{2})^2 \tag{5.9}$$

5.3.3 P-value

The statistical significance is determined using the p-value, which represents the probability of obtaining the observed statistical value when the null hypothesis is true. This study defines a significance level of $\alpha = 0.05$, which is commonly used (Demšar, 2006). If the p-value is greater than α, the null hypothesis can not be rejected, whereas if the p-value is less than α, the null hypothesis is rejected.

5.4 FINAL CONSIDERATIONS

This chapter presented the datasets, evaluation metrics, and statistical methods used to perform experiments and evaluate the segmentation and data augmentation methods in this work. The following chapter presents the experiments and evaluations performed in this work.

6 EVALUATION AND EXPERIMENTS

This work can be divided into three main phases, and this chapter presents the experiments and results obtained in each phase of this work. Section 6.1 (first phase) comprehensively analyzes 120 Encoder-Decoder segmentation models applied to COVID-19 Computed Tomography (CT) image segmentation. Next, Section 6.2 (second phase) presents a detailed analysis of 20 traditional data augmentation techniques applied to the COVID-19 CT image segmentation. This Section is divided into Subsection 6.2.1 and Subsection 6.2.2. The techniques were evaluated on each training set individually in the first Subsection, and in the second Subsection, the techniques were evaluated in a unified training set of all datasets.

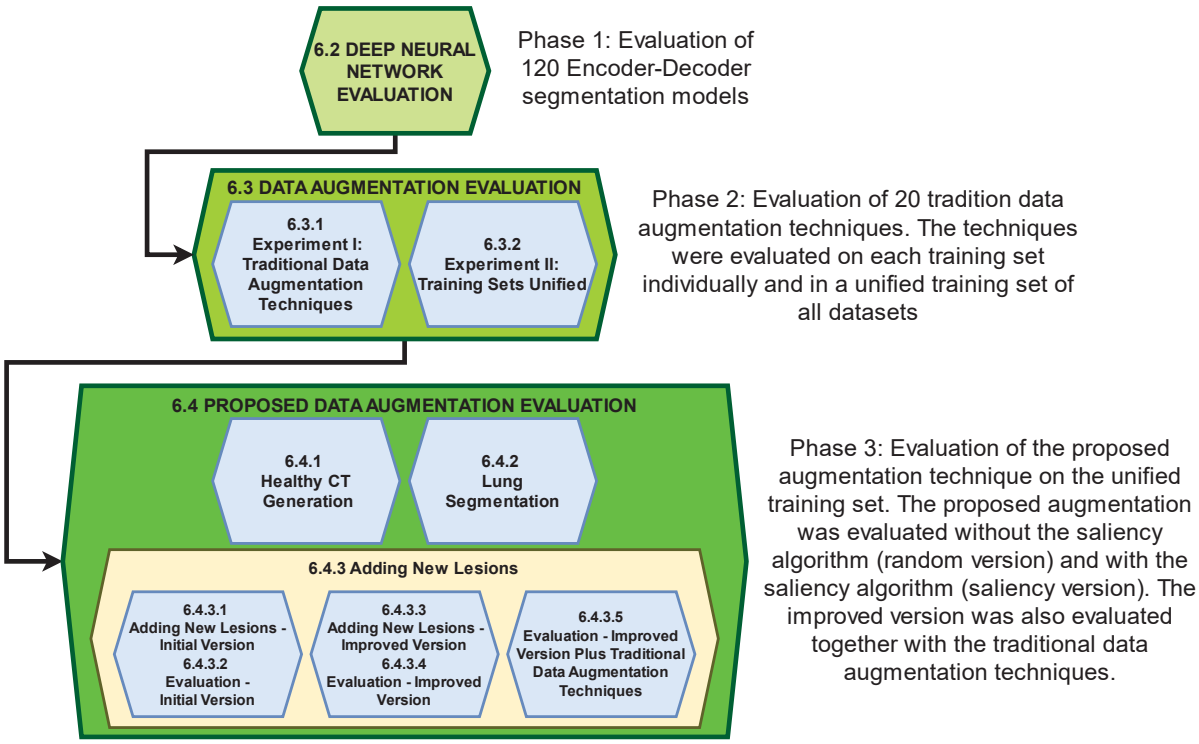


Figure 6.1: This work can be divided into two main phased. Phase 1: Evaluation of 120 Encoder-Decoder segmentation models. Phase 2: Evaluation of 20 tradition data augmentation techniques. Phase 3: Evaluation of the proposed augmentation technique on the unified training set.

Section 6.3 (third phase) presents a step-by-step evaluation of the proposed data augmentation method based on visual salience. Firstly, Subsection 6.3.1 presents the healthy CT generation process. Subsection 6.3.2 presents the segmentation model trained to perform semantic segmentation of the lung areas of the healthy CT images. Then, Subsection 6.3.3 presents the experiments performed in the proposed data augmentation technique.

Two versions of the proposed augmentation were evaluated. The first version uses a random algorithm to place the lesions in the healthy CT images. Subsubsection 6.3.3.1 details the algorithm, and Subsubsection 6.3.3.2 presents the experiments performed with the first version of the proposed augmentation. The second version uses visual salience features to place the lesions in healthy CT images. Subsubsection 6.3.3.3 details the algorithm, and Subsubsection 6.3.3.4 presents the experiments performed with the second version of the proposed augmentation.

Subsubsection 6.3.3.5 presents the evaluation of the second version of the proposed augmentation applied together with the traditional augmentation techniques. Finally, Section 6.4 concludes the chapter by summarizing the main findings of this work. Figure 6.1 illustrates the roadmap of the experiments performed in this work.

6.1 DEEP NEURAL NETWORK EVALUATION

Aiming to perform the segmentation of COVID-19 CTs, many studies apply Deep Learning techniques and Deep Neural Networks, achieving impressive results in the task (Shi et al., 2021). In general, Deep Neural Networks are widely applied in many segmentation problems due to their great generalization capacity, enabling them to learn how to represent different classes of objects in the image (Garcia-Garcia et al., 2018). However, as new approaches are being quickly proposed due to the urgency of the global pandemic, there is a pressing need for a comprehensive evaluation of these methods in the COVID-19 CT segmentation field. Therefore, to provide a standardized comparison of various architectures on multiple recently proposed datasets, the first step of this study involves an extensive benchmark of encoder-decoder models. In this step, twenty encoder models where combined with six decoder models, totaling one hundred twenty models evaluated.

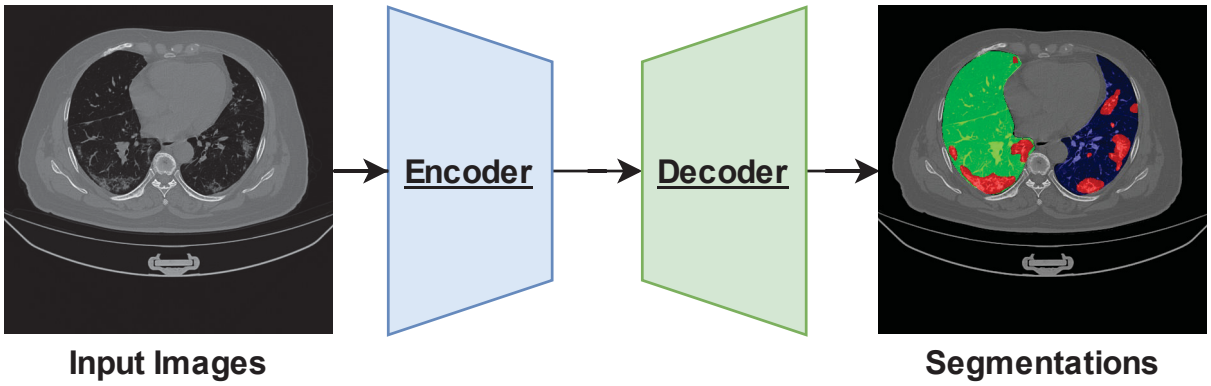


Figure 6.2: The encoder-decoder structure. The images inputs the encoder, the output of the encoder inputs the decoder, and the output of the decoder are the segmentation masks.

Figure 6.2 presents the encoder-decoder structure of the Deep Learning models evaluated in this step. In this step, twenty encoder models where combined with six decoder models, totaling one hundred twenty models. The encoder architectures evaluated in this step are: ResNet-50 and ResNet-101 (He et al., 2016); ResNeXt50 and ResNeXt101 (Xie et al., 2017); Res2Net50 and Res2Net101 (Gao et al., 2021); Visual Geometry Group (VGG)-16 (Simonyan and Zisserman, 2015); DenseNet121, DenseNet169, and DenseNet201 (Huang et al., 2017); Squeeze-and-Excitation Networks (SE-Net)-ResNet50, SE-Net-ResNet101, SE-Net-ResNeXt50, and SE-Net-ResNeXt101 (Hu et al., 2018); RegNetx-002, RegNetx-004, RegNetx-006, RegNety-002, RegNety-004, and RegNety-006 (Radosavovic et al., 2020). And decoders evaluated are: U-Net (Ronneberger et al., 2015), U-Net++ (Zhou et al., 2018), Feature Pyramid Network (FPN) (Lin et al., 2017), Pyramid Scene Parsing Network (PSPNet) (Zhao et al., 2017), LinkNet (Chaurasia and Culurciello, 2017), and Multi-scale Attention Net (MA-Net) (Fan et al., 2020b). Table 6.1 presents a list of the twenty encoders evaluated and each architecture's respective number of parameters in millions of parameters (M). Table 6.2 presents a list of all six decoders evaluated.

In order to evaluate the segmentation models, the PyTorch Segmentation Models (Yakubovskiy, 2020) implementation was utilized. Additionally, a transfer learning (Zhuang

Table 6.1: List of all twenty encoders evaluated and the respective number of parameters of each architecture in millions of parameters (M).

Encoder	Parameters	Encoder	Parameters
ResNet-50	23M	SE-ResNet-50	26M
ResNet-101	42M	SE-ResNet-101	47M
ResNeXt-50 $32 \times 4d$	22M	SE-ResNeXt-50 $32 \times 4d$	25M
ResNeXt-101 $32 \times 8d$	86M	SE-ResNeXt-101 $32 \times 4d$	46M
Res2Net-50 $26w$-$4s$	23M	RegNetx-002	2M
Res2Net-101 $26w$-$4s$	43M	RegNetx-004	4M
VGG-16	14M	RegNetx-006	5M
Densenet-121	6M	RegNety-002	2M
Densenet-169	12M	RegNety-004	3M
Densenet-201	18M	RegNety-006	10M

Table 6.2: List of all six decoders.

Decoders					
U-Net	U-Net++	FPN	PSPNet	LinkNet	MA-Net

et al., 2021) approach was adopted where the pre-trained weights from Imagenet (Deng et al., 2009), available in the library by (Yakubovskiy, 2020), were used to initialize the models. With the transfer learning, the model starts with a foundation of learned features, which enables the models to leverage knowledge gained from pretraining on large datasets and applying it to specific tasks with limited data, such as medical imaging. Additionally, transfer learning can improve model performance by providing a more generalized understanding of data patterns, which enhances its ability to adapt and generalize to new but problems (Zhuang et al., 2021).

The models were trained and evaluated on five distinct CTs datasets: MedSeg (MedSeg, 2021), Zenodo (Jun et al., 2020; Ma et al., 2021), CC-CCII (Zhang et al., 2020), MosMed (Morozov et al., 2020), and Ricord1a (Tsai et al., 2020). In order to perform the training and validation, a standard strategy adopted in the Covid-19 CT segmentation problem is dividing the training set in K-folds, with a 5-fold cross-validation being the most common division adopted (Müller et al., 2021; Saood and Hatem, 2021; Yazdekhasty et al., 2021; Fung et al., 2021; Sun et al., 2022). Following that, the 5-fold cross-validation strategy was applied in this evaluation to avoid the results being attached to a particular training and validation division, resulting in a total of 3,000 experiments.

Furthermore, all datasets were divided into training and test sets following the Pareto principle (Erridge, 2006; Dunford et al., 2014), with the training set being composed of 80% of the images and the test set with 20% of the images. This division is also applied to other segmentation problems (Khan and Borji, 2018; Dmitriev and Kaufman, 2019; Pandey et al., 2020; Li et al., 2019; Habili et al., 2023), including medical segmentation problems (Dong et al., 2018; Chen et al., 2019; Cao et al., 2020; Li et al., 2023; Gite et al., 2022). Then, all networks were trained for 50 epochs in training set with a batch size of 8. The learning rate started at 0.001 and was divided by 10 every 10 epochs. The parameters' values were selected by referencing the available values on the GitHub repository where the network implementations were available (Yakubovskiy, 2020). Additionally, preliminary experiments were conducted to evaluate the networks' training behavior and the training curves when the parameters were changed.

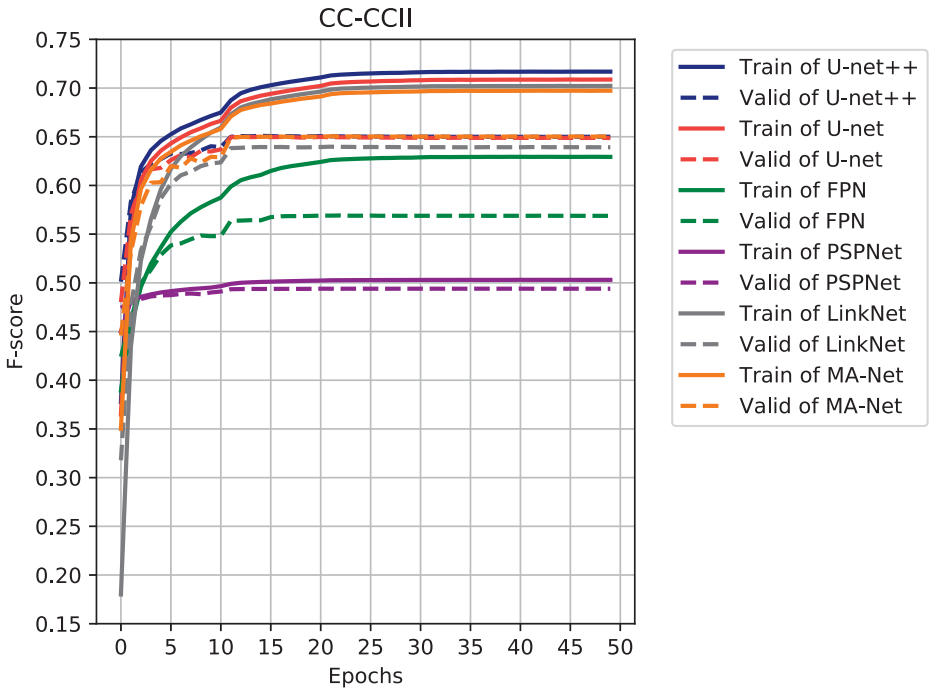


Figure 6.3: Average training and validation curves of the CC-CCII dataset. The curves are color-coded by decoder (U-Net, U-Net++, FPN, PSPNet, LinkNet, and MA-Net) and represents the average F-score of the five-folds of all encoders (ResNet-50-fold0, ..., ResNet-50-fold4, ResNet-101-fold0, ..., ResNet-101-fold4, etc) using that decoder. Source: (Krinski et al., 2021).

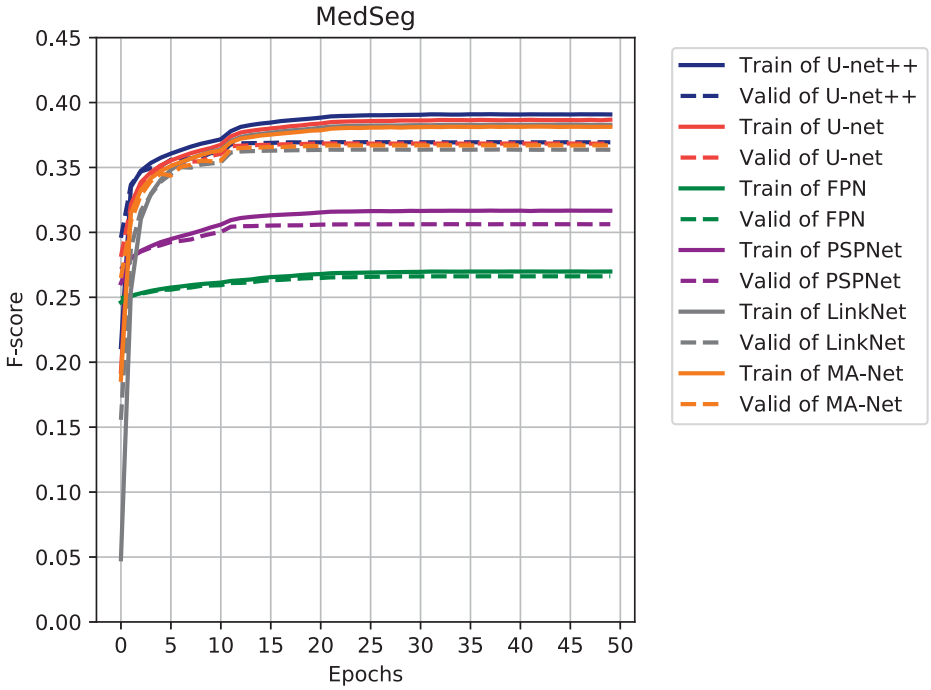


Figure 6.4: Average training and validation curves of the MedSeg dataset. The curves are color-coded by decoder (U-Net, U-Net++, FPN, PSPNet, LinkNet, and MA-Net) and represents the average F-score of the five-folds of all encoders (ResNet-50-fold0, ..., ResNet-50-fold4, ResNet-101-fold0, ..., ResNet-101-fold4, etc) using that decoder. Source: (Krinski et al., 2021).

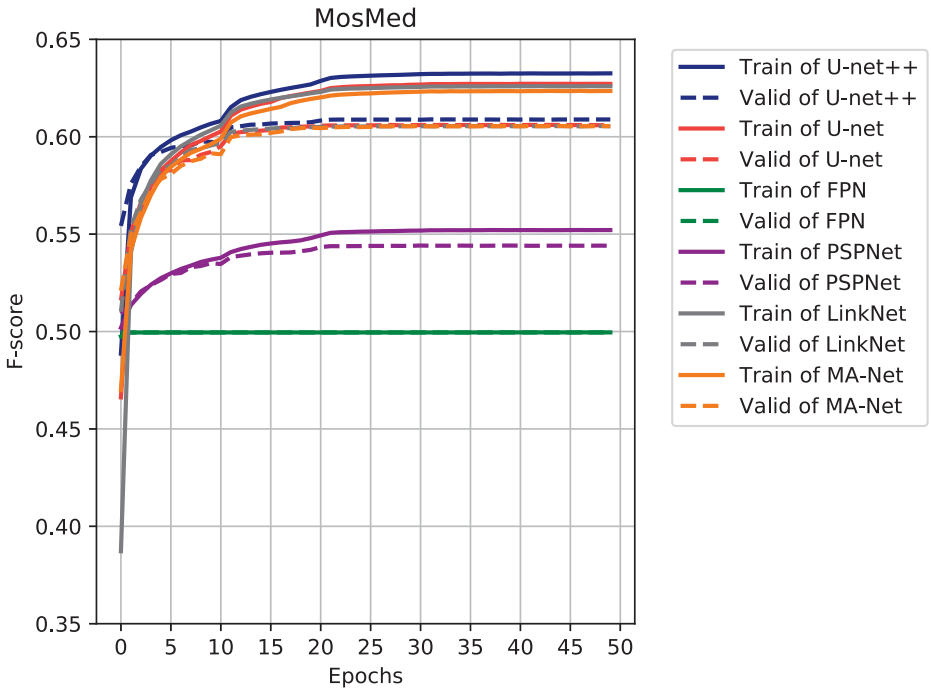


Figure 6.5: Average training and validation curves of the MosMed dataset. The curves are color-coded by decoder (U-Net, U-Net++, FPN, PSPNet, LinkNet, and MA-Net) and represents the average F-score of the five-folds of all encoders (ResNet-50-fold0, ..., ResNet-50-fold4, ResNet-101-fold0, ..., ResNet-101-fold4, etc) using that decoder. Source: (Krinski et al., 2021).

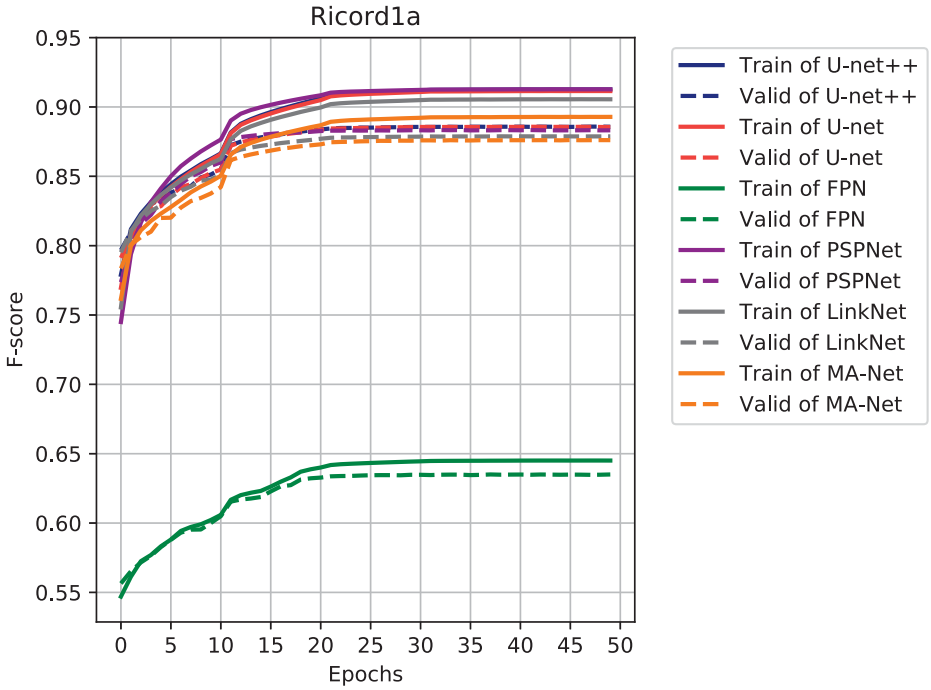


Figure 6.6: Average training and validation curves of the Ricord1a dataset. The curves are color-coded by decoder (U-Net, U-Net++, FPN, PSPNet, LinkNet, and MA-Net) and represents the average F-score of the five-folds of all encoders (ResNet-50-fold0, ..., ResNet-50-fold4, ResNet-101-fold0, ..., ResNet-101-fold4, etc) using that decoder. Source: (Krinski et al., 2021).

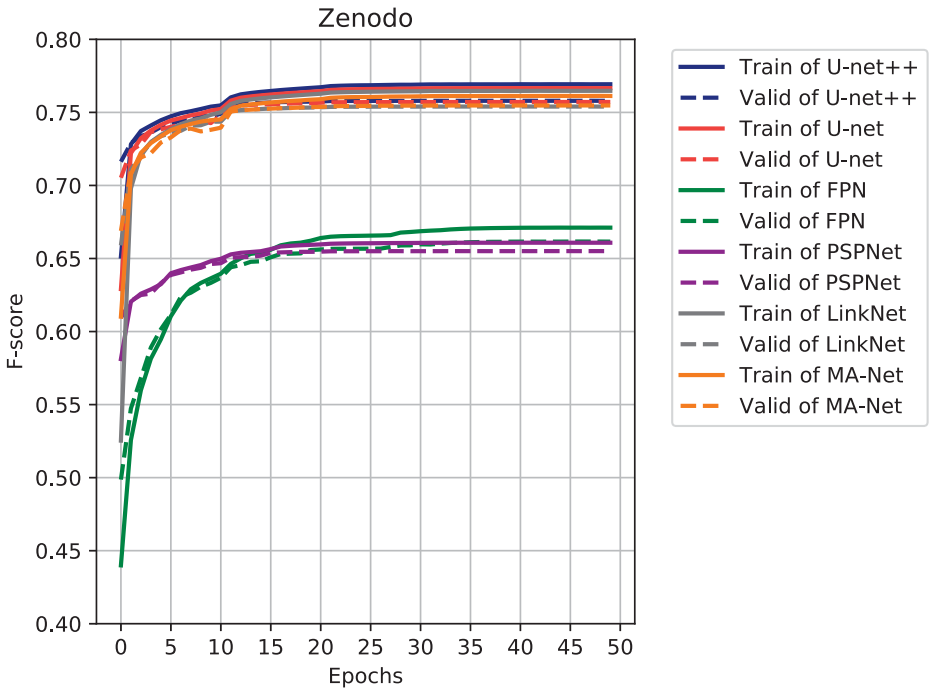


Figure 6.7: Average training and validation curves of Zenodo dataset. The curves are color-coded by decoder (U-Net, U-Net++, FPN, PSPNet, LinkNet, and MA-Net) and represents the average F-score of the five-folds of all encoders (ResNet-50-fold0, ..., ResNet-50-fold4, ResNet-101-fold0, ..., ResNet-101-fold4, etc) using that decoder. Source: (Krinski et al., 2021).

In order to evaluate only the networks, no data augmentation was applied and the input images were resized to 256×256. To minimize distortion on resizing the images, the scaling factor was based on the largest dimension of the image, and the smallest size was padded with zeros accordingly. Figures 6.3, 6.4, 6.5, 6.6, and 6.7 presents training and validation curves generated by the six encoders on each dataset. The curves are color-coded by decoder (U-Net, U-Net++, FPN, PSPNet, LinkNet, and MA-Net) and represents the average F-score of the five-folds of all encoders (ResNet-50-fold0, ..., ResNet-50-fold4, ResNet-101-fold0, ..., ResNet-101-fold4, etc) attached to that decoder.

The U-Net++, U-Net, LinkNet, and MA-Net architectures demonstrated similar performance, with U-Net++ achieving the best results in most datasets. FPN and PSPNet achieved the lowest results overall, although PSPNet achieved very similar results to U-Net++ in the Ricord1a dataset. Across all models, generalization was strong, with the validation curve closely tracking the training curve in most datasets. However, the CC-CCII dataset presented a challenge to generalization, with most models showing a significant gap between the validation and training curves. The only network that performed well on this dataset was PSPNet. Moreover, the FPN could not capable of learning the Ground Glass Opacity (GGO)-Consolidation label in the MosMed dataset. As presented in Figure 6.5, both train and validation curves are straight lines with an F-score of 50%. The FPN achieved an F-score of 99% for background and 0% for lesion class. Such behavior happened due to the critical class imbalance present in this dataset (Background (0.99810) and GGO-Consolidation (0.00189)). The number of classes in this dataset (two classes) also collaborated with this result. In the Ricord1a, another dataset with two classes, the FPN achieved results very far from the other networks.

Table 6.3: The average value of the five-folds cross-validation strategy for test set evaluated with the last train weight. The best F-score values are highlighted in blue, and the best Intersection over Union (IoU) values are highlighted in red. Source: (Krinski et al., 2021).

Decoder	Encoder	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
U-Net++	ResNet-50	0.6460	0.6068	0.3732	0.3475	0.6135	0.5880	0.8802	0.8185	0.7351	0.7065
	ResNet-101	0.6440	0.6043	0.3719	0.3464	0.6114	0.5859	0.8697	0.8061	0.7338	0.7052
	ResNeXt50_32x4d	0.6462	0.6065	0.3740	0.3479	0.6115	0.5861	0.8789	0.8172	0.7352	0.7069
	ResNeXt101_32x8d	0.6477	0.6078	0.3721	0.3464	0.6103	0.5847	0.8721	0.8092	0.7344	0.7058
	Res2Net-50_26w_4s	0.6467	0.6076	0.3744	0.3485	0.6115	0.5863	0.8776	0.8158	0.7353	0.7071
	Res2Net-101_26w_4s	0.6458	0.6061	0.3733	0.3474	0.6132	0.5876	0.8790	0.8176	0.7350	0.7066
	VGG-16	0.6482	0.6082	0.3717	0.3459	0.6160	0.5913	0.8686	0.8053	0.7357	0.7075
	Densenet-121	**0.6516**	**0.6121**	0.3756	0.3494	0.6117	0.5857	0.8873	0.8269	0.7358	0.7077
	Densenet-169	0.6505	0.6108	0.3750	0.3490	0.6123	0.5862	0.8836	0.8228	0.7354	0.7070
	Densenet-201	0.6474	0.6074	0.3743	0.3480	0.6124	0.5868	0.8814	0.8207	0.7359	0.7078
	SE-ResNet-50	0.6482	0.6076	0.3767	0.3510	**0.6219**	**0.5969**	0.8908	0.8308	0.7360	0.7078
	SE-ResNet-101	0.6457	0.6057	**0.3770**	**0.3512**	0.6205	0.5955	0.8910	0.8311	0.7358	0.7077
	SE-ResNeXt50_32x4d	0.6479	0.6076	0.3760	0.3505	0.6168	0.5927	0.8963	0.8377	**0.7367**	**0.7091**
	SE-ResNeXt101_32x4d	0.6440	0.6042	0.3763	0.3509	0.6192	0.5947	0.8963	0.8378	0.7366	0.7088
	RegNetx-002	0.6414	0.6008	0.3744	0.3486	0.6112	0.5862	0.8954	0.8369	0.7360	0.7079
	RegNetx-004	0.6384	0.5981	0.3727	0.3468	0.6127	0.5878	0.8909	0.8317	0.7364	0.7085
	RegNetx-006	0.6386	0.5991	0.3742	0.3484	0.6108	0.5860	0.8940	0.8354	0.7352	0.7069
	RegNety-002	0.6402	0.6001	0.3743	0.3486	0.6134	0.5885	0.8954	0.8368	0.7364	0.7084
	RegNety-004	0.6402	0.6007	0.3710	0.3455	0.6162	0.5907	**0.8983**	**0.8407**	0.7364	0.7086
	RegNety-006	0.6400	0.6001	0.3735	0.3476	0.6146	0.5895	0.8962	0.8378	0.7358	0.7076
U-Net	ResNet-50	**0.6512**	**0.6105**	0.3739	0.3478	0.6123	0.5868	0.8744	0.8115	0.7342	0.7052
	ResNet-101	0.6477	0.6076	0.3725	0.3462	0.6095	0.5841	0.8719	0.8087	0.7333	0.7041
	ResNeXt50_32x4d	0.6452	0.6054	0.3732	0.3474	0.6121	0.5869	0.8734	0.8108	0.7345	0.7057
	ResNeXt101_32x8d	0.6466	0.6065	0.3712	0.3452	0.6140	0.5884	0.8700	0.8068	0.7339	0.7050
	Res2Net-50_26w_4s	0.6479	0.6079	0.3727	0.3468	0.6131	0.5874	0.8755	0.8134	0.7342	0.7053
	Res2Net-101_26w_4s	0.6492	0.6086	0.3727	0.3468	0.6130	0.5876	0.8730	0.8100	0.7341	0.7051
	VGG-16	0.6428	0.6031	0.3704	0.3443	0.6156	0.5913	0.8672	0.8035	0.7351	0.7068
	Densenet-121	0.6483	0.6092	0.3726	0.3464	0.6127	0.5864	0.8835	0.8226	0.7346	0.7058
	Densenet-169	0.6489	0.6095	0.3719	0.3459	0.5899	0.5688	0.8850	0.8241	0.7343	0.7055
	Densenet-201	0.6475	0.6075	0.3721	0.3460	0.5906	0.5694	0.8794	0.8177	0.7346	0.7058
	SE-ResNet-50	0.6454	0.6057	0.3765	0.3508	0.6208	0.5958	0.8910	0.8312	0.7354	0.7070
	SE-ResNet-101	0.6493	0.6096	0.3760	0.3507	**0.6209**	**0.5960**	0.8922	0.8326	0.7357	0.7074
	SE-ResNeXt50_32x4d	0.6462	0.6058	0.3757	0.3502	0.6182	0.5941	0.8953	0.8365	0.7354	0.7072
	SE-ResNeXt101_32x4d	0.6470	0.6070	**0.3772**	**0.3516**	0.6181	0.5939	0.8961	0.8378	**0.7362**	**0.7083**
	RegNetx-002	0.6360	0.5925	0.3707	0.3447	0.6138	0.5889	**0.9042**	**0.8482**	0.7354	0.7070
	RegNetx-004	0.6369	0.5947	0.3707	0.3445	0.6107	0.5859	0.8986	0.8417	0.7344	0.7056
	RegNetx-006	0.6337	0.5937	0.3707	0.3448	0.6108	0.5862	0.8975	0.8399	0.7343	0.7055
	RegNety-002	0.6402	0.5973	0.3712	0.3452	0.6120	0.5869	0.9028	0.8461	0.7358	0.7075
	RegNety-004	0.6370	0.5963	0.3717	0.3458	0.6107	0.5865	0.8968	0.8389	0.7355	0.7071
	RegNety-006	0.6333	0.5940	0.3705	0.3448	0.6116	0.5868	0.8893	0.8296	0.7348	0.7061
FPN	ResNet-50	0.5505	0.5150	0.2478	0.2458	0.4995	0.4990	0.7282	0.6873	0.7314	0.7012
	ResNet-101	0.6313	0.5902	0.2677	0.2611	0.4995	0.4990	0.5621	0.5403	0.7303	0.6997
	ResNeXt50_32x4d	0.4737	0.4434	0.2705	0.2636	0.4995	0.4990	0.5664	0.5453	0.5507	0.5240
	ResNeXt101_32x8d	0.6329	0.5915	0.2692	0.2624	0.4995	0.4990	0.6458	0.6146	0.6310	0.6043
	Res2Net-50_26w_4s	0.6341	0.5931	**0.3177**	**0.3013**	0.4995	0.4990	0.8032	0.7515	0.7293	0.6981
	Res2Net-101_26w_4s	0.6338	0.5925	0.2765	0.2675	0.4995	0.4990	0.6343	0.6016	0.7313	0.7011
	VGG-16	0.4931	0.4546	0.2478	0.2458	0.4995	0.4990	0.4882	0.4776	0.2355	0.2241
	Densenet-121	**0.6381**	**0.5968**	0.2907	0.2792	0.4995	0.4990	0.8844	0.8231	0.7331	0.7035
	Densenet-169	0.6349	0.5942	0.2722	0.2652	0.4995	0.4990	**0.8935**	**0.8342**	0.7324	0.7028
	Densenet-201	0.6374	0.5963	0.2715	0.2644	**0.4997**	**0.4991**	0.8928	0.8337	0.7324	0.7026
	SE-ResNet-50	0.6306	0.5896	0.3176	0.3012	0.4995	0.4990	0.6524	0.6226	0.6921	0.6622
	SE-ResNet-101	0.6319	0.5902	0.2961	0.2845	0.4995	0.4990	0.6433	0.6116	0.6922	0.6624
	SE-ResNeXt50_32x4d	0.6362	0.5948	0.2478	0.2458	0.4995	0.4990	0.5705	0.5504	0.7325	0.7028
	SE-ResNeXt101_32x4d	0.4752	0.4445	0.2478	0.2458	0.4995	0.4990	0.5720	0.5525	**0.7334**	**0.7038**
	RegNetx-002	0.5380	0.5014	0.2478	0.2458	0.4995	0.4990	0.4882	0.4776	0.3639	0.3381
	RegNetx-004	0.4633	0.4324	0.2478	0.2458	0.4995	0.4990	0.5713	0.5517	0.6309	0.6038
	RegNetx-006	0.3881	0.3632	0.2478	0.2458	0.4995	0.4990	0.5711	0.5514	0.6320	0.6055
	RegNety-002	0.6152	0.5727	0.2712	0.2643	0.4995	0.4990	0.4882	0.4776	0.6834	0.6525
	RegNety-004	0.6212	0.5794	0.2478	0.2458	0.4995	0.4990	0.4882	0.4776	0.6316	0.6049
	RegNety-006	0.3117	0.2922	0.2478	0.2458	0.4995	0.4990	0.5650	0.5436	0.5331	0.5104

Table 6.4: The average value of the five-fold cross-validation strategy for test set evaluated with the last train weight. The best F-score values are highlighted in blue, and the best IoU values are highlighted in red (Continuation of Table 6.3). Source: (Krinski et al., 2021).

Decoder	Encoder	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
PSPNet	ResNet-50	0.4822	0.4698	0.3598	0.3336	0.5201	0.5145	0.8881	0.8276	0.6192	0.5993
	ResNet-101	0.4817	0.4692	0.2942	0.2814	0.5950	0.5704	0.8858	0.8247	0.6386	0.6156
	ResNeXt50_32x4d	0.5366	0.5122	0.3161	0.2996	0.5846	0.5634	0.8917	0.8322	0.6614	0.6364
	ResNeXt101_32x8d	0.4823	0.4701	0.3616	0.3354	0.5631	0.5471	0.8901	0.8303	0.6406	0.6183
	Res2Net-50_26w_4s	0.5044	0.4872	0.3119	0.2958	0.5832	0.5623	0.8910	0.8313	0.6412	0.6190
	Res2Net-101_26w_4s	0.4822	0.4700	0.3337	0.3129	0.5620	0.5461	0.8908	0.8310	0.6612	0.6360
	VGG-16	0.4816	0.4687	0.3394	0.3159	0.5803	0.5592	0.8793	0.8171	**0.7001**	**0.6690**
	Densenet-121	0.4992	0.4831	0.2843	0.2743	0.5632	0.5470	0.8940	0.8350	0.6200	0.6004
	Densenet-169	0.4995	0.4832	**0.3621**	**0.3358**	0.5520	0.5375	0.8930	0.8337	0.6198	0.6002
	Densenet-201	0.4933	0.4788	0.3162	0.2997	0.5564	0.5414	0.8920	0.8324	0.6198	0.6001
	SE-ResNet-50	**0.5420**	**0.5153**	0.3620	**0.3358**	0.5831	0.5620	0.8925	0.8329	0.6411	0.6190
	SE-ResNet-101	0.5245	0.5020	0.3443	0.3215	0.5550	0.5404	0.8913	0.8315	0.6621	0.6372
	SE-ResNeXt50_32x4d	0.4832	0.4708	0.3338	0.3130	0.5428	0.5318	0.8941	0.8355	0.6623	0.6378
	SE-ResNeXt101_32x4d	0.4826	0.4702	0.3549	0.3305	**0.6080**	**0.5815**	**0.8957**	**0.8369**	0.6197	0.6003
	RegNetx-002	0.4756	0.4589	0.2478	0.2458	0.4995	0.4990	0.8418	0.7731	0.6470	0.6162
	RegNetx-004	0.4783	0.4628	0.2478	0.2458	0.4995	0.4990	0.8601	0.7944	0.5197	0.4970
	RegNetx-006	0.4800	0.4659	0.2478	0.2458	0.4995	0.4990	0.8747	0.8118	0.6364	0.6122
	RegNety-002	0.4764	0.4596	0.2478	0.2458	0.4995	0.4990	0.8451	0.7769	0.6288	0.6013
	RegNety-004	0.4799	0.4657	0.2478	0.2458	0.4995	0.4990	0.8764	0.8136	0.6170	0.5959
	RegNety-006	0.4808	0.4668	0.2478	0.2458	0.4995	0.4990	0.8788	0.8168	0.6172	0.5963
LinkNet	ResNet-50	0.6432	0.6023	0.3561	0.3288	0.6111	0.5857	0.8612	0.7960	0.7323	0.7031
	ResNet-101	0.6438	0.6030	0.3632	0.3328	0.6072	0.5817	0.8563	0.7903	0.7315	0.7018
	ResNeXt50_32x4d	0.6281	0.5878	0.3688	0.3412	0.6125	0.5868	0.8643	0.7998	0.7339	0.7051
	ResNeXt101_32x8d	0.6455	0.6049	0.3696	0.3432	0.6103	0.5843	0.8643	0.7997	0.7322	0.7029
	Res2Net-50_26w_4s	0.6448	0.6044	0.3666	0.3375	0.6096	0.5838	0.8679	0.8040	0.7323	0.7030
	Res2Net-101_26w_4s	0.6429	0.6023	0.3695	0.3431	0.6085	0.5824	0.8716	0.8078	0.7324	0.7031
	VGG-16	0.6414	0.6024	0.3715	0.3453	0.6136	0.5888	0.8647	0.8002	0.7353	0.7070
	Densenet-121	0.6142	0.5707	0.3708	0.3445	0.6076	0.5816	0.8755	0.8129	0.7331	0.7039
	Densenet-169	0.6275	0.5862	0.3629	0.3314	0.6076	0.5820	0.8730	0.8101	0.7329	0.7038
	Densenet-201	0.6353	0.5941	0.3693	0.3430	0.6056	0.5800	0.8702	0.8065	0.7332	0.7040
	SE-ResNet-50	0.6461	0.6062	0.3743	0.3482	0.6207	0.5959	0.8893	0.8288	0.7096	0.6779
	SE-ResNet-101	0.6455	0.6056	0.3753	0.3496	0.6211	0.5963	0.8895	0.8292	0.7354	0.7070
	SE-ResNeXt50_32x4d	**0.6467**	**0.6064**	0.3757	0.3500	0.6212	0.5964	0.8934	0.8339	**0.7361**	**0.7081**
	SE-ResNeXt101_32x4d	0.6449	0.6054	**0.3761**	**0.3505**	**0.6213**	**0.5965**	0.8944	0.8352	**0.7361**	**0.7081**
	RegNetx-002	0.6135	0.5704	0.3601	0.3333	0.6038	0.5793	0.8913	0.8318	0.7250	0.6926
	RegNetx-004	0.6185	0.5766	0.3671	0.3409	0.6085	0.5836	0.8935	0.8346	0.7328	0.7032
	RegNetx-006	0.6215	0.5810	0.3673	0.3415	0.6088	0.5836	0.8895	0.8296	0.7303	0.6998
	RegNety-002	0.6135	0.5705	0.3659	0.3396	0.6063	0.5808	0.8911	0.8311	0.7314	0.7011
	RegNety-004	0.6316	0.5906	0.3675	0.3411	0.6070	0.5811	**0.8946**	**0.8357**	0.7335	0.7043
	RegNety-006	0.6286	0.5884	0.3687	0.3427	0.6127	0.5869	0.8934	0.8346	0.7343	0.7053
MA-Net	ResNet-50	0.6468	0.6036	0.3682	0.3422	0.6132	0.5880	0.8691	0.8055	0.7310	0.7008
	ResNet-101	0.6448	0.6023	0.3692	0.3430	0.6109	0.5856	0.8660	0.8017	0.7306	0.7002
	ResNeXt50_32x4d	0.6461	0.6047	0.3709	0.3449	0.6133	0.5875	0.8733	0.8104	0.7315	0.7016
	ResNeXt101_32x8d	0.6462	0.6052	0.3712	0.3446	0.6086	0.5831	0.8634	0.7982	0.7313	0.7013
	Res2Net-50_26w_4s	0.6468	0.6051	0.3701	0.3438	0.6122	0.5866	0.8689	0.8053	0.7312	0.7011
	Res2Net-101_26w_4s	0.6475	0.6062	0.3669	0.3406	0.6109	0.5852	0.8741	0.8112	0.7308	0.7004
	VGG-16	0.6452	0.6047	0.3737	0.3474	0.6169	0.5923	0.8789	0.8171	0.7351	**0.7067**
	Densenet-121	0.6470	0.6055	0.3645	0.3381	0.6099	0.5840	0.8565	0.7907	0.7304	0.7002
	Densenet-169	0.6455	0.6039	0.3659	0.3389	0.5797	0.5599	0.8597	0.7944	0.7296	0.6990
	Densenet-201	0.6470	0.6057	0.3647	0.3383	0.6084	0.5823	0.8615	0.7965	0.7301	0.6997
	SE-ResNet-50	**0.6484**	**0.6076**	**0.3759**	**0.3502**	**0.6212**	**0.5960**	0.8819	0.8208	0.7331	0.7038
	SE-ResNet-101	0.6479	0.6075	0.3758	0.3501	0.6206	0.5956	0.8873	0.8265	0.7333	0.7041
	SE-ResNeXt50_32x4d	0.6431	0.6026	0.3751	0.3497	0.6172	0.5928	**0.8918**	0.8321	0.7335	0.7043
	SE-ResNeXt101_32x4d	0.6432	0.6031	0.3755	0.3498	0.6197	0.5948	0.8917	**0.8323**	**0.7352**	**0.7067**
	RegNetx-002	0.6402	0.5983	0.3725	0.3467	0.6130	0.5880	0.8888	0.8288	0.7345	0.7053
	RegNetx-004	0.6392	0.5987	0.3714	0.3458	0.6089	0.5844	0.8874	0.8269	0.7340	0.7048
	RegNetx-006	0.6372	0.5969	0.3721	0.3461	0.6118	0.5873	0.8817	0.8204	0.7332	0.7038
	RegNety-002	0.6440	0.6020	0.3715	0.3457	0.6105	0.5859	0.8825	0.8210	0.7336	0.7043
	RegNety-004	0.6395	0.5988	0.3713	0.3453	0.6109	0.5862	0.8835	0.8223	0.7343	0.7054
	RegNety-006	0.6351	0.5949	0.3710	0.3454	0.6101	0.5852	0.8851	0.8241	0.7330	0.7034

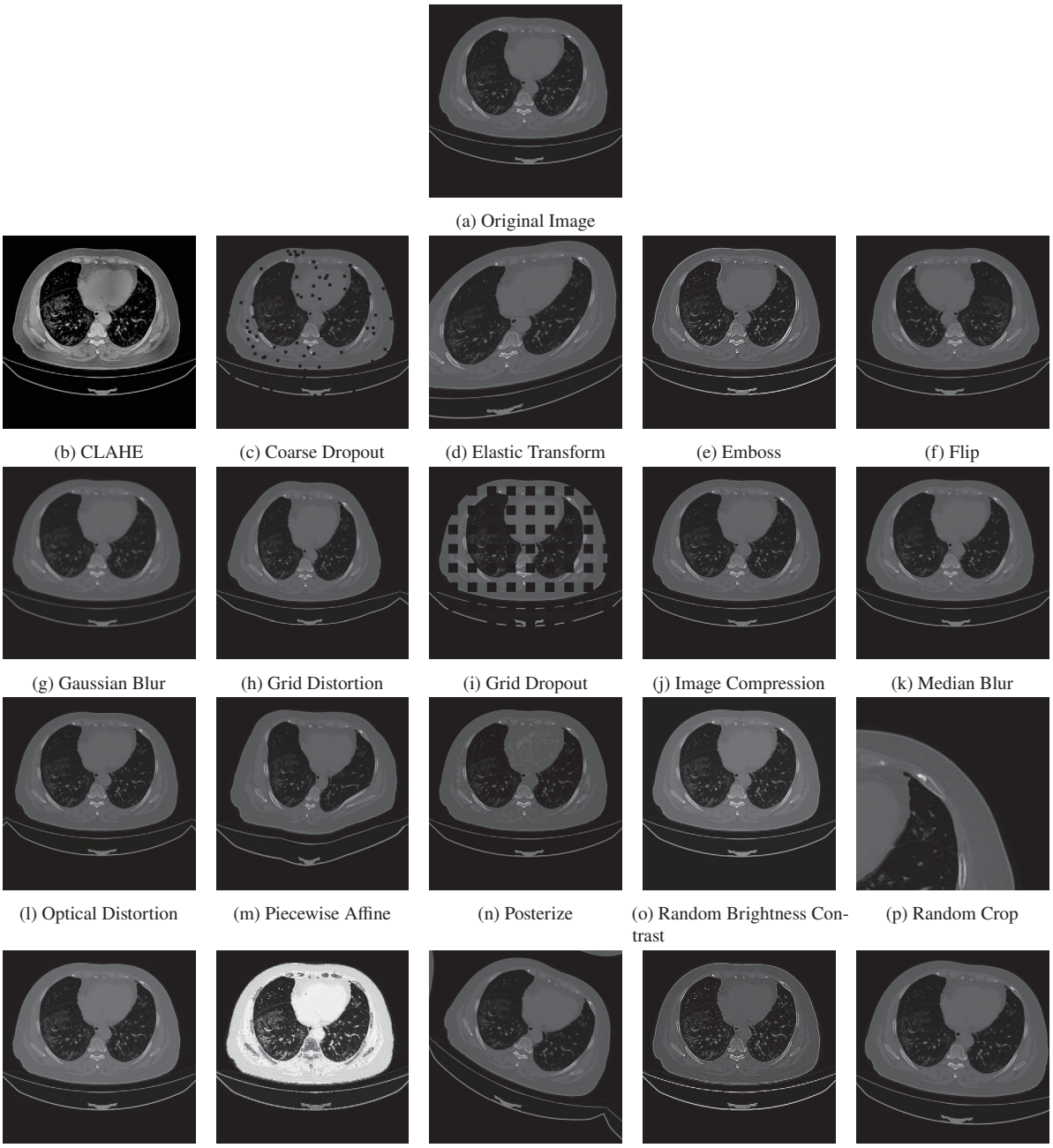
Table 6.3 and Table 6.4 display the results obtained in the test set, calculated as the average of the five-fold cross-validation strategy using the last trained weight. The best F-score values are highlighted in blue, and the best IoU values are highlighted in red. The U-Net, U-Net++, and MA-Net showed more consistent results with similar performance across different encoders. However, the FPN, PSPNet, and LinkNet appeared to be more sensitive to encoder change, exhibiting higher variations in F-score and IoU. The FPN and PSPNet obtained the worst results among the decoders, while the other decoders achieved comparable results. The MedSeg was the most challenging dataset, with all the networks achieving poor results. When compared to other datasets for COVID-19 segmentation, the MedSeg dataset presents some challenges, such as coarse segmentation masks that bypass the lung region and additional markings like circles and arrows pointing to the lesion area. These issues can make the learning process for Deep Neural Networks more difficult.

The Friedman test was utilized to perform statistical analysis on the trained models. This test assumes that the models have the same distribution. Each trained model's F-score value was used to build their respective distribution using the test set. The encoders and decoders were compared, and the Friedman test was first applied to each decoder to determine the statistical difference between the encoders trained with that decoder. Next, the Friedman test was applied to compare the statistical difference between decoders. In both cases, the Friedman test returned a P-value of zero, leading to the null hypothesis being rejected, implying a statistical difference between encoders and decoders. Hence, even though the average F-score and IoU of each model were comparable, as shown in Table 6.3 and Table 6.4, changing the encoders paired with an encoder generates a distinct distribution in the test images, and the encoders have a different distribution in the test images as well.

Despite the large number of models compared, there is no definitive answer for the best encoder-decoder combination to be used in the COVID-19 CT segmentation problem, as expected. However, this study provides, as main contribution, a robust guideline on encoder-decoder selection for future works, setting of variables like the number of network parameters, and number of training epochs. Furthermore, among the decoders, the U-Net and the U-Net++ achieved the highest results, with the LinkNet and MA-Net as close contenders. On the other hand, FPN and PSPNet were found unsuitable for this problem. In terms of encoders, the performance varied depending on the decoder combination and dataset applied. None of the encoders stood out, as they all achieved comparable results. This analysis also revealed some weak points in the datasets, such as the small number of images and critical class imbalance, which can impact the learning process of deep neural networks.

6.2 DATA AUGMENTATION EVALUATION

The next step of this work addresses the issues of small datasets and class imbalance, highlighted in Section 6.1, by evaluating data augmentation techniques. Generally, the impacts of data augmentation techniques are not approached in studies proposed for the COVID-19 CT segmentation problem (Saood and Hatem, 2021; Mahmud et al., 2021a). Generally, studies on COVID-19 CT segmentation have not evaluated data augmentation techniques extensively, with only a few studies utilizing them (Zhao et al., 2021; Qiblawey et al., 2021; Joseph Raj et al., 2021; Müller et al., 2021; Chen et al., 2020c; Xu et al., 2020). Moreover, the data augmentation techniques applied in these studies are limited to Flip and Rotation operations. To correctly measure the impact of data augmentation on the COVID-19 CT segmentation problem, this section presents an evaluation of 20 different data augmentation techniques not based on neural networks: CLAHE, Coarse Dropout, Elastic Transform, Emboss, Flip, Gaussian Blur, Grid Distortion, Grid Dropout,



(a) Original Image

(b) CLAHE (c) Coarse Dropout (d) Elastic Transform (e) Emboss (f) Flip

(g) Gaussian Blur (h) Grid Distortion (i) Grid Dropout (j) Image Compression (k) Median Blur

(l) Optical Distortion (m) Piecewise Affine (n) Posterize (o) Random Brightness Contrast (p) Random Crop

(q) Random Gamma (r) Random Snow (s) Rotate (t) Sharpen (u) Shift Scale Rotate

Figure 6.8: Illustration of the 20 traditional data augmentation techniques (i.e., not based on neural networks) applied to a CT image. In 6.8(a), the original image is presented. The data augmentation techniques are illustrated in the following sequence: Contrast Limited Adaptive Histogram Equalization (CLAHE) 6.8(b), Coarse Dropout 6.8(c), Elastic Transform 6.8(d), Emboss 6.8(e), Flip 6.8(f), Gaussian Blur 6.8(g), Grid Distortion 6.8(h), Grid Dropout 6.8(i), Image Compression 6.8(j), Median Blur 6.8(k), Optical Distortion 6.8(l), Piecewise Affine 6.8(m), Posterize 6.8(n), Random Brightness Contrast (RBC) 6.8(o), Random Crop 6.8(p), Random Gamma 6.8(q), Random Snow 6.8(r), Rotate 6.8(s), Sharpen 6.8(t), and Shift Scale Rotate 6.8(u).

Image Compression, Median Blur, Optical Distortion, Piecewise Affine, Posterize, RBC, Random Crop, Random Gamma, Random Snow, Rotate, Sharpen, Shift Scale Rotate. Figure 6.8 illustrates the 20 data augmentation techniques applied to a CT image. These data augmentation techniques were applied using the Albumentations library (Buslaev et al., 2020), which has been successfully explored in various areas of computer vision (Kupyn et al., 2019; Kaissis et al., 2021; Laroca et al., 2022b). In order to perform this evaluation, the augmentations were applied with the

default parameters of the Albumentations library. Parameter optimization of the best techniques was left for future work.

As mentioned in the previous section, a problem identified was the class imbalance due to several images with just the background class. Moreover, recent works have shown that several applications suffer from class imbalance problem (Johnson and Khoshgoftaar, 2019; Laroca et al., 2021, 2022a; Sanagavarapu et al., 2021). To address this issue, images with ground-truth masks containing only lung pixels or no labeled pixels were eliminated from the datasets. In the CC-CCII dataset, 201 images were removed, while 1,676 images were removed from the Zenodo dataset. Similarly, 457 images were removed from the MedSeg dataset, and 1,264 images were removed from the MosMed dataset. The details of the number of images and the number of images in each dataset are presented in Table 6.5. After these images being removed, the datasets were divided into 80% for training and 20% for testing, following the Pareto principle (Erridge, 2006). Furthermore, to encourage generalization, the original classes of the five datasets were reorganized into two classes: **background** and **lesion**. Everything that was not a type of lesion was converted to the background, and all lesion subtypes were merged into a single class. The training set was subjected to a 5-fold cross-validation strategy for training and validation purposes.

Table 6.5: The segmentation models were trained and evaluated across five different datasets of CT scans: MedSeg (MedSeg, 2021), Zenodo (Jun et al., 2020), CC-CCII (Zhang et al., 2020), MosMed (Morozov et al., 2020) and Ricord1a (Tsai et al., 2020). Source (Krinski et al., 2023).

Dataset	Type	Number of Images	Removed Images	Labels
CC-CCII	Segmentation	750	201	Background, Lung Field, GGO, and Consolidation
MedSeg	Segmentation	929	457	Background, GGO, Consolidation, and Pleural Effusion
MosMed	Segmentation	2,049	1,264	Background and GGO-Consolidation
Ricord1a	Segmentation	9,166	0	Background and Infections
Zenodo	Segmentation	3,520	1,676	Background, Left Lung, Right Lung, and Infections

The encoder-decoder network chosen to evaluate the augmentation techniques was the RegNetx-002 (Radosavovic et al., 2020) encoder and U-net++ (Zhou et al., 2018) decoder. The architecture was trained for up to 100 epochs with the patience of 10 epochs. The initial learning rate was set to 0.001 and was divided by 10 every 10 epochs. The augmentation algorithms were applied using online augmentation, where each batch of images has a probability of being augmented before being fed into the network. The impact of increasing the number of augmented images was also evaluated with ten probabilities of applying data augmentation: 0.05, 0.1, 0.15, 0.2, 0.25, 0.30, 0.35, 0.4, 0.45, and 0.5. Generally, the studies in the literature do not specify the probabilities used for data augmentation. When they do, they rarely provide a rationale for the chosen values. The range selected in this work aims to encompass the probabilities mentioned in the literature (Müller et al., 2020, 2021; Zhang et al., 2022). Additionally, higher probabilities were reserved for future research due to the significant time required to evaluate each probability.

In order to avoid information loss when the image is downscaled, the segmentation network was trained with the original resolution of 512×512 instead of the downscaled 256×256

used in the previous section. In general, downsizing images can lead to information loss, adversely affecting network learning. Furthermore, this evaluation consists of two stages. Firstly, the augmentation techniques were applied to the training set of each dataset separately, generating over 5.000 experiments. The results obtained using this approach are presented in Sub-section 6.2.1. Besides this conventional training methodology, where the training and test sets are separate subsets of the same dataset, a distinct methodology was also evaluated, generating over 1.000 experiments. In this approach, the training subsets are merged into a larger set while the testing procedure remained unchanged. The original classes were converted to **background** and **lesion** to ensure a fair comparison. Refer to Subsection 6.2.2 for more information on this methodology.

6.2.1 Experiment I: Traditional Data Augmentation Techniques

This Subsection presents the evaluation of the segmentation model trained with the augmentation techniques presented in Figure 6.8. Table 6.6 presents the evaluation results for probabilities 0.05 and 0.1, Table 6.7 presents the results for probabilities 0.15 and 0.2, Table 6.8 presents the results for probabilities 0.25 and 0.3, Table 6.9 presents the results for probabilities 0.35 and 0.4, and Table 6.10 presents the results for probabilities 0.45 and 0.5. The Wilcoxon Signed-Rank Test was utilized to perform statistical analysis between the segmentation model trained applying data augmentation technique and training the segmentation model without applying data augmentation. The values highlighted in green show the data augmentation techniques where the P-value achieved values lower than 0.05, and the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation).

Most data augmentation techniques failed to enhance the F-score and the IoU values. At a probability of 0.05, no technique produced an F-score improvement greater than 1% compared to the baseline, where no data augmentation was applied. However, seven methods exhibited statistical differences in Ricord1a, while one technique showed statistical differences in CC-CCII and Zenodo. At a probability of 0.1, Piecewise Affine and Rotate enhanced the F-score in CC-CCII, Shift Scale Rotate improved the F-score in MedSeg, and Grid Distortion improved the F-score in MosMed. Optical Distortion improved the F-score in CC-CCII at a probability of 0.15. At a probability of 0.2, six data augmentation techniques have improved the F-score in CC-CCII, while five methods improved the F-score in MosMed. Elastic Transform improved the F-score in the Zenodo dataset. At a probability of 0.25, three data augmentation techniques produced better F-score values on CC-CCII, and three techniques achieved better F-score values on MosMed. At a probability of 0.3, seven data augmentation techniques improved the F-score value in CC-CCII, five techniques improved the F-score in MosMed, and one technique (Gaussian Blur) improved the F-score in Ricord1a.

At a probability of 0.35, five data augmentation techniques improved the F-score value in CC-CCII, one technique (Elastic Transform) improved the F-score value in MedSeg, and six techniques improved the F-score in MosMed. At a probability of 0.4, five data augmentation techniques improved the F-score value in CC-CCII, and five techniques improved the F-score in MosMed. At a probability of 0.45, six data augmentation techniques improved the F-score value in CC-CCII, five techniques improved the F-score in MosMed, two techniques improved the F-score in Ricord1a, and Grid Distortion improved the F-score in Zenodo. At a probability of 0.5, six data augmentation techniques improved the F-score value in CC-CCII, two data augmentation techniques improved the F-score value in MedSeg and five techniques improved the F-score in MosMed, and Median Blur improved the F-score in Ricord1a. Considering the 10 probabilities evaluated, the probability of 0.5 achieved the best outcomes considering the number of augmentation techniques that improved the F-score values.

Table 6.6: Results of the data augmentation evaluation. *p* stands for probability, the blue-colored values indicate the best F-score values, while the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). Source: (Krinski et al., 2023)

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8450	0.7896	0.8885	0.8261	0.8180	0.7528	0.8875	0.8275	0.9072	0.8490
0.05	CLAHE	0.8364	0.7810	0.8839	0.8211	0.8137	0.7478	0.8871	0.8274	0.9072	0.8488
	Coarse Dropout	0.8463	0.7911	0.8869	0.8247	0.8160	0.7499	0.8872	0.8271	0.9077	0.8498
	Elastic Transform	0.8467	0.7915	0.8886	0.8266	0.8136	0.7482	0.8856	0.8256	0.9073	0.8489
	Emboss	0.8477	0.7916	0.8867	0.8244	0.8162	0.7505	0.8866	0.8268	0.9082	0.8500
	Flip	0.8458	0.7905	0.8869	0.8246	0.8189	**0.7542**	0.8884	0.8281	**0.9087**	**0.8509**
	Gaussian Blur	0.8451	0.7894	0.8856	0.8229	0.8126	0.7467	0.8883	0.8280	0.9086	0.8502
	Grid Distortion	0.8457	0.7897	0.8868	0.8242	**0.8197**	0.7536	0.8855	0.8252	0.9077	0.8494
	Grid Dropout	0.8391	0.7840	0.8870	0.8245	0.8149	0.7499	0.8866	0.8265	0.9069	0.8486
	Image Compression	0.8386	0.7835	0.8868	0.8248	0.8118	0.7465	0.8881	0.8282	0.9074	0.8491
	Median Blur	0.8446	0.7887	0.8849	0.8225	0.8116	0.7455	0.8888	0.8289	0.9078	0.8492
	Optical Distortion	0.8431	0.7884	0.8865	0.8246	0.8134	0.7485	0.8852	0.8250	0.9081	0.8499
	Piecewise Affine	0.8468	0.7914	0.8874	0.8257	0.8155	0.7498	0.8879	0.8279	0.9076	0.8495
	Posterize	0.8455	0.7899	0.8873	0.8252	0.8168	0.7504	**0.8895**	**0.8299**	0.9076	0.8489
	RBC	0.8441	0.7883	0.8872	0.8247	0.8129	0.7474	0.8870	0.8268	0.9084	0.8502
	Random Crop	0.8440	0.7886	0.8853	0.8224	0.8156	0.7500	0.8862	0.8265	0.9067	0.8486
	Random Gamma	0.8431	0.7890	0.8866	0.8244	0.8176	0.7523	0.8889	0.8289	0.9083	0.8499
	Random Snow	0.8432	0.7870	0.8861	0.8237	0.8152	0.7497	0.8863	0.8260	0.9063	0.8476
	Rotate	0.8479	0.7917	0.8885	0.8262	0.8175	0.7523	0.8861	0.8258	0.9084	0.8500
	Sharpen	**0.8481**	**0.7931**	0.8873	0.8249	0.8154	0.7504	0.8882	0.8283	0.9085	0.8502
	Shift Scale Rotate	0.8459	0.7894	**0.8897**	**0.8281**	0.8185	0.7526	0.8860	0.8259	0.9083	0.8500
0.1	CLAHE	0.8372	0.7816	0.8864	0.8240	0.8125	0.7472	0.8874	0.8273	0.9070	0.8486
	Coarse Dropout	0.8477	0.7922	0.8878	0.8259	0.8171	0.7516	0.8874	0.8272	0.9080	0.8499
	Elastic Transform	0.8481	0.7931	0.8875	0.8255	0.8158	0.7503	0.8871	0.8271	**0.9088**	0.8510
	Emboss	0.8436	0.7872	0.8880	0.8260	0.8158	0.7503	0.8885	0.8283	**0.9088**	0.8510
	Flip	0.8483	0.7928	0.8877	0.8260	0.8204	0.7547	0.8861	0.8260	0.9081	0.8495
	Gaussian Blur	0.8436	0.7891	0.8843	0.8213	0.8136	0.7480	0.8874	0.8273	0.9071	0.8487
	Grid Distortion	0.8458	0.7910	0.8889	0.8266	**0.8211**	**0.7564**	0.8864	0.8267	**0.9088**	**0.8511**
	Grid Dropout	0.8422	0.7869	0.8855	0.8229	0.8152	0.7498	0.8840	0.8237	0.9073	0.8484
	Image Compression	0.8490	0.7938	0.8875	0.8255	0.8150	0.7489	0.8869	0.8268	0.9086	0.8504
	Median Blur	0.8460	0.7907	0.8861	0.8235	0.8167	0.7501	0.8888	0.8288	0.9074	0.8491
	Optical Distortion	0.8483	0.7940	0.8893	0.8275	0.8164	0.7500	0.8888	0.8290	0.9080	0.8498
	Piecewise Affine	**0.8505**	**0.7949**	0.8892	0.8276	0.8194	0.7536	0.8871	0.8270	0.9072	0.8488
	Posterize	0.8448	0.7904	0.8862	0.8239	0.8092	0.7435	0.8866	0.8268	0.9077	0.8497
	RBC	0.8415	0.7870	0.8873	0.8249	0.8095	0.7426	0.8871	0.8270	0.9079	0.8495
	Random Crop	0.8375	0.7819	0.8869	0.8248	0.8137	0.7480	0.8877	0.8279	0.9080	0.8499
	Random Gamma	0.8447	0.7901	0.8870	0.8251	0.8118	0.7451	0.8867	0.8268	0.9071	0.8486
	Random Snow	0.8460	0.7906	0.8851	0.8222	0.8114	0.7457	0.8861	0.8258	0.9068	0.8479
	Rotate	0.8503	0.7947	0.8874	0.8256	0.8185	0.7526	0.8848	0.8248	0.9077	0.8491
	Sharpen	0.8492	0.7931	0.8849	0.8224	0.8173	0.7519	**0.8890**	**0.8291**	0.9076	0.8494
	Shift Scale Rotate	0.8491	0.7944	**0.8905**	**0.8287**	0.8186	0.7537	0.8861	0.8260	0.9086	0.8504

To sum up, the best outcome for each dataset was as follows: in CC-CCII, Shift Scale Rotate applied with a probability of 0.45 achieved an F-score of 0.8598; in MedSeg, Elastic Transform applied with a probability of 0.35 resulted in an F-score of 0.8906; in MosMed, Grid Distortion applied with a probability of 0.35 produced an F-score of 0.8307; in Ricord1a, Median Blur applied with a probability of 0.5 obtained an F-score of 0.8904; and in Zenodo, Grid Distortion applied with a probability of 0.45 achieved an F-score of 0.9104. Considering

Table 6.7: Results of the data augmentation evaluation (Continuation of Table 6.6). p stands for probability, the blue-colored values indicate the best F-score values, while the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques where the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). Source: (Krinski et al., 2023)

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8450	0.7896	0.8885	0.8261	0.8180	0.7528	0.8875	0.8275	0.9072	0.8490
0.15	CLAHE	0.8414	0.7863	0.8836	0.8208	0.8117	0.7451	0.8886	0.8288	0.9082	0.8497
	Coarse Dropout	0.8415	0.7869	0.8856	0.8232	0.8118	0.7466	**0.8896**	**0.8297**	0.9065	0.8481
	Elastic Transform	0.8477	0.7926	**0.8891**	**0.8273**	0.8195	0.7538	0.8857	0.8255	0.9091	0.8512
	Emboss	0.8457	0.7905	0.8880	0.8255	0.8155	0.7504	0.8889	0.8295	0.9080	0.8498
	Flip	0.8478	0.7919	0.8881	0.8255	**0.8220**	**0.7562**	0.8846	0.8240	0.9080	0.8489
	Gaussian Blur	0.8462	0.7912	0.8871	0.8246	0.8137	0.7484	0.8883	0.8280	0.9073	0.8491
	Grid Distortion	0.8484	0.7935	0.8882	0.8258	0.8205	0.7541	0.8831	0.8224	0.9083	0.8504
	Grid Dropout	0.8379	0.7819	0.8838	0.8207	0.8152	0.7491	0.8866	0.8263	0.9078	0.8491
	Image Compression	0.8437	0.7879	0.8867	0.8248	0.8148	0.7498	0.8890	0.8294	0.9065	0.8480
	Median Blur	0.8445	0.7898	0.8827	0.8196	0.8141	0.7479	0.8870	0.8273	0.9081	0.8497
	Optical Distortion	**0.8506**	**0.7944**	0.8889	**0.8273**	0.8148	0.7499	0.8863	0.8263	**0.9092**	**0.8513**
	Piecewise Affine	0.8485	0.7943	0.8869	0.8248	0.8189	0.7541	0.8861	0.8260	0.9088	0.8509
	Posterize	0.8426	0.7873	0.8862	0.8238	0.8157	0.7498	0.8880	0.8276	0.9071	0.8488
	RBC	0.8449	0.7892	0.8866	0.8240	0.8132	0.7466	0.8860	0.8256	0.9071	0.8485
	Random Crop	0.8447	0.7903	0.8877	0.8254	0.8139	0.7484	0.8872	0.8272	0.9081	0.8498
	Random Gamma	0.8455	0.7909	0.8866	0.8244	0.8134	0.7478	0.8880	0.8277	0.9081	0.8500
	Random Snow	0.8430	0.7861	0.8845	0.8217	0.8170	0.7506	0.8842	0.8240	0.9062	0.8472
	Rotate	0.8491	0.7924	0.8865	0.8239	0.8190	0.7530	0.8842	0.8233	0.9084	0.8500
	Sharpen	0.8472	0.7925	0.8877	0.8255	0.8136	0.7479	0.8865	0.8265	0.9071	0.8490
	Shift Scale Rotate	0.8461	0.7919	0.8888	0.8266	0.8198	0.7542	0.8841	0.8236	0.9083	0.8498
0.2	CLAHE	0.8421	0.7868	0.8847	0.8213	0.8141	0.7480	0.8860	0.8255	0.9072	0.8483
	Coarse Dropout	0.8448	0.7890	0.8857	0.8234	0.8114	0.7465	0.8870	0.8269	0.9081	0.8500
	Elastic Transform	0.8488	0.7932	0.8870	0.8249	0.8198	0.7546	0.8846	0.8242	**0.9100**	**0.8522**
	Emboss	0.8516	0.7957	0.8886	0.8266	0.8134	0.7473	0.8873	0.8273	0.9075	0.8491
	Flip	**0.8551**	**0.7988**	0.8870	0.8241	0.8231	0.7574	0.8803	0.8192	0.9094	0.8505
	Gaussian Blur	0.8415	0.7872	0.8859	0.8236	0.8131	0.7468	0.8877	0.8273	0.9078	0.8497
	Grid Distortion	0.8533	0.7980	0.8887	0.8268	0.8237	**0.7579**	0.8862	0.8263	0.9082	0.8499
	Grid Dropout	0.8424	0.7878	0.8849	0.8220	0.8122	0.7458	0.8855	0.8254	0.9072	0.8484
	Image Compression	0.8416	0.7851	0.8864	0.8241	0.8108	0.7462	0.8879	0.8282	0.9071	0.8489
	Median Blur	0.8452	0.7897	0.8863	0.8238	0.8145	0.7483	0.8866	0.8259	0.9067	0.8481
	Optical Distortion	0.8494	0.7937	**0.8895**	**0.8279**	0.8163	0.7509	0.8881	0.8285	0.9082	0.8502
	Piecewise Affine	0.8510	0.7959	0.8857	0.8231	0.8233	0.7568	0.8865	0.8263	0.9087	0.8503
	Posterize	0.8478	0.7921	0.8874	0.8250	0.8148	0.7494	0.8871	0.8268	0.9089	0.8508
	RBC	0.8410	0.7858	0.8868	0.8239	0.8110	0.7437	0.8867	0.8263	0.9065	0.8481
	Random Crop	0.8468	0.7909	0.8861	0.8241	0.8164	0.7509	0.8866	0.8269	0.9073	0.8491
	Random Gamma	0.8502	0.7939	0.8861	0.8242	0.8141	0.7483	**0.8892**	**0.8290**	0.9083	0.8500
	Random Snow	0.8455	0.7896	0.8814	0.8175	0.8119	0.7456	0.8856	0.8251	0.9056	0.8467
	Rotate	0.8498	0.7941	0.8885	0.8261	**0.8238**	0.7577	0.8816	0.8203	0.9073	0.8483
	Sharpen	0.8457	0.7900	0.8886	0.8261	0.8131	0.7478	0.8874	0.8273	0.9078	0.8499
	Shift Scale Rotate	0.8512	0.7955	0.8890	0.8271	0.8233	0.7571	0.8830	0.8224	0.9090	0.8507

the augmentations that achieved the highest scores, the probabilities of 0.35 and 0.45 achieved the most promising results.

In CC-CCII, MedSeg, MosMed, and Zenodo datasets, spatial transformations yielded the highest F-score, while in Ricord1a, a color operation achieved the highest F-score. Spatial-based techniques yielded better results in these datasets as they encourage variations in the shape of the lesion regions. However, Ricord1a comprises very similar images with minimal lung position and shape changes, making spatial operations counterproductive. Therefore, a color operation

Table 6.8: Results of the data augmentation evaluation (Continuation of Tables 6.6 and 6.7). p stands for probability, the blue-colored values indicate the best F-score values, while the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). Source: (Krinski et al., 2023)

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8450	0.7896	0.8885	0.8261	0.8180	0.7528	0.8875	0.8275	0.9072	0.8490
0.25	CLAHE	0.8396	0.7842	0.8842	0.8211	0.8108	0.7450	0.8869	0.8263	0.9058	0.8469
	Coarse Dropout	0.8406	0.7852	0.8847	0.8219	0.8152	0.7501	0.8858	0.8257	0.9076	0.8494
	Elastic Transform	**0.8537**	**0.7980**	0.8887	0.8269	0.8195	0.7534	0.8844	0.8239	**0.9096**	**0.8519**
	Emboss	0.8448	0.7892	0.8883	0.8260	0.8169	0.7511	0.8881	0.8279	0.9072	0.8491
	Flip	0.8526	0.7963	0.8851	0.8226	**0.8284**	**0.7631**	0.8827	0.8221	0.9082	0.8494
	Gaussian Blur	0.8483	0.7932	0.8852	0.8225	0.8125	0.7465	0.8887	0.8287	0.9073	0.8489
	Grid Distortion	0.8525	0.7971	0.8870	0.8248	0.8227	0.7568	0.8871	0.8269	0.9085	0.8503
	Grid Dropout	0.8398	0.7849	0.8820	0.8189	0.8136	0.7470	0.8837	0.8227	0.9052	0.8456
	Image Compression	0.8396	0.7845	0.8872	0.8253	0.8124	0.7467	0.8857	0.8259	0.9075	0.8494
	Median Blur	0.8439	0.7892	0.8846	0.8218	0.8177	0.7514	0.8834	0.8226	0.9069	0.8490
	Optical Distortion	0.8481	0.7928	0.8883	0.8263	0.8208	0.7547	0.8864	0.8263	0.9089	0.8509
	Piecewise Affine	0.8478	0.7921	0.8868	0.8248	0.8240	0.7581	0.8872	0.8268	0.9087	0.8503
	Posterize	0.8447	0.7891	0.8875	0.8255	0.8183	0.7533	0.8851	0.8245	0.9080	0.8497
	RBC	0.8405	0.7847	0.8863	0.8234	0.8122	0.7448	0.8843	0.8231	0.9064	0.8476
	Random Crop	0.8467	0.7910	0.8870	0.8249	0.8109	0.7454	0.8880	0.8287	0.9070	0.8486
	Random Gamma	0.8392	0.7849	0.8859	0.8232	0.8142	0.7481	0.8884	0.8283	0.9065	0.8483
	Random Snow	0.8439	0.7867	0.8802	0.8160	0.8157	0.7488	0.8860	0.8252	0.9037	0.8443
	Rotate	0.8495	0.7932	0.8879	0.8256	0.8222	0.7561	0.8787	0.8175	0.9071	0.8482
	Sharpen	0.8418	0.7870	**0.8891**	0.8268	0.8109	0.7454	**0.8895**	**0.8297**	0.9080	0.8496
	Shift Scale Rotate	0.8493	0.7951	**0.8891**	**0.8274**	0.8188	0.7527	0.8823	0.8214	0.9083	0.8497
0.3	CLAHE	0.8400	0.7846	0.8846	0.8215	0.8085	0.7428	0.8824	0.8215	0.9062	0.8474
	Coarse Dropout	0.8413	0.7860	0.8856	0.8235	0.8114	0.7457	0.8883	0.8283	0.9081	0.8497
	Elastic Transform	0.8546	0.7996	0.8893	0.8273	0.8250	0.7590	0.8847	0.8241	**0.9097**	**0.8520**
	Emboss	0.8458	0.7904	0.8881	0.8259	0.8138	0.7480	0.8877	0.8276	0.9078	0.8498
	Flip	0.8490	0.7918	0.8846	0.8213	0.8257	**0.7605**	0.8792	0.8183	0.9087	0.8496
	Gaussian Blur	0.8442	0.7892	0.8865	0.8243	0.8097	0.7440	**0.8903**	**0.8303**	0.9058	0.8471
	Grid Distortion	0.8556	0.7993	0.8894	0.8274	**0.8258**	0.7593	0.8867	0.8267	0.9089	0.8507
	Grid Dropout	0.8408	0.7854	0.8838	0.8210	0.8131	0.7472	0.8840	0.8231	0.9061	0.8469
	Image Compression	0.8508	0.7945	0.8871	0.8252	0.8138	0.7483	0.8867	0.8269	0.9080	0.8500
	Median Blur	0.8452	0.7907	0.8836	0.8209	0.8143	0.7478	0.8872	0.8272	0.9086	0.8505
	Optical Distortion	0.8527	0.7966	**0.8896**	**0.8277**	0.8184	0.7532	0.8890	0.8293	0.9091	0.8513
	Piecewise Affine	0.8561	0.8006	0.8884	0.8266	0.8224	0.7562	0.8846	0.8243	0.9087	0.8505
	Posterize	0.8440	0.7877	0.8865	0.8241	0.8127	0.7472	0.8858	0.8257	0.9065	0.8477
	RBC	0.8421	0.7862	0.8865	0.8239	0.8097	0.7432	0.8866	0.8260	0.9066	0.8478
	Random Crop	0.8476	0.7920	0.8869	0.8248	0.8144	0.7487	0.8867	0.8267	0.9073	0.8490
	Random Gamma	0.8513	0.7949	0.8865	0.8243	0.8128	0.7465	0.8867	0.8267	0.9080	0.8498
	Random Snow	0.8468	0.7901	0.8815	0.8169	0.8148	0.7484	0.8848	0.8237	0.9043	0.8450
	Rotate	0.8494	0.7933	0.8876	0.8248	0.8253	0.7592	0.8789	0.8174	0.9066	0.8472
	Sharpen	0.8423	0.7891	0.8862	0.8231	0.8093	0.7440	0.8866	0.8260	0.9089	0.8512
	Shift Scale Rotate	**0.8567**	**0.8013**	0.8887	0.8267	0.8237	0.7571	0.8808	0.8199	0.9092	0.8506

(Median Blur) achieved the highest F-score, indicating that this dataset is more sensitive to color operations. However, the results achieved in this experiment, did not yielded interesting results. When compared with the baseline, where the network was trained without data augmentation, the augmentation techniques produced small gains. Furthermore, the data augmentation process resulted in a small difference between the probabilities applied, making it unclear which is the optimal probability in this case.

Table 6.9: Results of the data augmentation evaluation (Continuation of Tables 6.6, 6.7, and 6.8). *p* stands for probability, the blue-colored values indicate the best F-score values, while the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation).

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8450	0.7896	0.8885	0.8261	0.8180	0.7528	0.8875	0.8275	0.9072	0.8490
0.35	CLAHE	0.8382	0.7820	0.8846	0.8213	0.8085	0.7441	0.8849	0.8240	0.9057	0.8470
	Coarse Dropout	0.8403	0.7850	0.8848	0.8222	0.8135	0.7484	0.8871	0.8267	0.9063	0.8477
	Elastic Transform	**0.8551**	**0.7993**	**0.8906**	**0.8286**	0.8214	0.7561	0.8849	0.8243	**0.9090**	**0.8510**
	Emboss	0.8483	0.7926	0.8870	0.8243	0.8175	0.7509	0.8890	0.8289	0.9074	0.8488
	Flip	0.8476	0.7925	0.8861	0.8228	0.8258	0.7596	0.8773	0.8157	0.9083	0.8491
	Gaussian Blur	0.8490	0.7932	0.8853	0.8225	0.8148	0.7493	0.8869	0.8262	0.9082	0.8499
	Grid Distortion	0.8528	0.7975	0.8884	0.8263	**0.8307**	**0.7653**	0.8868	0.8264	0.9089	0.8506
	Grid Dropout	0.8372	0.7813	0.8811	0.8174	0.8140	0.7479	0.8839	0.8231	0.9050	0.8459
	Image Compression	0.8446	0.7887	0.8861	0.8238	0.8140	0.7486	0.8887	0.8289	0.9082	0.8501
	Median Blur	0.8489	0.7928	0.8877	0.8255	0.8174	0.7509	0.8871	0.8268	0.9067	0.8481
	Optical Distortion	0.8477	0.7923	0.8869	0.8243	0.8212	0.7553	0.8837	0.8234	0.9088	0.8508
	Piecewise Affine	0.8537	0.7981	0.8884	0.8264	0.8262	0.7595	0.8849	0.8246	0.9082	0.8496
	Posterize	0.8396	0.7841	0.8883	0.8264	0.8172	0.7513	**0.8895**	0.8293	0.9077	0.8495
	RBC	0.8442	0.7879	0.8854	0.8223	0.8127	0.7468	0.8846	0.8239	0.9063	0.8476
	Random Crop	0.8366	0.7811	0.8872	0.8251	0.8138	0.7480	0.8873	0.8275	0.9067	0.8483
	Random Gamma	0.8439	0.7888	0.8857	0.8230	0.8134	0.7474	0.8879	0.8282	0.9074	0.8491
	Random Snow	0.8389	0.7836	0.8773	0.8125	0.8099	0.7429	0.8854	0.8245	0.9045	0.8448
	Rotate	0.8528	0.7971	0.8870	0.8246	0.8258	0.7603	0.8776	0.8160	0.9056	0.8462
	Sharpen	0.8431	0.7889	0.8859	0.8234	0.8120	0.7466	0.8894	**0.8296**	0.9089	0.8508
	Shift Scale Rotate	0.8538	0.7989	0.8868	0.8247	0.8238	0.7578	0.8820	0.8208	0.9081	0.8495
0.4	CLAHE	0.8417	0.7866	0.8810	0.8170	0.8066	0.7413	0.8877	0.8272	0.9047	0.8454
	Coarse Dropout	0.8429	0.7873	0.8852	0.8228	0.8134	0.7476	0.8878	0.8278	0.9077	0.8494
	Elastic Transform	0.8544	0.7983	0.8888	0.8268	0.8266	0.7599	0.8848	0.8246	**0.9094**	**0.8513**
	Emboss	0.8372	0.7825	0.8886	0.8259	0.8117	0.7454	0.8873	0.8274	0.9070	0.8485
	Flip	0.8457	0.7898	0.8864	0.8236	0.8250	0.7583	0.8772	0.8156	0.9061	0.8465
	Gaussian Blur	0.8480	0.7918	0.8864	0.8237	0.8134	0.7477	0.8857	0.8255	0.9076	0.8494
	Grid Distortion	0.8540	0.7993	**0.8905**	**0.8287**	0.8287	**0.7624**	0.8853	0.8250	0.9092	0.8509
	Grid Dropout	0.8411	0.7852	0.8837	0.8204	0.8115	0.7457	0.8834	0.8225	0.9048	0.8453
	Image Compression	0.8468	0.7912	0.8860	0.8239	0.8168	0.7518	0.8874	0.8277	0.9075	0.8490
	Median Blur	0.8441	0.7899	0.8860	0.8235	0.8187	0.7531	0.8883	0.8284	0.9077	0.8493
	Optical Distortion	0.8492	0.7925	0.8892	0.8274	0.7556	0.7032	**0.8896**	**0.8303**	**0.9094**	0.8512
	Piecewise Affine	0.8508	0.7956	0.8859	0.8234	0.8261	0.7601	0.8844	0.8239	0.9080	0.8494
	Posterize	0.8429	0.7882	0.8853	0.8224	0.8196	0.7527	0.8859	0.8250	0.9076	0.8493
	RBC	0.8417	0.7861	0.8865	0.8239	0.8122	0.7449	0.8853	0.8244	0.9064	0.8474
	Random Crop	0.8459	0.7910	0.8877	0.8254	0.8107	0.7453	0.8866	0.8265	0.9081	0.8498
	Random Gamma	0.8412	0.7864	0.8865	0.8241	0.8130	0.7469	0.8867	0.8264	0.9074	0.8490
	Random Snow	0.8420	0.7859	0.8782	0.8135	0.8113	0.7444	0.8824	0.8211	0.9041	0.8444
	Rotate	0.8565	0.7996	0.8864	0.8238	0.8258	0.7597	0.8758	0.8140	0.9058	0.8460
	Sharpen	0.8423	0.7874	0.8894	0.8271	0.8122	0.7459	0.8877	0.8277	0.9074	0.8493
	Shift Scale Rotate	**0.8593**	**0.8038**	0.8879	0.8255	0.8279	0.7622	0.8812	0.8198	0.9074	0.8486

6.2.2 Experiment II: Training Sets Unified

Inspired by promising outcomes observed in other research areas (Laroca et al., 2021, 2023), an alternative methodology that integrates training subsets from multiple datasets into a single, larger set, was evaluated alongside the traditional approach of using separate subsets from a single dataset for training and testing. Table 6.11 presents the evaluation results for probabilities 0.05 and 0.1, Table 6.12 presents the results for probabilities 0.15 and 0.2, Table 6.13 presents the results for probabilities 0.25 and 0.3, Table 6.14 presents the results for probabilities 0.35 and 0.4,

Table 6.10: Results of the data augmentation evaluation (Continuation of Tables 6.6, 6.7, 6.8, and 6.9). *p* stands for probability, the blue-colored values indicate the best F-score values, while the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation).

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8450	0.7896	0.8885	0.8261	0.8180	0.7528	0.8875	0.8275	0.9072	0.8490
0.45	CLAHE	0.8394	0.7831	0.8809	0.8171	0.8041	0.7386	0.8854	0.8249	0.9053	0.8465
	Coarse Dropout	0.8458	0.7907	0.8833	0.8206	0.8156	0.7502	0.8870	0.8267	0.9069	0.8484
	Elastic Transform	0.8522	0.7964	**0.8895**	**0.8277**	0.8231	0.7572	0.8833	0.8226	0.9087	0.8503
	Emboss	0.8440	0.7889	0.8870	0.8242	0.8136	0.7465	0.8881	0.8282	0.9065	0.8479
	Flip	0.8531	0.7961	0.8865	0.8234	0.8262	0.7599	0.8760	0.8142	0.9050	0.8450
	Gaussian Blur	0.8468	0.7915	0.8857	0.8231	0.8159	0.7495	**0.8902**	0.8302	0.9061	0.8479
	Grid Distortion	0.8532	0.7987	0.8889	0.8269	0.8259	0.7603	0.8839	0.8233	**0.9104**	**0.8524**
	Grid Dropout	0.8364	0.7803	0.8821	0.8187	0.8112	0.7449	0.8812	0.8201	0.9042	0.8445
	Image Compression	0.8433	0.7871	0.8887	0.8266	0.8146	0.7487	0.8877	0.8280	0.9067	0.8484
	Median Blur	0.8480	0.7927	0.8839	0.8215	0.8164	0.7506	0.8850	0.8243	0.9076	0.8492
	Optical Distortion	0.8487	0.7938	0.8889	0.8270	0.8189	0.7534	**0.8902**	**0.8306**	0.9095	0.8515
	Piecewise Affine	0.8586	0.8032	0.8891	0.8273	0.8262	0.7601	0.8851	0.8246	0.9091	0.8512
	Posterize	0.8413	0.7868	0.8888	0.8269	0.8207	0.7550	0.8851	0.8244	0.9068	0.8485
	RBC	0.8438	0.7876	0.8880	0.8255	0.8085	0.7417	0.8833	0.8217	0.9064	0.8478
	Random Crop	0.8415	0.7859	0.8871	0.8249	0.8138	0.7483	0.8881	0.8282	0.9073	0.8487
	Random Gamma	0.8472	0.7914	0.8860	0.8234	0.8121	0.7462	0.8880	0.8279	0.9078	0.8498
	Random Snow	0.8387	0.7825	0.8786	0.8137	0.8100	0.7434	0.8842	0.8229	0.9045	0.8446
	Rotate	0.8562	0.7987	0.8868	0.8239	0.8264	0.7600	0.8741	0.8119	0.9064	0.8465
	Sharpen	0.8419	0.7871	0.8878	0.8256	0.8124	0.7462	0.8881	0.8278	0.9087	0.8508
	Shift Scale Rotate	**0.8598**	**0.8039**	0.8890	0.8270	**0.8272**	**0.7612**	0.8773	0.8155	0.9074	0.8486
0.5	CLAHE	0.8369	0.7820	0.8815	0.8183	0.8073	0.7426	0.8864	0.8260	0.9064	0.8473
	Coarse Dropout	0.8406	0.7857	0.8835	0.8205	0.8159	0.7501	0.8856	0.8254	0.9066	0.8483
	Elastic Transform	0.8519	0.7965	0.8895	0.8276	0.8286	0.7624	0.8845	0.8240	0.9090	0.8504
	Emboss	0.8398	0.7849	0.8871	0.8245	0.8134	0.7466	0.8886	0.8284	0.9070	0.8481
	Flip	0.8529	0.7955	0.8850	0.8219	0.8264	0.7600	0.8759	0.8138	0.9070	0.8476
	Gaussian Blur	0.8438	0.7883	0.8822	0.8192	0.8123	0.7475	0.8886	0.8287	0.9082	0.8502
	Grid Distortion	0.8535	0.7981	**0.8902**	0.8283	**0.8328**	**0.7653**	0.8833	0.8227	0.9088	0.8506
	Grid Dropout	0.8388	0.7832	0.8808	0.8166	0.8117	0.7457	0.8820	0.8207	0.9043	0.8448
	Image Compression	0.8424	0.7870	0.8879	0.8263	0.8127	0.7472	0.8891	0.8290	0.9092	0.8511
	Median Blur	0.8440	0.7896	0.8833	0.8209	0.8189	0.7529	**0.8904**	**0.8303**	0.9078	0.8496
	Optical Distortion	0.8518	0.7966	0.8900	**0.8284**	0.8192	0.7534	0.8886	0.8291	0.9091	**0.8513**
	Piecewise Affine	**0.8571**	**0.8016**	0.8882	0.8260	0.8255	0.7590	0.8850	0.8244	**0.9096**	0.8512
	Posterize	0.8439	0.7887	0.8871	0.8251	0.8175	0.7514	0.8860	0.8253	0.9070	0.8486
	RBC	0.8422	0.7866	0.8859	0.8230	0.8104	0.7429	0.8859	0.8250	0.9062	0.8476
	Random Crop	0.8402	0.7844	0.8890	0.8271	0.8161	0.7503	0.8872	0.8269	0.9077	0.8495
	Random Gamma	0.8442	0.7883	0.8866	0.8243	0.8110	0.7438	0.8861	0.8261	0.9067	0.8482
	Random Snow	0.8413	0.7842	0.8771	0.8119	0.8114	0.7447	0.8820	0.8208	0.9043	0.8444
	Rotate	0.8525	0.7961	0.8856	0.8230	0.8243	0.7579	0.8735	0.8110	0.9061	0.8464
	Sharpen	0.8442	0.7893	0.8869	0.8242	0.8098	0.7441	0.8879	0.8278	0.9064	0.8479
	Shift Scale Rotate	0.8520	0.7964	0.8880	0.8258	0.8273	0.7609	0.8786	0.8171	0.9084	0.8496

and Table 6.15 presents the results for probabilities 0.45 and 0.5. The Wilcoxon Signed-Rank Test was utilized to perform statistical analysis between the segmentation model trained applying data augmentation technique and training the segmentation model without applying data augmentation. The highlighted values in green demonstrate the data augmentation techniques that yielded P-values less than 0.05, thus rejecting the null hypothesis, indicating a statistical difference and the achieved results are superior to those without data augmentation. A comparison between training with a unified training set and training with individual training sets was also conducted.

Table 6.11: Results of the data augmentation evaluation when unifying the training sets. p stands for probability, the blue-colored values indicate the best F-score value, and the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show the techniques where training with the unified set achieved a P-value lower than 0.05 when compared with training with a single training set, and the null hypothesis was rejected. Source: (Krinski et al., 2023)

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	CLAHE	0.8580	0.8025	0.8873	0.8247	0.8198	0.7554	0.8585	0.7932	0.9098	0.8521
	Coarse Dropout	0.8624	0.8071	0.8882	0.8260	0.8266	0.7631	0.8585	0.7930	0.9097	0.8517
	Elastic Transform	**0.8722**	**0.8168**	0.8913	0.8294	0.8281	0.7634	0.8553	0.7887	0.9113	0.8536
	Emboss	0.8659	0.8103	0.8876	0.8250	0.8237	0.7607	0.8552	0.7889	0.9113	0.8536
	Flip	0.8659	0.8103	0.8911	0.8290	0.8265	0.7628	0.8610	0.7958	0.9111	0.8535
	Gaussian Blur	0.8641	0.8089	0.8868	0.8242	0.8189	0.7555	0.8579	0.7925	0.9106	0.8527
	Grid Distortion	0.8687	0.8133	**0.8922**	**0.8303**	0.8272	0.7624	0.8605	0.7951	0.9115	0.8539
	Grid Dropout	0.8631	0.8069	0.8867	0.8243	0.8218	0.7576	0.8578	0.7917	0.9094	0.8511
	Image Compression	0.8620	0.8060	0.8870	0.8242	0.8203	0.7564	0.8589	0.7932	0.9101	0.8519
	Median Blur	0.8608	0.8055	0.8884	0.8259	0.8227	0.7584	0.8592	0.7933	0.9103	0.8526
	Optical Distortion	0.8685	0.8126	0.8898	0.8279	0.8236	0.7609	0.8608	0.7958	0.9110	0.8535
	Piecewise Affine	0.8708	0.8153	0.8915	0.8296	**0.8318**	**0.7682**	**0.8709**	**0.8073**	**0.9119**	**0.8544**
	Posterize	0.8643	0.8092	0.8897	0.8274	0.8261	0.7627	0.8641	0.7996	0.9110	0.8535
	RBC	0.8614	0.8054	0.8856	0.8226	0.8207	0.7569	0.8565	0.7905	0.9103	0.8522
	Random Crop	0.8607	0.8056	0.8870	0.8239	0.8195	0.7554	0.8524	0.7856	0.9088	0.8505
	Random Gamma	0.8675	0.8105	0.8854	0.8225	0.8192	0.7545	0.8526	0.7860	0.9087	0.8502
	Random Snow	0.8647	0.8096	0.8873	0.8244	0.8213	0.7580	0.8586	0.7931	0.9103	0.8527
	Rotate	0.8695	0.8144	0.8902	0.8279	0.8277	0.7635	0.8609	0.7956	0.9118	0.8543
	Sharpen	0.8661	0.8098	0.8901	0.8277	0.8167	0.7532	0.8545	0.7884	0.9108	0.8526
	Shift Scale Rotate	0.8680	0.8129	0.8910	0.8293	0.8289	0.7657	0.8647	0.8001	0.9115	0.8541
0.1	CLAHE	0.8561	0.8004	0.8866	0.8237	0.8172	0.7548	0.8606	0.7953	0.9084	0.8502
	Coarse Dropout	0.8609	0.8059	0.8883	0.8264	0.8239	0.7612	0.8648	0.8004	0.9103	0.8528
	Elastic Transform	0.8720	0.8176	0.8927	0.8313	0.8315	0.7676	0.8567	0.7904	**0.9122**	**0.8552**
	Emboss	0.8633	0.8083	0.8885	0.8263	0.8189	0.7564	0.8609	0.7958	0.9114	0.8539
	Flip	0.8691	0.8139	0.8907	0.8289	0.8267	0.7625	0.8542	0.7880	0.9104	0.8527
	Gaussian Blur	0.8657	0.8108	0.8880	0.8259	0.8134	0.7508	0.8576	0.7916	0.9098	0.8516
	Grid Distortion	0.8713	0.8173	0.8935	0.8317	0.8319	0.7685	0.8536	0.7870	0.9119	0.8545
	Grid Dropout	0.8610	0.8048	0.8873	0.8254	0.8297	0.7670	**0.8668**	**0.8023**	0.9108	0.8530
	Image Compression	0.8634	0.8081	0.8875	0.8250	0.8205	0.7570	0.8605	0.7950	0.9091	0.8511
	Median Blur	0.8627	0.8076	0.8890	0.8261	0.8155	0.7527	0.8561	0.7899	0.9100	0.8523
	Optical Distortion	0.8629	0.8085	0.8911	0.8291	0.8255	0.7615	0.8581	0.7925	0.9115	0.8539
	Piecewise Affine	0.8729	0.8180	**0.8937**	**0.8320**	0.8306	0.7666	0.8572	0.7910	0.9121	0.8547
	Posterize	0.8621	0.8074	0.8890	0.8265	0.8212	0.7576	0.8610	0.7958	0.9106	0.8526
	RBC	0.8617	0.8049	0.8885	0.8254	0.8241	0.7603	0.8608	0.7958	0.9098	0.8519
	Random Crop	0.8606	0.8050	0.8870	0.8239	0.8166	0.7536	0.8559	0.7899	0.9095	0.8513
	Random Gamma	0.8645	0.8091	0.8882	0.8253	0.8197	0.7550	0.8566	0.7906	0.9091	0.8510
	Random Snow	0.8616	0.8065	0.8896	0.8270	0.8227	0.7595	0.8576	0.7921	0.9106	0.8530
	Rotate	0.8724	0.8172	0.8924	0.8312	**0.8353**	**0.7710**	0.8568	0.7908	0.9117	0.8541
	Sharpen	0.8633	0.8080	0.8899	0.8278	0.8236	0.7603	0.8633	0.7985	0.9116	0.8545
	Shift Scale Rotate	**0.8743**	**0.8204**	0.8926	0.8317	0.8297	0.7653	0.8553	0.7890	0.9118	0.8544

The Wilcoxon Signed-Rank Test was utilized to perform statistical analysis between the segmentation model trained with the unified set and the segmentation model trained with the individual training sets. The underscored values presented in Tables 6.11, 6.12, 6.10, 6.14, and 6.15 are the techniques where training with the unified set resulted in a P-value lower than 0.05, thereby rejecting the null hypothesis, suggesting that there is a statistical difference and the achieved results with the unified training set are superior to those with the individual training sets.

Table 6.12: Results of the data augmentation evaluation when unifying the training sets (Continuation of Table 6.11). p stands for probability, the blue-colored values indicate the best F-score values, and the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show the techniques where training with the combined training sets achieved a P-value lower than 0.05 when compared with training with a single training set, and the null hypothesis was rejected. Source: (Krinski et al., 2023)

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.15	CLAHE	0.8548	0.7992	0.8883	0.8252	0.8179	0.7552	0.8602	0.7947	0.9088	0.8506
	Coarse Dropout	0.8629	0.8071	0.8895	0.8276	0.8283	0.7642	0.8610	0.7959	0.9100	0.8522
	Elastic Transform	0.8734	0.8203	0.8920	0.8310	0.8345	0.7704	0.8594	0.7938	0.9111	0.8537
	Emboss	0.8643	0.8097	0.8908	0.8291	0.8229	0.7591	0.8577	0.7918	0.9115	0.8541
	Flip	0.8704	0.8164	0.8922	0.8308	0.8358	0.7717	0.8587	0.7933	0.9123	0.8547
	Gaussian Blur	0.8641	0.8086	0.8912	0.8295	0.8207	0.7575	**0.8729**	**0.8100**	0.9127	0.8557
	Grid Distortion	0.8736	0.8193	0.8939	0.8326	0.8353	0.7720	0.8612	0.7961	0.9136	0.8568
	Grid Dropout	0.8633	0.8080	0.8873	0.8250	0.8246	0.7610	0.8635	0.7987	0.9102	0.8521
	Image Compression	0.8654	0.8103	0.8863	0.8235	0.8199	0.7561	0.8574	0.7915	0.9095	0.8511
	Median Blur	0.8649	0.8093	0.8901	0.8286	0.8258	0.7615	0.8677	0.8036	0.9116	0.8545
	Optical Distortion	0.8676	0.8130	0.8914	0.8297	0.8245	0.7610	0.8538	0.7874	0.9114	0.8541
	Piecewise Affine	0.8758	0.8225	**0.8944**	0.8335	0.8353	0.7713	0.8610	0.7955	0.9135	0.8565
	Posterize	0.8623	0.8070	0.8905	0.8280	0.8242	0.7604	0.8617	0.7967	0.9097	0.8520
	RBC	0.8581	0.8019	0.8884	0.8255	0.8222	0.7577	0.8570	0.7910	0.9086	0.8505
	Random Crop	0.8648	0.8095	0.8890	0.8267	0.8238	0.7602	0.8668	0.8026	0.9109	0.8533
	Random Gamma	0.8609	0.8045	0.8887	0.8258	0.8176	0.7536	0.8596	0.7938	0.9090	0.8508
	Random Snow	0.8609	0.8063	0.8907	0.8285	0.8281	0.7651	0.8643	0.7997	0.9104	0.8531
	Rotate	0.8762	0.8219	**0.8944**	0.8333	0.8350	0.7696	0.8517	0.7847	0.9115	0.8539
	Sharpen	0.8641	0.8087	0.8893	0.8273	0.8220	0.7580	0.8663	0.8022	0.9128	0.8558
	Shift Scale Rotate	**0.8766**	**0.8234**	0.8941	**0.8336**	**0.8374**	**0.7744**	0.8652	0.8007	**0.9144**	**0.8576**
0.2	CLAHE	0.8569	0.8020	0.8872	0.8240	0.8183	0.7546	0.8602	0.7946	0.9093	0.8509
	Coarse Dropout	0.8597	0.8052	0.8873	0.8255	0.8244	0.7605	0.8575	0.7915	0.9106	0.8528
	Elastic Transform	0.8746	0.8215	0.8959	0.8352	0.8356	0.7728	0.8572	0.7912	0.9133	0.8564
	Emboss	0.8632	0.8088	0.8907	0.8288	0.8285	0.7641	**0.8686**	**0.8046**	0.9131	0.8557
	Flip	0.8722	0.8177	0.8926	0.8310	0.8316	0.7672	0.8561	0.7900	0.9119	0.8542
	Gaussian Blur	0.8615	0.8067	0.8888	0.8264	0.8170	0.7527	0.8616	0.7965	0.9116	0.8541
	Grid Distortion	0.8750	0.8219	0.8961	0.8356	0.8391	0.7761	0.8597	0.7941	0.9133	0.8561
	Grid Dropout	0.8613	0.8058	0.8879	0.8259	0.8240	0.7611	0.8591	0.7934	0.9090	0.8506
	Image Compression	0.8625	0.8068	0.8876	0.8247	0.8166	0.7537	0.8569	0.7910	0.9091	0.8510
	Median Blur	0.8656	0.8104	0.8901	0.8282	0.8173	0.7545	0.8630	0.7979	0.9108	0.8531
	Optical Distortion	0.8678	0.8133	0.8921	0.8301	0.8264	0.7633	0.8632	0.7984	0.9121	0.8550
	Piecewise Affine	**0.8761**	**0.8231**	0.8957	0.8351	0.8365	0.7732	0.8603	0.7948	**0.9140**	**0.8569**
	Posterize	0.8609	0.8058	0.8889	0.8264	0.8201	0.7559	0.8530	0.7860	0.9104	0.8523
	RBC	0.8599	0.8045	0.8873	0.8240	0.8209	0.7565	0.8486	0.7812	0.9086	0.8502
	Random Crop	0.8632	0.8070	0.8871	0.8243	0.8209	0.7576	0.8632	0.7986	0.9103	0.8523
	Random Gamma	0.8631	0.8074	0.8860	0.8233	0.8206	0.7572	0.8587	0.7929	0.9096	0.8516
	Random Snow	0.8632	0.8089	0.8909	0.8286	0.8278	0.7650	0.8622	0.7971	0.9116	0.8540
	Rotate	0.8753	0.8216	0.8949	0.8343	0.8391	0.7741	0.8542	0.7878	0.9121	0.8544
	Sharpen	0.8631	0.8077	0.8895	0.8277	0.8202	0.7566	0.8633	0.7985	0.9115	0.8541
	Shift Scale Rotate	**0.8761**	0.8230	**0.8992**	**0.8392**	**0.8431**	**0.7808**	0.8666	0.8021	0.9135	0.8568

The results displayed in Tables 6.11, 6.12, 6.13, 6.14, and 6.15 demonstrate that employing the unified training set produced favorable outcomes compared to utilizing individual training sets. Specifically, for a probability of 0.05, the F-score was higher when training with the unified training set than with the baseline in all data augmentation techniques used in the CC-CCII dataset. This pattern was also observed in 11 data augmentation techniques in MosMed and 10 in Zenodo. At a probability of 0.1, the unified training set yielded superior F-score

Table 6.13: Results of the data augmentation evaluation when unifying the training sets (Continuation of Tables 6.11 and 6.12). p stands for probability, the blue-colored values indicate the best F-score values, and the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show the techniques in which training with the combined training sets achieved a P-value lower than 0.05 when compared with training with a single training set, and the null hypothesis was rejected. Source: (Krinski et al., 2023)

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.25	CLAHE	0.8565	0.8006	0.8856	0.8227	0.8212	0.7579	0.8654	0.8010	0.9098	0.8516
	Coarse Dropout	0.8635	0.8084	0.8899	0.8279	0.8292	0.7653	0.8637	0.7988	0.9113	0.8538
	Elastic Transform	0.8758	0.8235	**0.8981**	**0.8376**	0.8378	0.7739	0.8594	0.7937	0.9135	0.8568
	Emboss	0.8641	0.8105	0.8925	0.8306	0.8233	0.7600	**0.8707**	**0.8074**	0.9124	0.8555
	Flip	0.8717	0.8178	0.8926	0.8311	0.8338	0.7697	0.8546	0.7884	0.9113	0.8532
	Gaussian Blur	0.8604	0.8060	0.8898	0.8279	0.8218	0.7582	0.8682	0.8040	0.9115	0.8542
	Grid Distortion	0.8761	0.8236	0.8960	0.8349	0.8398	0.7759	0.8547	0.7882	0.9137	0.8568
	Grid Dropout	0.8628	0.8072	0.8890	0.8270	0.8270	0.7634	0.8607	0.7952	0.9079	0.8499
	Image Compression	0.8629	0.8078	0.8892	0.8269	0.8195	0.7569	0.8669	0.8029	0.9112	0.8535
	Median Blur	0.8622	0.8081	0.8881	0.8261	0.8219	0.7585	0.8632	0.7981	0.9123	0.8550
	Optical Distortion	0.8699	0.8161	0.8927	0.8317	0.8307	0.7681	0.8636	0.7987	0.9126	0.8560
	Piecewise Affine	**0.8778**	**0.8250**	0.8959	0.8357	0.8425	0.7791	0.8633	0.7982	**0.9144**	**0.8577**
	Posterize	0.8627	0.8068	0.8882	0.8259	0.8160	0.7521	0.8522	0.7857	0.9089	0.8506
	RBC	0.8602	0.8035	0.8882	0.8255	0.8228	0.7586	0.8602	0.7948	0.9098	0.8517
	Random Crop	0.8632	0.8072	0.8883	0.8257	0.8220	0.7587	0.8634	0.7990	0.9101	0.8526
	Random Gamma	0.8640	0.8087	0.8893	0.8272	0.8229	0.7600	0.8611	0.7957	0.9096	0.8518
	Random Snow	0.8634	0.8093	0.8913	0.8291	0.8288	0.7665	0.8665	0.8021	0.9130	0.8556
	Rotate	0.8762	0.8221	0.8952	0.8343	0.8409	0.7758	0.8514	0.7848	0.9118	0.8539
	Sharpen	0.8625	0.8074	0.8906	0.8282	0.8183	0.7549	0.8576	0.7919	0.9115	0.8540
	Shift Scale Rotate	0.8774	0.8244	0.8973	0.8372	**0.8432**	**0.7799**	0.8602	0.7945	0.9133	0.8560
0.3	CLAHE	0.8564	0.8001	0.8854	0.8229	0.8202	0.7569	0.8648	0.8000	0.9096	0.8516
	Coarse Dropout	0.8632	0.8084	0.8910	0.8290	0.8284	0.7645	0.8598	0.7945	0.9094	0.8517
	Elastic Transform	0.8747	0.8235	0.8978	0.8383	0.8414	0.7781	0.8622	0.7971	**0.9147**	**0.8583**
	Emboss	0.8657	0.8110	0.8901	0.8280	0.8219	0.7593	0.8682	0.8039	0.9127	0.8557
	Flip	0.8732	0.8193	0.8933	0.8323	0.8297	0.7658	0.8501	0.7831	0.9108	0.8523
	Gaussian Blur	0.8644	0.8085	0.8899	0.8281	0.8154	0.7530	**0.8718**	**0.8085**	0.9121	0.8548
	Grid Distortion	0.8750	0.8225	**0.8994**	**0.8394**	**0.8448**	**0.7814**	0.8625	0.7975	0.9141	0.8575
	Grid Dropout	0.8563	0.8019	0.8881	0.8262	0.8277	0.7646	0.8627	0.7973	0.9094	0.8511
	Image Compression	0.8629	0.8078	0.8882	0.8259	0.8243	0.7607	0.8639	0.7993	0.9115	0.8538
	Median Blur	0.8647	0.8093	0.8900	0.8274	0.8190	0.7550	0.8608	0.7954	0.9124	0.8552
	Optical Distortion	0.8700	0.8157	0.8928	0.8316	0.8238	0.7613	0.8637	0.7989	0.9141	0.8575
	Piecewise Affine	0.8764	0.8229	0.8952	0.8349	0.8384	0.7750	0.8598	0.7942	0.9137	0.8567
	Posterize	0.8625	0.8067	0.8902	0.8281	0.8243	0.7597	0.8593	0.7940	0.9111	0.8536
	RBC	0.8591	0.8028	0.8904	0.8281	0.8299	0.7653	0.8632	0.7982	0.9094	0.8514
	Random Crop	0.8631	0.8073	0.8859	0.8228	0.8193	0.7558	0.8659	0.8017	0.9107	0.8527
	Random Gamma	0.8635	0.8077	0.8883	0.8254	0.8199	0.7562	0.8574	0.7912	0.9092	0.8511
	Random Snow	0.8659	0.8112	0.8901	0.8284	0.8333	0.7703	0.8671	0.8029	0.9125	0.8557
	Rotate	**0.8774**	0.8237	0.8964	0.8356	0.8401	0.7757	0.8494	0.7823	0.9112	0.8533
	Sharpen	0.8635	0.8085	0.8890	0.8270	0.8144	0.7512	0.8629	0.7980	0.9113	0.8539
	Shift Scale Rotate	0.8767	**0.8239**	0.8984	0.8382	0.8417	0.7786	0.8583	0.7924	0.9126	0.8554

values in 2 data augmentation techniques in the MedSeg dataset, 13 techniques in MosMed, 11 in Zenodo, and all techniques in the CC-CCII dataset, as shown in Tables 6.11, 6.12, and 6.13. At a probability of 0.15, the unified training set produced higher F-score values in 6 techniques in the MedSeg dataset, 17 in MosMed, 13 in Zenodo, and 19 in CC-CCII. At a probability of 0.2, the unified training set yielded better F-score values in 8 techniques in MedSeg, 12 in MosMed, 11 in Zenodo, and all techniques in the CC-CCII dataset.

Table 6.14: Results of the data augmentation evaluation when unifying the training sets (Continuation of Tables 6.11, 6.12 and 6.13). *p* stands for probability, the blue-colored values indicate the best F-score values, and the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show the techniques in which training with the combined training sets achieved a P-value lower than 0.05 when compared with training with a single training set, and the null hypothesis was rejected.

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.35	CLAHE	0.8542	0.7986	0.8867	0.8233	0.8205	0.7568	0.8656	0.8012	0.9086	0.8507
	Coarse Dropout	0.8647	0.8105	0.8904	0.8284	0.8343	0.7706	0.8673	0.8031	0.9118	0.8541
	Elastic Transform	0.8771	0.8250	**0.8989**	**0.8394**	**0.8459**	**0.7834**	0.8647	0.7997	**0.9147**	**0.8581**
	Emboss	0.8643	0.8100	0.8913	0.8295	0.8229	0.7592	**0.8713**	**0.8079**	0.9130	0.8561
	Flip	0.8714	0.8171	0.8951	0.8338	0.8348	0.7714	0.8519	0.7855	0.9107	0.8523
	Gaussian Blur	0.8668	0.8113	0.8909	0.8293	0.8211	0.7574	0.8691	0.8050	0.9122	0.8548
	Grid Distortion	0.8786	**0.8266**	0.8985	0.8381	0.8379	0.7744	0.8548	0.7884	0.9138	0.8568
	Grid Dropout	0.8615	0.8069	0.8883	0.8263	0.8284	0.7636	0.8570	0.7907	0.9089	0.8505
	Image Compression	0.8658	0.8097	0.8864	0.8239	0.8146	0.7515	0.8569	0.7910	0.9110	0.8531
	Median Blur	0.8628	0.8071	0.8900	0.8283	0.8187	0.7545	0.8621	0.7970	0.9123	0.8549
	Optical Distortion	0.8726	0.8185	0.8935	0.8325	0.8262	0.7634	0.8566	0.7908	0.9136	0.8567
	Piecewise Affine	0.8774	0.8251	0.8982	0.8386	0.8413	0.7788	0.8623	0.7970	0.9138	0.8570
	Posterize	0.8584	0.8033	0.8908	0.8286	0.8249	0.7607	0.8642	0.7994	0.9100	0.8520
	RBC	0.8609	0.8046	0.8883	0.8252	0.8236	0.7595	0.8570	0.7912	0.9092	0.8514
	Random Crop	0.8588	0.8038	0.8885	0.8257	0.8212	0.7573	0.8585	0.7928	0.9088	0.8509
	Random Gamma	0.8644	0.8083	0.8892	0.8263	0.8158	0.7531	0.8623	0.7975	0.9102	0.8523
	Random Snow	0.8670	0.8121	0.8910	0.8290	0.8323	0.7691	0.8657	0.8011	0.9120	0.8549
	Rotate	**0.8787**	0.8248	0.8959	0.8354	0.8405	0.7765	0.8498	0.7832	0.9111	0.8530
	Sharpen	0.8625	0.8072	0.8920	0.8303	0.8194	0.7560	0.8598	0.7946	0.9110	0.8533
	Shift Scale Rotate	0.8783	0.8257	0.8977	0.8377	0.8454	0.7826	0.8598	0.7943	0.9135	0.8565
0.4	CLAHE	0.8530	0.7973	0.8865	0.8233	0.8154	0.7524	0.8642	0.7994	0.9097	0.8517
	Coarse Dropout	0.8635	0.8076	0.8896	0.8279	0.8253	0.7619	0.8594	0.7938	0.9105	0.8527
	Elastic Transform	0.8771	0.8252	0.8975	0.8378	0.8403	0.7772	0.8582	0.7923	**0.9144**	**0.8575**
	Emboss	0.8652	0.8104	0.8902	0.8285	0.8246	0.7610	0.8681	0.8042	0.9119	0.8549
	Flip	0.8722	0.8173	0.8935	0.8321	0.8360	0.7713	0.8484	0.7813	0.9096	0.8505
	Gaussian Blur	0.8637	0.8093	0.8909	0.8293	0.8247	0.7601	**0.8714**	**0.8080**	0.9122	0.8551
	Grid Distortion	**0.8784**	0.8260	0.8989	0.8390	0.8402	0.7767	0.8558	0.7895	0.9140	0.8570
	Grid Dropout	0.8629	0.8070	0.8886	0.8261	0.8235	0.7604	0.8531	0.7866	0.9083	0.8496
	Image Compression	0.8642	0.8087	0.8887	0.8265	0.8203	0.7570	0.8593	0.7939	0.9106	0.8527
	Median Blur	0.8644	0.8084	0.8883	0.8264	0.8158	0.7523	0.8556	0.7893	0.9120	0.8540
	Optical Distortion	0.8714	0.8169	0.8916	0.8303	0.8287	0.7641	0.8557	0.7897	0.9135	0.8564
	Piecewise Affine	0.8745	0.8227	0.8965	0.8363	0.8409	0.7788	0.8600	0.7943	0.9142	0.8573
	Posterize	0.8621	0.8075	0.8923	0.8304	0.8210	0.7575	0.8641	0.7991	0.9103	0.8528
	RBC	0.8596	0.8029	0.8870	0.8238	0.8213	0.7573	0.8510	0.7840	0.9079	0.8494
	Random Crop	0.8632	0.8078	0.8883	0.8262	0.8217	0.7575	0.8610	0.7959	0.9092	0.8515
	Random Gamma	0.8650	0.8090	0.8886	0.8258	0.8169	0.7541	0.8581	0.7923	0.9093	0.8515
	Random Snow	0.8700	0.8157	0.8922	0.8304	0.8343	0.7719	0.8690	0.8051	0.9124	0.8554
	Rotate	0.8770	0.8233	0.8954	0.8353	0.8408	0.7767	0.8491	0.7822	0.9110	0.8529
	Sharpen	0.8623	0.8068	0.8917	0.8298	0.8180	0.7543	0.8636	0.7990	0.9123	0.8552
	Shift Scale Rotate	**0.8784**	**0.8267**	**0.8993**	**0.8399**	**0.8472**	**0.7845**	0.8598	0.7943	0.9143	0.8574

For probability 0.25, the unified training set achieved better values in 9 data augmentation techniques in the MedSeg dataset, 17 in MosMed, and 15 in Zenodo. Similarly, for probability 0.3, the unified training set achieved better values in 9 data augmentation techniques in the MedSeg dataset, 16 in MosMed, and 13 in Zenodo. In both probabilities of 0.25 and 0.3, all data augmentation techniques achieved better values in the CC-CCII dataset through the unified training strategy. At a probability of 0.35, the unified training set achieved higher valued in all

Table 6.15: Results of the data augmentation evaluation when unifying the training sets (Continuation of Tables 6.11, 6.12, 6.13 and 6.14). *p* stands for probability, the blue-colored values indicate the best F-score values, and the red-colored values indicate the best IoU values. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show the techniques in which training with the combined training sets achieved a P-value lower than 0.05 when compared with training with a single training set, and the null hypothesis was rejected.

p	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.45	CLAHE	0.8524	0.7972	0.8851	0.8217	0.8177	0.7537	0.8543	0.7878	0.9081	0.8494
	Coarse Dropout	0.8666	0.8114	0.8918	0.8302	0.8333	0.7694	0.8703	0.8064	0.9117	0.8544
	Elastic Transform	0.8778	**0.8256**	0.8991	0.8392	0.8454	0.7831	0.8633	0.7981	**0.9152**	**0.8587**
	Emboss	0.8626	0.8084	0.8895	0.8273	0.8183	0.7543	0.8571	0.7909	0.9112	0.8537
	Flip	0.8722	0.8176	0.8935	0.8322	0.8339	0.7707	0.8486	0.7817	0.9084	0.8489
	Gaussian Blur	0.8612	0.8060	0.8908	0.8296	0.8188	0.7563	**0.8707**	**0.8073**	0.9131	0.8561
	Grid Distortion	0.8771	0.8253	**0.8994**	**0.8393**	0.8452	0.7826	0.8620	0.7967	0.9151	0.8585
	Grid Dropout	0.8632	0.8077	0.8900	0.8280	0.8279	0.7645	0.8646	0.7998	0.9097	0.8515
	Image Compression	0.8610	0.8054	0.8878	0.8253	0.8218	0.7582	0.8588	0.7935	0.9106	0.8527
	Median Blur	0.8652	0.8099	0.8904	0.8288	0.8181	0.7546	0.8626	0.7973	0.9113	0.8539
	Optical Distortion	0.8726	0.8199	0.8962	0.8361	0.8322	0.7693	0.8672	0.8030	0.9146	0.8585
	Piecewise Affine	0.8771	0.8253	0.8962	0.8360	0.8426	0.7792	0.8586	0.7926	0.9135	0.8564
	Posterize	0.8611	0.8054	0.8905	0.8282	0.8204	0.7570	0.8599	0.7944	0.9098	0.8518
	RBC	0.8586	0.8035	0.8888	0.8258	0.8196	0.7563	0.8532	0.7866	0.9096	0.8513
	Random Crop	0.8608	0.8064	0.8863	0.8231	0.8180	0.7538	0.8548	0.7885	0.9088	0.8505
	Random Gamma	0.8630	0.8076	0.8873	0.8240	0.8184	0.7549	0.8574	0.7917	0.9111	0.8532
	Random Snow	0.8628	0.8091	0.8926	0.8306	0.8347	0.7716	0.8681	0.8041	0.9125	0.8555
	Rotate	**0.8790**	0.8250	0.8990	0.8389	**0.8477**	**0.7841**	0.8500	0.7834	0.9105	0.8525
	Sharpen	0.8604	0.8055	0.8914	0.8298	0.8251	0.7608	0.8704	0.8069	0.9136	0.8570
	Shift Scale Rotate	0.8766	0.8239	0.8987	0.8389	0.8457	0.7819	0.8558	0.7897	0.9126	0.8555
0.5	CLAHE	0.8561	0.8002	0.8847	0.8216	0.8136	0.7513	0.8544	0.7879	0.9088	0.8503
	Coarse Dropout	0.8585	0.8031	0.8921	0.8306	0.8329	0.7692	0.8604	0.7950	0.9112	0.8535
	Elastic Transform	0.8780	0.8263	0.8991	0.8396	0.8443	0.7820	0.8622	0.7969	0.9147	0.8580
	Emboss	0.8662	0.8112	0.8894	0.8277	0.8221	0.7591	0.8646	0.7999	0.9116	0.8544
	Flip	0.8688	0.8137	0.8955	0.8341	0.8396	0.7761	0.8488	0.7818	0.9084	0.8491
	Gaussian Blur	0.8649	0.8098	0.8905	0.8287	0.8214	0.7582	0.8681	0.8040	0.9124	0.8548
	Grid Distortion	0.8782	0.8271	**0.9002**	0.8403	0.8431	0.7796	0.8555	0.7892	0.9145	0.8577
	Grid Dropout	0.8628	0.8070	0.8887	0.8263	0.8244	0.7616	0.8586	0.7926	0.9073	0.8487
	Image Compression	0.8596	0.8054	0.8879	0.8255	0.8172	0.7547	0.8595	0.7942	0.9108	0.8531
	Median Blur	0.8662	0.8113	0.8896	0.8278	0.8206	0.7565	0.8615	0.7963	0.9120	0.8547
	Optical Distortion	0.8710	0.8172	0.8958	0.8355	0.8310	0.7675	0.8647	0.7999	0.9147	**0.8586**
	Piecewise Affine	0.8800	**0.8283**	0.8994	0.8401	0.8453	0.7833	0.8636	0.7987	**0.9153**	**0.8586**
	Posterize	0.8631	0.8072	0.8915	0.8293	0.8217	0.7571	0.8613	0.7962	0.9111	0.8534
	RBC	0.8566	0.8004	0.8891	0.8265	0.8225	0.7581	0.8499	0.7827	0.9084	0.8501
	Random Crop	0.8654	0.8098	0.8887	0.8261	0.8221	0.7585	0.8600	0.7948	0.9101	0.8523
	Random Gamma	0.8644	0.8081	0.8889	0.8261	0.8203	0.7568	0.8569	0.7910	0.9095	0.8514
	Random Snow	0.8664	0.8115	0.8911	0.8290	0.8362	0.7732	0.8675	0.8030	0.9122	0.8550
	Rotate	**0.8806**	0.8273	0.8986	0.8384	0.8470	0.7826	0.8487	0.7820	0.9098	0.8514
	Sharpen	0.8641	0.8093	0.8903	0.8283	0.8240	0.7601	**0.8704**	**0.8066**	0.9136	0.8568
	Shift Scale Rotate	0.8780	0.8261	0.8996	**0.8405**	**0.8477**	**0.7852**	0.8586	0.7929	0.9137	0.8568

augmentation techniques in CC-CCII, 10 data augmentation techniques in the MedSeg dataset, 15 in MosMed, and 14 in Zenodo. At a probability of 0.4, the unified training set achieved higher valued in 19 augmentation techniques in CC-CCII, 7 data augmentation techniques in the MedSeg dataset, 15 in MosMed, and 14 in Zenodo.

At a probability of 0.45, the unified training set achieved higher valued in all augmentation techniques in CC-CCII, 12 data augmentation techniques in the MedSeg dataset, 14 in MosMed,

and 12 in Zenodo. At a probability of 0.5, the unified training set achieved higher valued in all augmentation techniques in CC-CCII, 9 data augmentation techniques in the MedSeg dataset, 17 in MosMed, and 12 in Zenodo. Besides the promising results achieved with the unified training strategy and data augmentation in CC-CCII, MedSeg, MosMed, and Zenodo, the unified training set did not improve the performance of the Ricord1a dataset, which could be attributed to the balancing approach that prioritized the small datasets, which performed poorly in the traditional training strategy compared to Ricord1a.

Moreover, data augmentation techniques were consistently more effective with this training strategy in all ten probabilities evaluated. For a probability of 0.05, Elastic Transform and Piecewise Affine achieved a higher F-score on the CC-CCII dataset. In the MedSeg dataset, seven data augmentation techniques achieved a higher F-score. In the MosMed dataset, twelve data augmentation techniques achieved a higher F-score, and Piecewise Affine increased the F-score by 2%. As explained in Section 6.2.1, applying data augmentation to the Zenodo dataset using standard training sets did not improve the results, with only one data augmentation technique improving the F-score. However, twelve data augmentation techniques achieved a higher F-score on the Zenodo dataset with this training strategy. In the Ricord1a dataset, four data augmentation techniques achieved a higher F-score, and Piecewise Affine also increased the F-score by 2%.

For a probability of 0.1, 5 data augmentation techniques improved the F-score on the CC-CCII, 7 on the MedSeg, 13 on the MosMed, 4 on the Ricord1a, and 12 on the Zenodo. Also, in MosMed, the Elastic Transform, Grid Distortion, Piecewise Affine, and Rotate increased the F-score by 2%. For a probability of 0.15, 6 data augmentation methods improved the F-score on the CC-CCII, 12 on the MedSeg, 16 on the MosMed, 9 on the Ricord1a, and 13 on the Zenodo. In MosMed, six data augmentation techniques increased the F-score by 2%. For a probability of 0.2, 5 data augmentation techniques improved the F-score on CC-CCII, 10 on MedSeg, 12 on the MosMed, 8 on the Ricord1a, and 13 on Zenodo. In MosMed, six data augmentation techniques increased the F-score by 2%, and the Shift Scale Rotate technique increased the F-score by 3%. For a probability of 0.25, 6 data augmentation techniques improved the F-score on CC-CCII, 10 on the MedSeg, 16 on MosMed, 10 on Ricord1a, and 15 on Zenodo. In the MosMed dataset, three data augmentation techniques increased the F-score by 3%. For a probability of 0.3, 7 data augmentation techniques improved the F-score on CC-CCII, 12 on MedSeg, 14 on MosMed, 12 on Ricord1a, and 13 on Zenodo. In the MosMed dataset, four data augmentation techniques increased the F-score by 3%.

For a probability of 0.35, 7 data augmentation techniques improved the F-score on CC-CCII, 13 on MedSeg, 14 on MosMed, 10 on Ricord1a, and 13 on Zenodo. For a probability of 0.4, 7 data augmentation techniques improved the F-score on CC-CCII, 13 on MedSeg, 14 on MosMed, 7 on Ricord1a, and 13 on Zenodo. For a probability of 0.45, 7 data augmentation techniques improved the F-score on CC-CCII, 14 on MedSeg, 12 on MosMed, 9 on Ricord1a, and 13 on Zenodo. For a probability of 0.5, 7 data augmentation techniques improved the F-score on CC-CCII, 12 on MedSeg, 15 on MosMed, 9 on Ricord1a, and 13 on Zenodo. The probability 0.3 yielded better outcomes, considering the number of techniques that improved the F-score values.

The best result per dataset was: CC-CCII with Rotate applied with a probability of 0.5, reaching an F-score of 0.8806; MedSeg with Grid Distortion applied with a probability of 0.5, achieving an F-score of 0.9002; MosMed with Rotate applied with a probability of 0.45 and Shift Scale Rotate applied with a probability of 0.5, attaining an F-score of 0.8477; Ricord1a with Gaussian Blur applied with a probability of 0.15, achieving an F-score of 0.8729; Zenodo with Piecewise Affine applied with a probability of 0.5, reaching an F-score of 0.9153. Thus, in the CC-CCII, MedSeg, MosMed and Zenodo datasets the highest F-score value was attained through a spatial transformation, while in Ricord1a it was achieved with a color transformation. With the

unified training set, the spatial operations also provided the best results, confirming that these operations are the best choices for the approached problem because they encourage shape variation of the lesion regions. The Ricord1a dataset is an exception, with the Gaussian Blur achieving the highest F-score value, suggesting that this dataset is more sensitive to color operations.

In general, the data augmentation techniques were consistently more effective (i.e., they reached better F-score and IoU values) with this training strategy in four of five evaluated datasets. The Ricord1a was the only dataset which did not achieved improvements with the unified training set. This happened due to the balancing approach that prioritized the small datasets that achieved poor results in the traditional training strategy when compared with the Ricord1a. Also, although applying data augmentation techniques with a probability of 0.5 in the first evaluation did not show interesting results, the same probability produced the highest number of data augmentation techniques that improved the F-score when the datasets were combined into a single training set. Furthermore, the slight difference achieved by the data augmentation process in the individual training sets resulted in a small difference between the probabilities applied, making it unclear which is the optimal probability in this case. Meanwhile, applying data augmentation in the unified training set achieved overall higher results, clarifying that increasing the probability of the data augmentation techniques generates better results, opposing previously works from literature (Müller et al., 2021).

6.3 PROPOSED DATA AUGMENTATION EVALUATION

In the preceding sections, a comprehensive comparison of segmentation networks and data augmentation techniques was conducted for the COVID-19 segmentation problem. The analysis revealed that data augmentation is crucial for addressing the challenges encountered in this particular problem. Due to the limited number of available images and the critical issue of class imbalance in the COVID-19 segmentation datasets, a thorough examination of generic data augmentation techniques was undertaken. However, the results demonstrated that these techniques slightly improved segmentation performance.

Consequently, the next phase of this research focused on developing and evaluating a domain-specific data augmentation technique specifically designed to address the COVID-19 segmentation problem. This technique is based on visual salience. The evaluation of this approach is divided into three steps, as outlined in the following Subsections: 6.3.1, 6.3.2, and 6.3.3. In the first step of the proposed augmentation technique, two Generative Adversarial Networks (GANs) are employed to generate healthy CT images. The second step involves utilizing an encoder-decoder segmentation network to generate segmentation masks for the lung regions in the healthy CT images generated by the GAN. This process enables the isolation and delineation of the lung regions. Finally, the third step introduces new lesions within the segmented lung regions, creating new CT images with COVID-19 lesions.

6.3.1 Healthy CT Generation

The initial phase of the proposed data augmentation method involves generating healthy CT samples using GANs. To accomplish this, two GAN models, Stargan (Choi et al., 2020) and Stylegan (Karras et al., 2020), were evaluated. These models were selected based on their promising results during this work's development (Ibrahim et al., 2024). The evaluation of Stargan commenced by training the network for one million iterations, each involving a batch of four samples. The evaluation of Stargan utilized the FID metric, and the corresponding training curves are depicted in Figure 6.9. The figure includes two curves: FID latent (shown in red) and

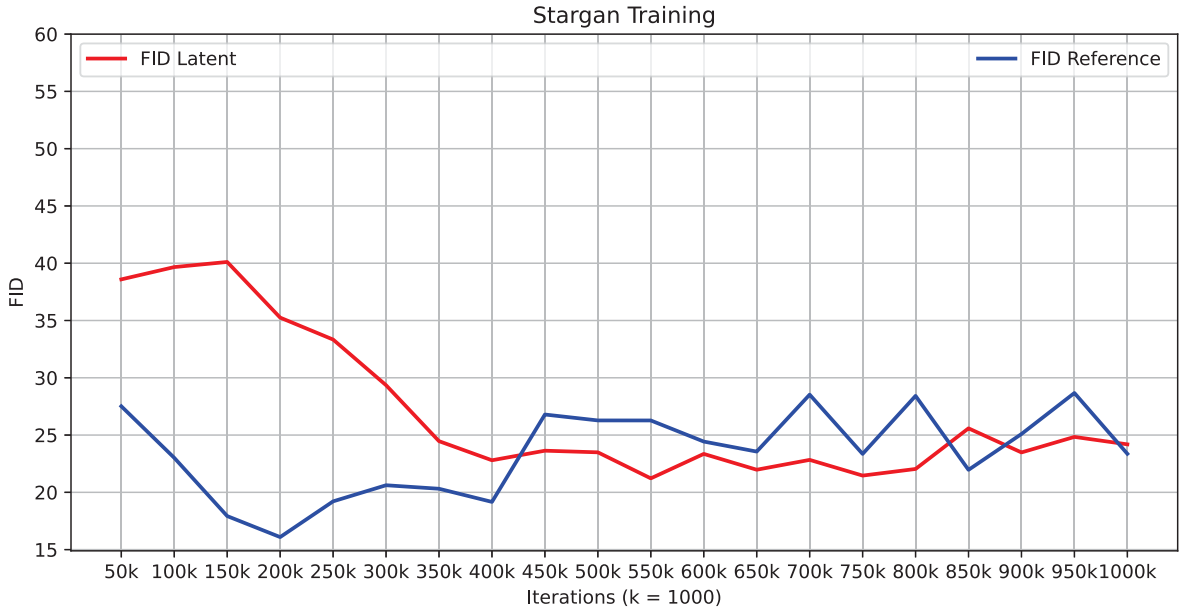


Figure 6.9: The Stargan training curves. Two curves are presented: Fréchet Inception Distance (FID) latent (in red) and FID reference (in blue). FID latent represents the FID values calculated using images generated with random latent vectors, while FID reference represents the FID values calculated using images generated with reference images. The FID reference curve demonstrated superior values in the first four hundred thousand iterations. After that, both latent and reference curves achieved similar results, with a slight gain of the latent curve.

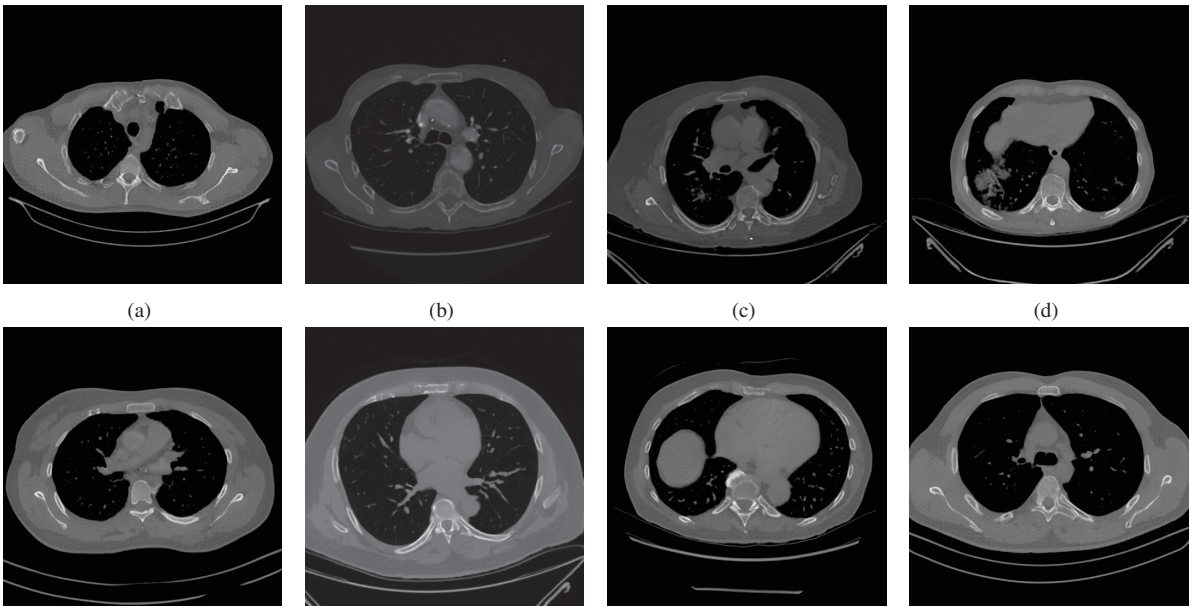


Figure 6.10: Examples of healthy lung images generated by the Stargan. The images were generated with the weight of the iteration two hundred thousand since it is the iteration that achieved the lower FID reference value.

FID reference (shown in blue). FID latent represents the FID values calculated using images generated with random latent vectors, while FID reference represents the FID values calculated using images generated with reference images. The FID reference curve demonstrated superior values in the first four hundred thousand iterations. After that, both latent and reference curves achieved similar results, with a slight gain of the latent curve.

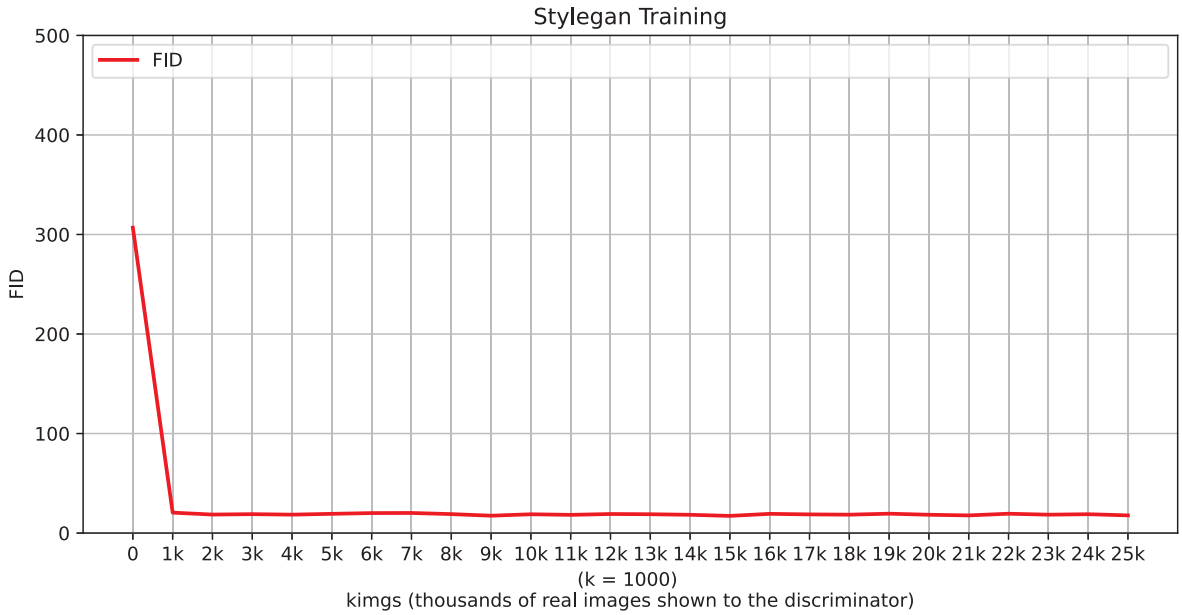


Figure 6.11: The Stylegan training curve. The Stylegan was trained twenty-five thousand kimgs, a metric that counts the number of thousand images shown to the network. The Stylegan achieved the lowest FID at kimg three thousand six hundred, not presented here due to space in the figure.

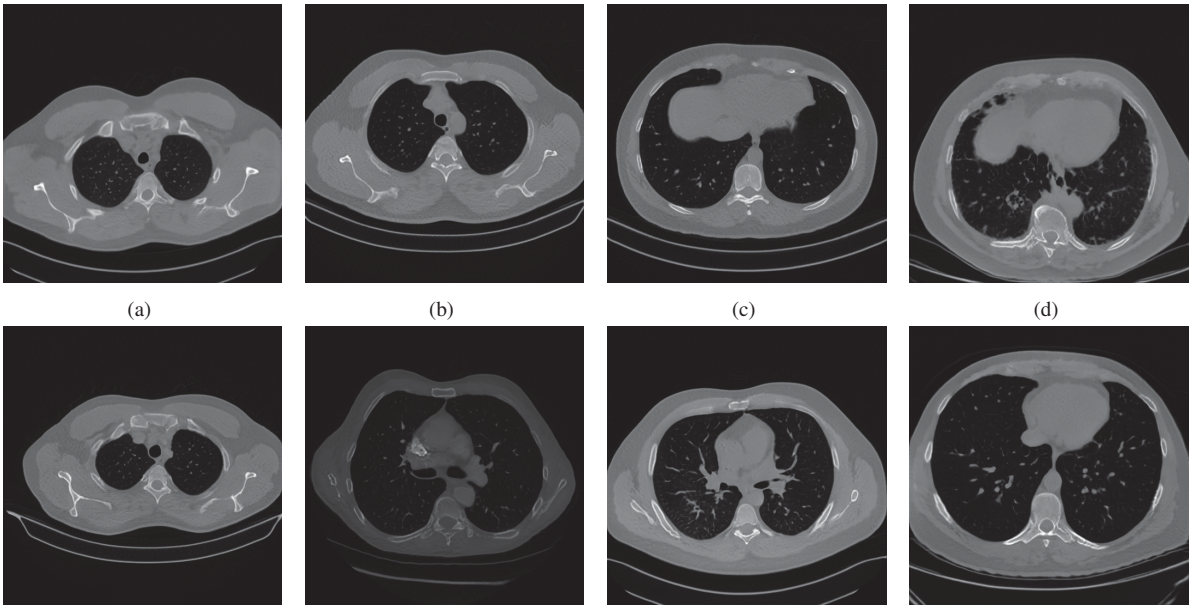


Figure 6.12: Examples of healthy images generated with the Stylegan. The images generated with the Stylegan are also promising to be used in the next steps of the data augmentation framework.

As Stargan is designed explicitly for image-to-image translation, it necessitates the utilization of two distinct domains during the training process. In the context of this study, the two domains correspond to healthy lung images and COVID-19-infected lung images. The COVID-19-infected lung images were sourced from the Ricord1a dataset, while the healthy lung images were sourced from Ricord1b. Figure 6.10 presents eight examples of healthy lung images generated by the Stargan. The images were generated with the weight of the iteration two hundred thousand, since it is the iteration that achieved the lower FID reference value (see Figure 6.9). In

general, the images generated by the Stargan are promising to be used in the next steps of the data augmentation framework.

The second GAN model evaluated was the Stylegan, and its training curve is depicted in Figure 6.11. The training of Stylegan involved twenty-five thousand kimgs, representing the count of a thousand images shown to the network. Stylegan achieved its lowest FID score at kimg three thousand six hundred, which is not included in Figure 6.11 due to space limitations. Unlike Stargan, Stylegan only required a single dataset for training the network. The Ricord1b dataset, which provided healthy lung images, was employed in this case. Figure 6.12 showcases eight examples of healthy images generated using Stylegan. These generated images exhibit promising potential for utilization in the subsequent stages of the data augmentation framework.

As demonstrated, both GAN models achieved remarkable results by generating synthetic healthy lung images that closely resemble real healthy lung images. However, Stargan attained slightly superior performance, yielding a GAN score of 16.09, while Stylegan achieved a GAN score of 16.78. The next phase of the proposed data augmentation process involves employing a segmentation network to generate a segmentation mask of the lung regions in the images generated by both GAN models.

6.3.2 Lung Segmentation

The second phase of the proposed data augmentation approach involves employing a segmentation network to generate lung segmentation masks for the healthy CT images generated by the GAN models. In this step, a dedicated segmentation model receives the images from the GAN models and generates accurate lung segmentation masks for these newly generated images. These lung segmentation masks ensure that any subsequently added lesions are placed exclusively within the lung regions of the images.

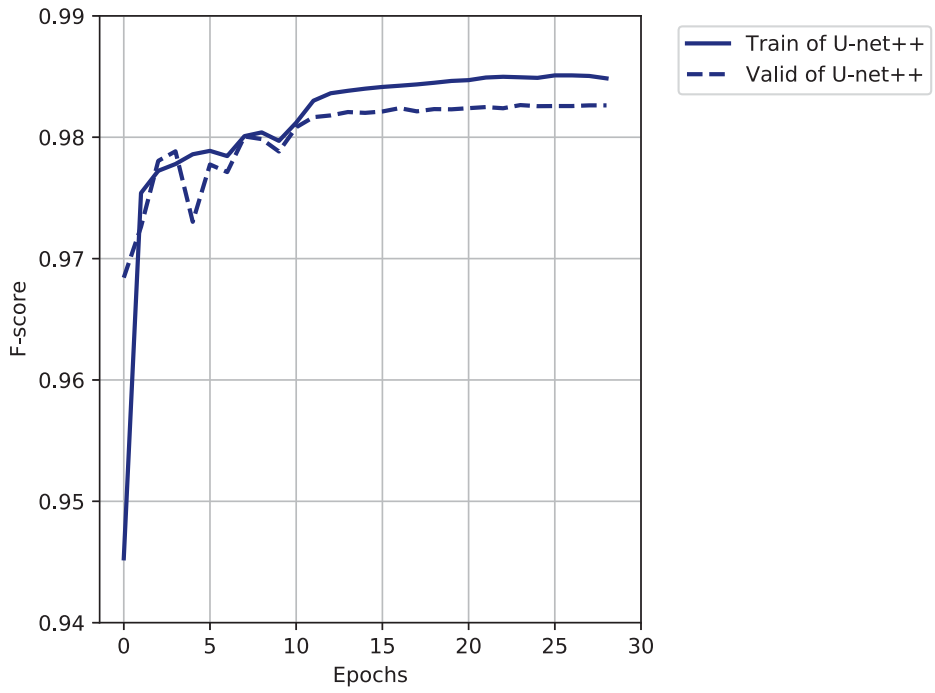


Figure 6.13: The average F-score curves of the training and validation sets of the lung segmentation network. Even for a mix of two datasets, the networks achieved impressive results, achieving an average F-score between 98% and 99%.

Based on the findings highlighted in Section 6.1, the selected encoder-decoder architecture for this step is the combination of RegNetx-002 as the encoder and U-Net++ as the decoder. This combination demonstrated outstanding performance in the experiments detailed in Section 6.1. Furthermore, the utilization of RegNetx-002 as the encoder offers the advantage of being a compact network (as indicated in Table 6.1), resulting in faster training and evaluation times. The architecture was trained for up to 100 epochs with the patience of 10 epochs. The initial learning rate was set to 0.001 and was divided by 10 every 10 epochs. The CC-CCII and Zenodo datasets were merged to train the network. These datasets were chosen as they were the only ones with labeled lung region data. The experiment was further validated using a five-fold cross-validation strategy, while no data augmentation techniques were applied during this process. Figure 6.13 presents the average F-score curves of the training and validation sets of the lung segmentation network. It is worth pointing out that the lung segmentation network was trained and validated with a mix of two datasets, CC-CCII and Zenodo, and was tested with images generated by GANs trained in another dataset: Ricord1a and Ricord1b. Even for a mix of two datasets, the networks achieved impressive results, achieving an average F-score between 98% and 99%.

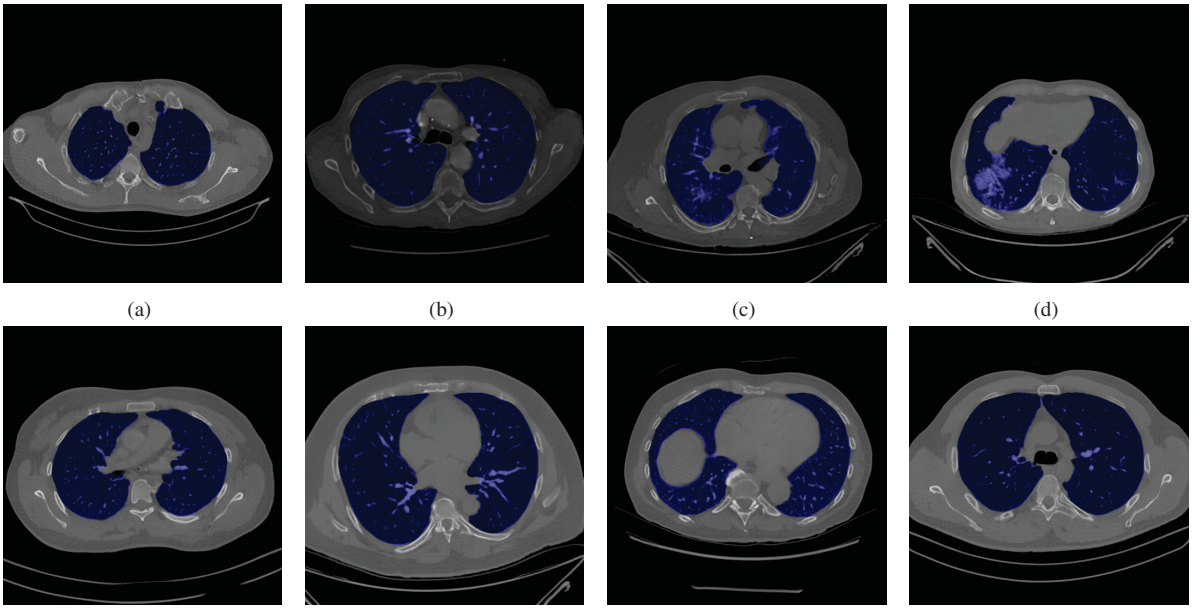


Figure 6.14: Examples of lung segmentation in the images generated by the Stargan. The blue colored regions are the lung regions segmented by the network.

The generated images from Stargan and Stylegan served as the test sets for the segmentation network. Consequently, during the training and validation stages, the segmentation network had no exposure to the images produced by the GAN models. As the test sets lacked ground-truth labels for quantitative analysis, the evaluation of the segmentation network's performance was primarily qualitative. Figure 6.14 showcases eight examples of lung segmentation in the images generated by Stargan, while Figure 6.15 displays eight examples of lung segmentation in the images generated by Stylegan. The lung regions segmented by the network are highlighted in blue. As illustrated in both figures, the segmentation network demonstrated promising qualitative results, with the generated segmentation masks closely resembling the expected outcome of a manual labeling process.

In this phase, the segmentation model was also utilized to generate segmentation masks for the lung regions of the images in the Ricord1a dataset. These segmentation masks play

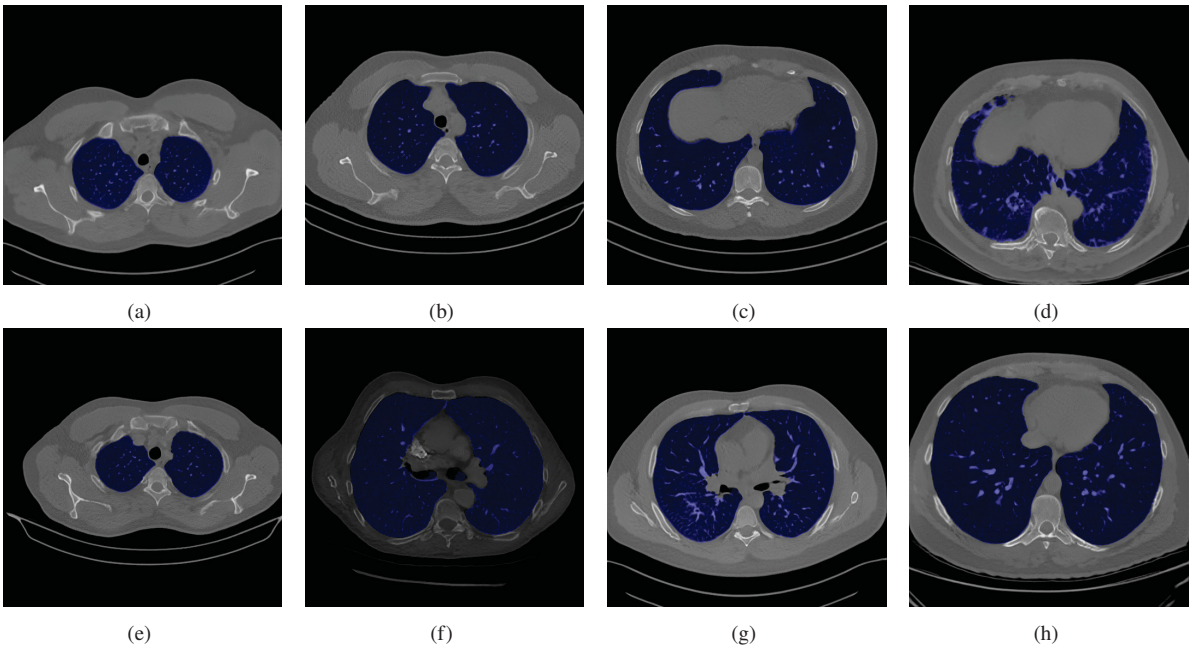


Figure 6.15: Examples of lung segmentation in the images generated by the Stylegan. The blue colored regions are the lung regions segmented by the network.

a crucial role in the subsequent step of the workflow, as they are employed to align the lung regions of the dataset images with the lung regions of the images generated by the GAN models. Figure 6.16 displays eight examples of lung segmentation of images in the Ricord1a dataset. As anticipated, the segmentation model produced promising segmentation masks that closely resemble the expected masks generated through a manual labeling process.

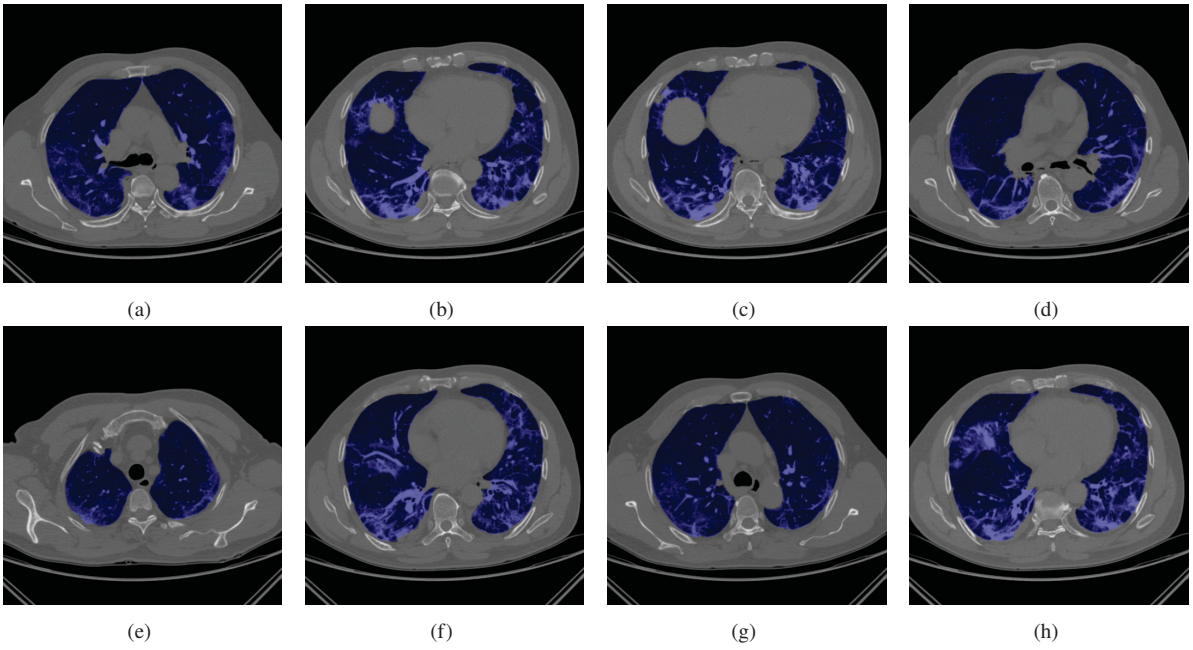


Figure 6.16: Examples of lung segmentation in the images from Ricord1a dataset. The blue colored regions are the lung regions segmented by the network.

6.3.3 Adding New Lesions

In this step, two versions of the proposed augmentation is evaluated. The first version, called DACov, incorporates COVID-19 lesions in the healthy CT images through a random algorithm, while the second version, called SalDACov, incorporates COVID-19 lesions in the healthy CT images through a salience based algorithm.

6.3.3.1 Adding New Lesions - DACov version

The subsequent step involves incorporating COVID-19 lesions within the lung regions of CT scans generated by the GANs (Subsection 6.3.1). In this DACov version, the match between the healthy image generated by the GANs and the lesion from the dataset is made randomly. The resulting images, displayed in Figure 6.17 for a Stargan-generated image and Figure 6.18 for Stylegan-generated images, showcase the incorporation of COVID-19 lesions. Sub-figures (a) to (d) exhibit examples of GAN-generated images with COVID-19 lesions in both sets of images. At the same time, Sub-figures (e) to (h) present the corresponding segmentation masks, with the lesion regions depicted in red. As anticipated, owing to the effective lung segmentation step outlined in Subsection 6.3.2, the COVID-19 lesions are accurately positioned within the lung regions.

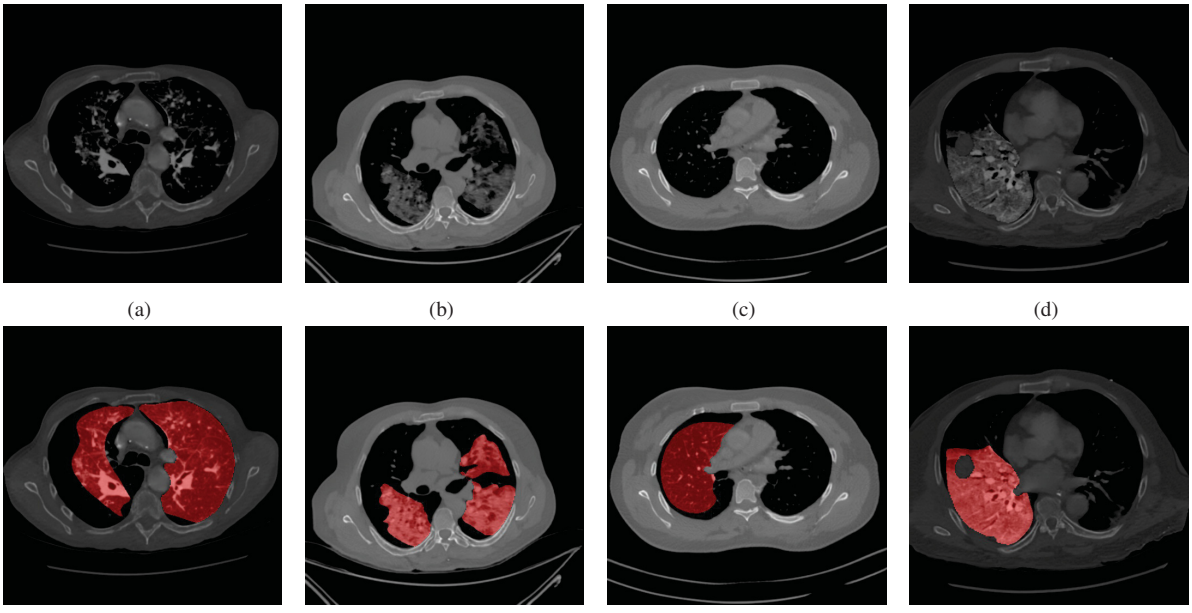


Figure 6.17: Examples of images generated by the Stargan after the addition of COVID-19 lesions. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in red.

6.3.3.2 Evaluation - DACov

The DACov version of the proposed data augmentation technique was evaluated using the unified training strategy outlined in Subsection 6.2.2. That means that the proposed data augmentation was applied after merging and balancing the training sets into a single, large training set. So, this evaluation is a continuation of the evaluation presented in Subsection 6.2.2. However, unlike previously tested data augmentation techniques which were applied online, the newly proposed augmentation approach is not suitable for this strategy due to its multi-step image generation

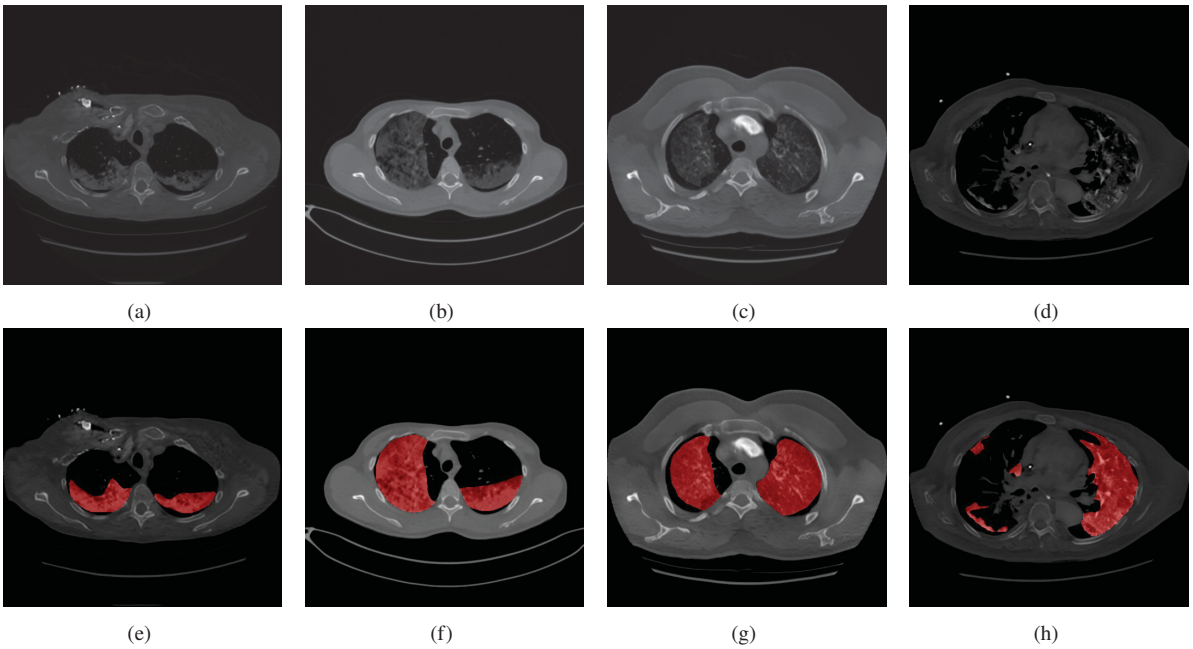


Figure 6.18: Examples of images generated by the Stylegan after the addition of COVID-19 lesions. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in red.

process. Consequently, the proposed data augmentation assessment was conducted offline. Under this methodology, the data augmentation was performed before the network training phase, increasing the volume of training images available.

The online data augmentation strategy has a probability of applying the augmentation technique within each batch presented to the network. As elaborated in Section 6.2, the data augmentation techniques were evaluated using the following probabilities: 0.05, 0.1, 0.15, 0.2, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, and 0.5. This implies that roughly 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, and 50% of the training set presented to the network consisted of artificially generated data from the augmentation techniques. To maintain a consistent quantity of artificially augmented data introduced to the network, the number of augmented training images generated with the proposed augmentation approach follows the same probabilities. That means that the training set was augmented in 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, and 50%.

Table 6.16 presents the evaluation results for the proposed data augmentation technique in the unified training set. The proposed augmentation was evaluated with two variations (with and without flip). First, the lesions from the dataset were added to the healthy lung images generated by the GANs without altering the lesion images. Then, in the second variation, the lesion images were horizontally flipped before being added to the healthy image generated by the GANs. The Wilcoxon Signed-Rank Test test was utilized to perform statistical analysis between the segmentation model trained applying data augmentation technique and training the segmentation model without applying data augmentation. The highlighted values in green demonstrate the data augmentation techniques that yielded P-values less than 0.05, thus rejecting the null hypothesis, indicating a statistical difference and the achieved results are superior to those without data augmentation. In general, incorporating lesion flipping led to a slight enhancement compared to images generated without lesion flipping. Furthermore, there was a minimal disparity in the results obtained from images generated using Stargan and Stylegan.

Table 6.16: Results of the random version of the proposed data augmentation evaluation when unifying the training sets. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). Two variations of the proposed algorithm were evaluated. The + F indicates the variation in which the lesion images were horizontally flipped before being added to the healthy image generated by the GANs.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8604	0.8054	0.8879	0.8251	0.8179	0.7546	0.8671	0.8033	0.9100	0.8522
	Stargan+ F	0.8596	0.8053	0.8877	0.8249	0.8196	0.7559	0.8610	0.7962	0.9099	0.8521
	Stylegan	0.8643	0.8088	0.8885	0.8259	0.8174	0.7544	0.8590	0.7939	0.9098	0.8518
	Stylegan+ F	0.8646	0.8092	0.8887	0.8261	0.8211	0.7584	0.8696	0.8065	0.9106	0.8530
0.1	Stargan	0.8657	0.8100	0.8852	0.8224	0.8163	0.7530	0.8577	0.7924	0.9091	0.8508
	Stargan+ F	0.8635	0.8084	0.8888	0.8262	0.8187	0.7562	0.8648	0.8006	0.9103	0.8525
	Stylegan	0.8657	0.8101	0.8873	0.8244	0.8182	0.7545	0.8561	0.7906	0.9085	0.8502
	Stylegan+ F	0.8656	0.8087	0.8914	0.8291	0.8208	0.7576	0.8614	0.7967	0.9093	0.8514
0.15	Stargan	0.8608	0.8059	0.8852	0.8226	0.8225	0.7594	0.8595	0.7950	0.9095	0.8515
	Stargan+ F	0.8631	0.8072	0.8887	0.8261	0.8186	0.7551	0.8681	0.8047	0.9100	0.8522
	Stylegan	0.8658	0.8104	0.8888	0.8258	0.8213	0.7576	0.8640	0.8003	0.9105	0.8527
	Stylegan+ F	0.8641	0.8096	0.8897	0.8270	0.8254	0.7617	0.8708	0.8077	0.9112	0.8537
0.2	Stargan	0.8637	0.8075	0.8889	0.8264	0.8217	0.7584	0.8684	0.8050	0.9104	0.8527
	Stargan+ F	0.8673	0.8120	0.8882	0.8261	0.8197	0.7569	0.8693	0.8061	0.9112	0.8536
	Stylegan	0.8646	0.8095	0.8893	0.8264	0.8168	0.7535	0.8621	0.7979	0.9096	0.8516
	Stylegan+ F	0.8639	0.8096	0.8892	0.8265	0.8272	0.7638	0.8733	0.8106	0.9105	0.8530
0.25	Stargan	0.8642	0.8080	0.8874	0.8250	0.8188	0.7560	0.8685	0.8053	0.9106	0.8527
	Stargan+ F	0.8660	0.8098	0.8891	0.8271	0.8254	0.7619	0.8739	0.8118	0.9106	0.8531
	Stylegan	0.8647	0.8089	0.8905	0.8285	0.8212	0.7580	0.8714	0.8086	0.9115	0.8541
	Stylegan+ F	0.8659	0.8100	0.8887	0.8258	0.8185	0.7553	0.8683	0.8049	0.9103	0.8526
0.3	Stargan	0.8665	0.8109	0.8877	0.8255	0.8190	0.7555	0.8654	0.8018	0.9096	0.8516
	Stargan+ F	0.8626	0.8074	0.8876	0.8248	0.8227	0.7595	0.8703	0.8074	0.9104	0.8525
	Stylegan	0.8649	0.8094	0.8893	0.8270	0.8198	0.7566	0.8666	0.8031	0.9101	0.8526
	Stylegan+ F	0.8684	0.8124	0.8884	0.8260	0.8241	0.7605	0.8688	0.8054	0.9104	0.8525
0.35	Stargan	0.8647	0.8089	0.8850	0.8225	0.8228	0.7595	0.8650	0.8013	0.9097	0.8516
	Stargan+ F	0.8651	0.8095	0.8909	0.8286	0.8242	0.7608	0.8732	0.8107	0.9116	0.8541
	Stylegan	0.8655	0.8095	0.8868	0.8242	0.8216	0.7580	0.8628	0.7988	0.9105	0.8524
	Stylegan+ F	0.8688	0.8136	0.8899	0.8273	0.8271	0.7638	0.8769	0.8150	0.9122	0.8551
0.4	Stargan	0.8642	0.8083	0.8885	0.8266	0.8231	0.7605	0.8744	0.8122	0.9117	0.8542
	Stargan+ F	0.8652	0.8088	0.8902	0.8285	0.8280	0.7648	0.8726	0.8099	0.9115	0.8542
	Stylegan	0.8629	0.8076	0.8891	0.8274	0.8250	0.7624	0.8769	0.8152	0.9125	0.8557
	Stylegan+ F	0.8660	0.8105	0.8913	0.8294	0.8287	0.7659	0.8768	0.8152	0.9125	0.8556
0.45	Stargan	0.8644	0.8102	0.8915	0.8297	0.8239	0.7610	0.8752	0.8131	0.9108	0.8532
	Stargan+ F	0.8627	0.8078	0.8886	0.8266	0.8240	0.7600	0.8737	0.8116	0.9110	0.8534
	Stylegan	0.8665	0.8111	0.8866	0.8239	0.8196	0.7567	0.8617	0.7975	0.9097	0.8520
	Stylegan+ F	0.8674	0.8115	0.8900	0.8280	0.8259	0.7629	0.8782	0.8165	0.9120	0.8548
0.5	Stargan	0.8592	0.8044	0.8893	0.8271	0.8248	0.7608	0.8719	0.8092	0.9107	0.8529
	Stargan+ F	0.8646	0.8087	0.8914	0.8292	0.8303	0.7669	0.8782	0.8168	0.9123	0.8553
	Stylegan	0.8643	0.8083	0.8885	0.8258	0.8223	0.7591	0.8700	0.8073	0.9106	0.8530
	Stylegan+ F	0.8669	0.8110	0.8907	0.8288	0.8293	0.7670	0.8783	0.8167	0.9123	0.8552

In contrast to the baseline, the proposed data augmentation technique demonstrated better F-scores across all five datasets. Specifically, within the Ricord1a dataset (previously identified as more susceptible to color alterations), this augmentation approach yielded the highest F-score among all assessed methods using the unified training strategy when probabilities

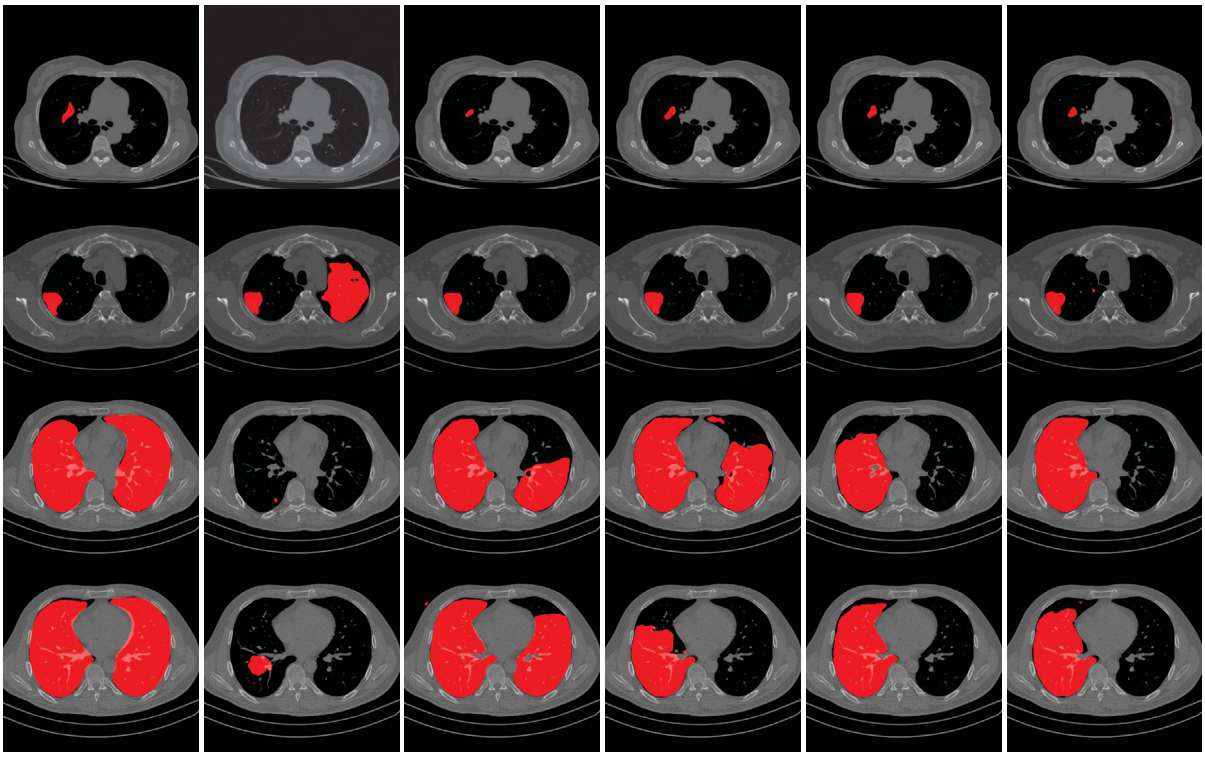


Figure 6.19: Qualitative comparison between the proposed data augmentation and the baseline without data augmentation. With the proposed data augmentation, the segmentation network found the lesion region closer to the expected in the ground truth.

were set at 0.2, 0.25, 0.35, 0.4, 0.45, and 0.5. Also, it achieved competitive results compared to the traditional approaches on the CC-CCII, MedSeg, MosMed, and Zenodo datasets, previously identified as more susceptible to shape alterations. The underlying hypothesis suggests that integrating real lesions (comprising a relatively small portion of pixels) onto entirely synthetic backgrounds (constituting the majority of pixels) fosters a heightened divergence in both color and shape, demonstrating greater effectiveness for this particular problem compared to conventional methods concentrating solely on a single aspect.

Moreover, in certain instances, the newly generated samples led to a remarkable enhancement in mask quality, as illustrated in Figure 6.19. In the first row, the segmentation network trained without data augmentation failed to identify any lesion, a critical scenario with profound biological implications. In the subsequent example depicted in the second row, the absence of data augmentation resulted in the segmentation network displaying a meager recall value, thus predicting a notably extensive false positive regions. The precision was notably low within the third and fourth rows (images where the lesions almost entirely covered the lungs), manifesting numerous false negative pixels and consequently yielding a low F-score value.

In medical problems, false negative outcomes typically represent the most critical scenario, substantially elevating patient risks as indicated by several studies (Woloshin et al., 2020; Wikramaratna et al., 2020; Kanji et al., 2021). In these instances, the proposed data augmentation strategy is pivotal in enabling the network to identify a segmentation mask that closely aligns with the ground truth, thereby reducing false positive outcomes. Additionally, as depicted in Figure 6.19, the four variations of the proposed technique consistently yielded improved outcomes using the same network architecture, outperforming the results achieved

without integrating data augmentation. Consequently, the diversity introduced by the proposed technique improved the generalization in some cases.

6.3.3.3 Adding New Lesions - SalDACov

In the SalDACov version of the proposed augmentation, the random matching algorithm that chooses the lesions from the dataset is replaced by a visual salience-based algorithm to guide the choice of the lesions that will be added to each lung. In total, nine versions of the proposed augmentation were evaluated considering the lung side and the salience distance between the healthy image generated by the GAN and the lesion image:

- The lesions are placed in the same lung side from the original image from the dataset and the lesions are chosen considering the maximum salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.17.
- The lesions are placed in the same lung side from the original image from the dataset and the lesions are chosen considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.18.
- The lesions are placed in the same lung side from the original image from the dataset and the lesions are chosen considering a random salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.19.
- The lesions are placed in the opposite lung side from the original image from the dataset and the lesions are chosen considering the maximum salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.20.
- The lesions are placed in the opposite lung side from the original image from the dataset and the lesions are chosen considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.21.
- The lesions are placed in the opposite lung side from the original image from the dataset and the lesions are chosen considering a random salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.22.
- The lesions are placed in a random lung side from the original image from the dataset and the lesions are chosen considering the maximum salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.23.
- The lesions are placed in a random lung side from the original image from the dataset and the lesions are chosen considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.24.

- The lesions are placed in a random lung side from the original image from the dataset and the lesions are chosen considering a random salience distance between the image generated by the GAN and the lesion image from the dataset. The results are presented on Table 6.25.

Figure 6.20 and Figure 6.21 showcase the images generated by StarGAN and StyleGAN, respectively, following the inclusion of COVID-19 lesions with the SalDACov version of the proposed augmentation. These lesions were selected based on the maximum salience distance between the GAN's generated image and the dataset's lesion image. This approach resulted in new training samples emphasizing lesions in small-sized areas.

Figure 6.22 and Figure 6.23 showcase the images generated by StarGAN and StyleGAN, respectively, following the inclusion of COVID-19 lesions with the SalDACov version of the proposed augmentation. These lesions were selected based on the minimum salience distance between the GAN's generated image and the dataset's lesion image. This approach resulted in new training samples emphasizing lesions in small-sized areas.

Figure 6.24 and Figure 6.25 showcase the images generated by StarGAN and StyleGAN, respectively, following the inclusion of COVID-19 lesions with the SalDACov version of the proposed augmentation. These lesions were selected based on a random salience distance between the GAN's generated image and the dataset's lesion image. This approach resulted in new training samples emphasizing lesions in small-sized areas.

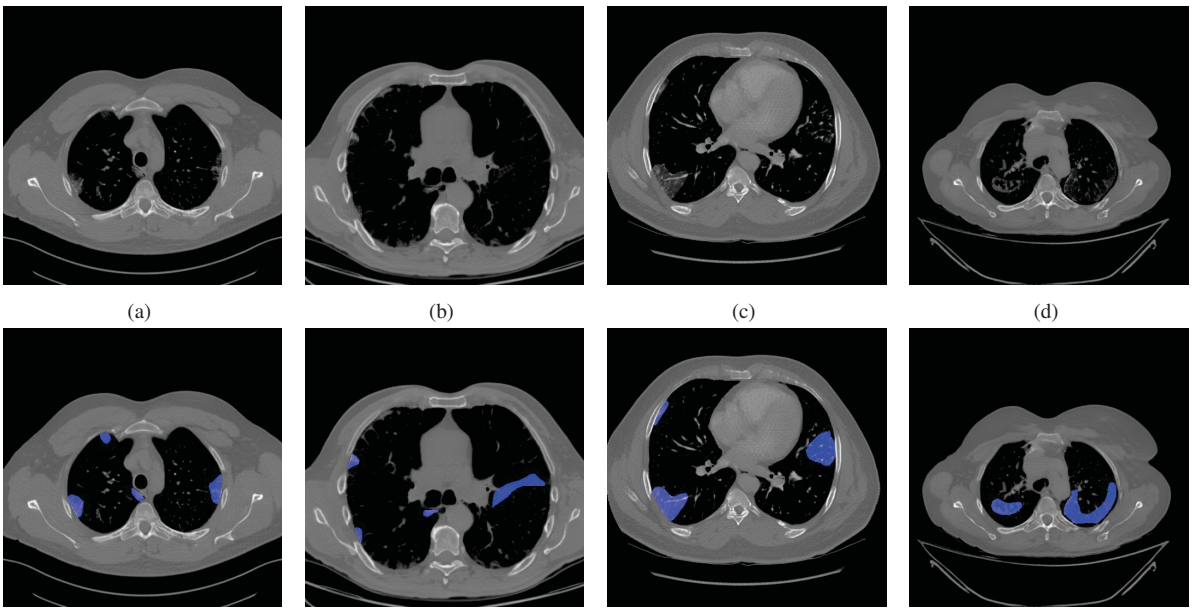
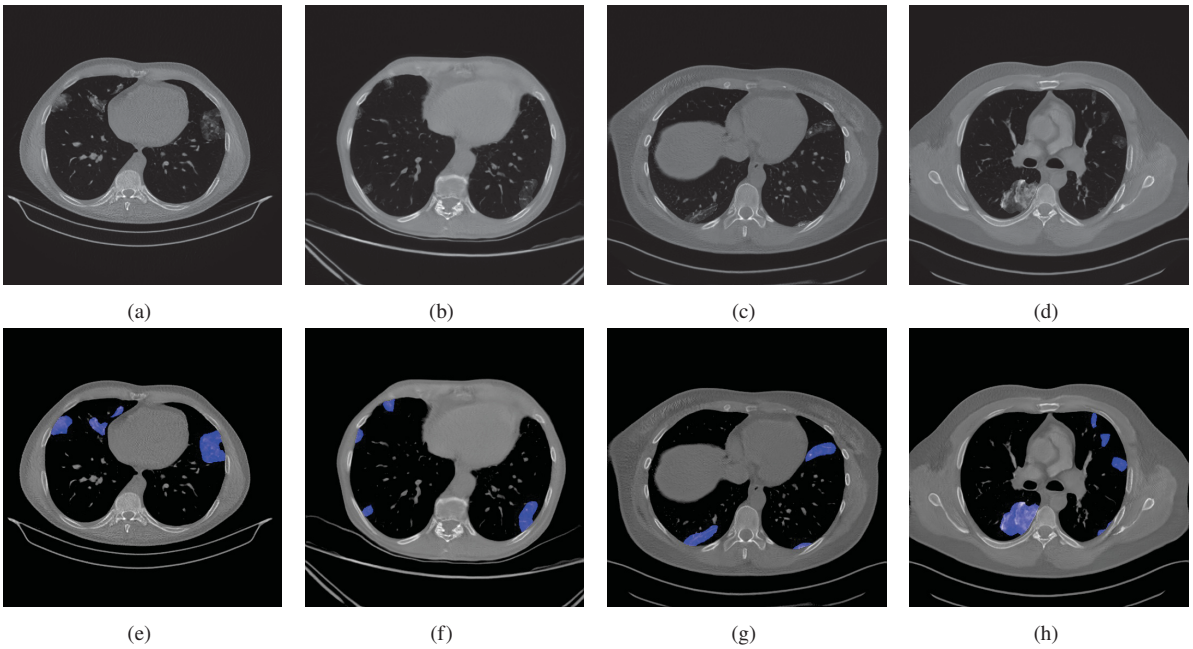


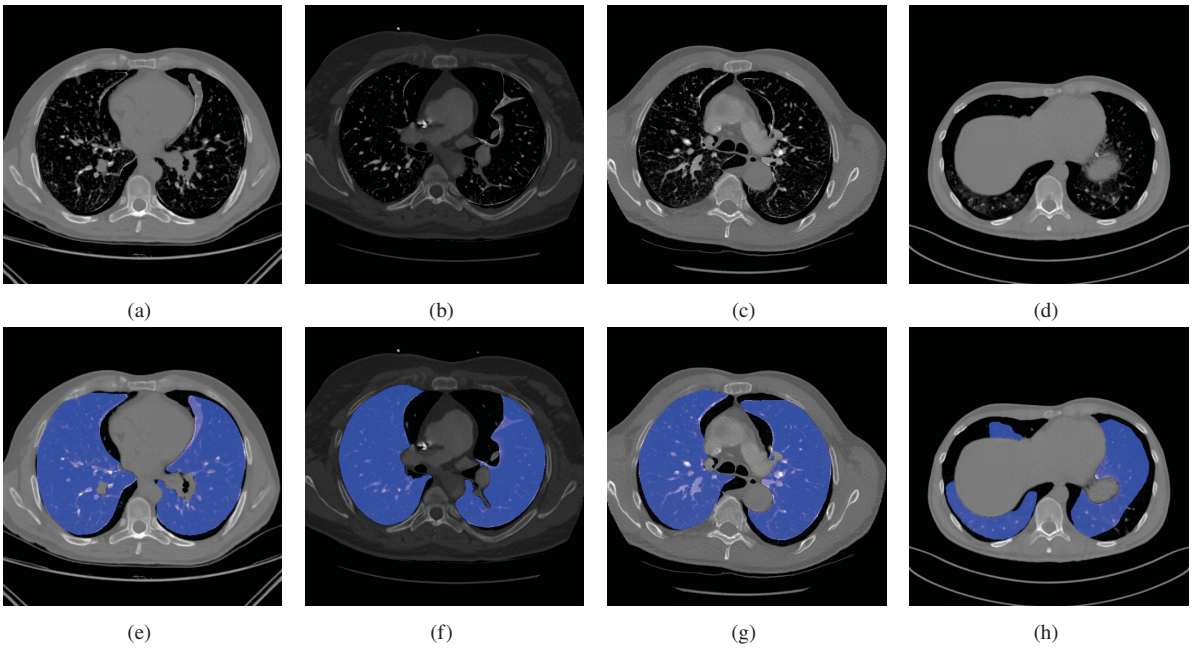
Figure 6.20: Examples of images generated by the Stargan after the addition of COVID-19 lesions chosen considering the maximum salience distance between the image generated by the GAN and the lesion image from the dataset. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in blue.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 6.21: Examples of images generated by the Stylegan after the addition of COVID-19 lesions chosen considering the maximum salience distance between the image generated by the GAN and the lesion image from the dataset. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in blue.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 6.22: Examples of images generated by the Stargan after the addition of COVID-19 lesions chosen considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in blue.

6.3.3.4 Evaluation - SalDACov

The SalDACov version of the proposed data augmentation method was also evaluated using the unified training approach detailed in Subsection 6.2.2, maintaining identical configurations to the evaluation of the DACov version of the proposed augmentation. This evaluation extends the

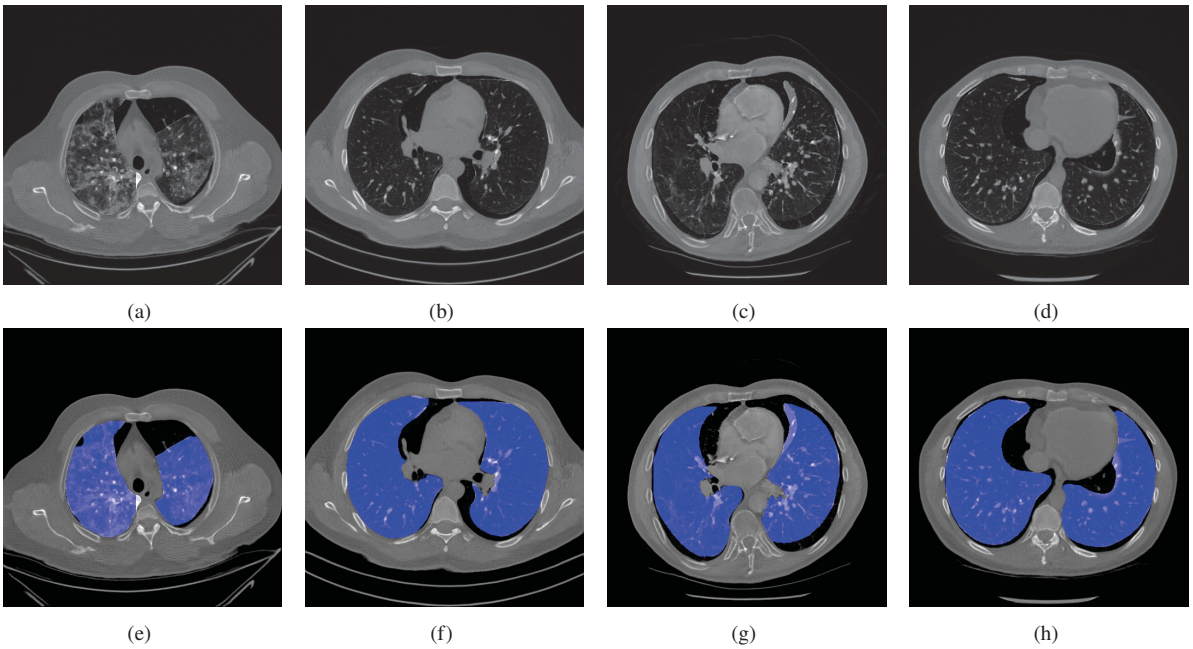


Figure 6.23: Examples of images generated by the Stylegan after the addition of COVID-19 lesions chosen considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in blue.

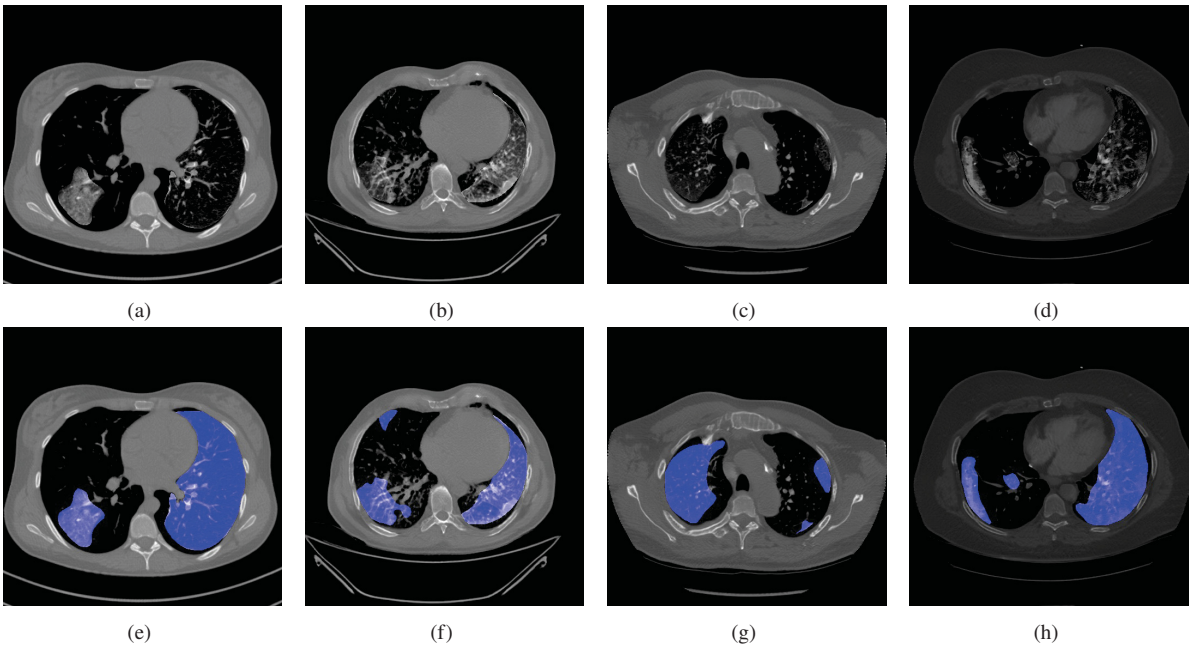


Figure 6.24: Examples of images generated by the Stargan after the addition of COVID-19 lesions chosen considering a random salience distance between the image generated by the GAN and the lesion image from the dataset. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in blue.

analysis presented in Subsection 6.2.2. Furthermore, the proposed data augmentation assessment was performed offline, augmenting the training set by 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, and 50%. The subsequent tables continue the findings depicted in Table 6.16, explicitly focusing on the evaluation results for the enhanced version of the proposed data augmentation technique within the unified training set. The F-scores highlighted in blue, and

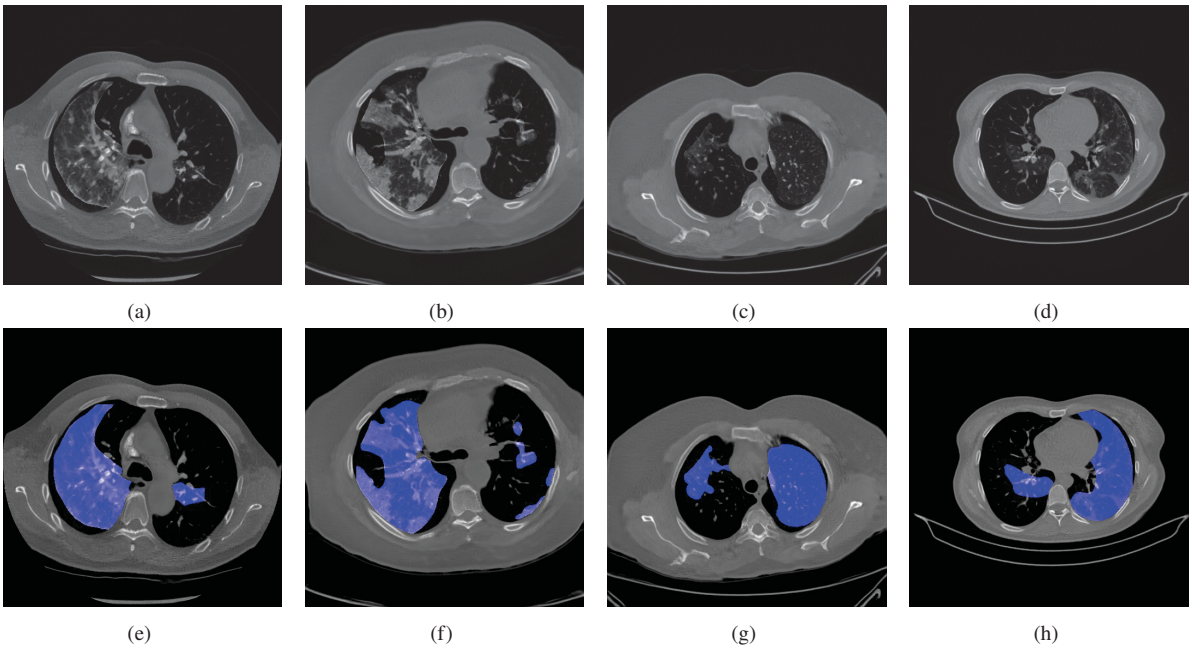


Figure 6.25: Examples of images generated by the Stylegan after the addition of COVID-19 lesions chosen considering a random salience distance between the image generated by the GAN and the lesion image from the dataset. Sub-figures (a) to (d) presents examples of images generated by the GAN with COVID-19 lesions and Sub-figures (e) to (h) presents the respective segmentation masks, with the lesion regions presented in blue.

the IoUs highlighted in red indicate the metrics where the SalDACov version of the proposed augmentation achieved higher results compared to both the generic data augmentation techniques presented in Subsection 6.2.2 and the DACov version of the proposed augmentation presented in Subsection 6.3.3.2.

The Wilcoxon Signed-Rank Test was utilized to perform statistical analysis between the segmentation model trained by applying the data augmentation technique and training the segmentation model without applying data augmentation. The green-highlighted values indicate data augmentation techniques resulting in P-values below 0.05, thereby rejecting the null hypothesis, signifying a statistical disparity and superior outcomes to those without data augmentation. The Wilcoxon Signed-Rank Test was also utilized to perform statistical analysis between training the segmentation model with the SalDACov version of the proposed augmentation and the DACov version of the proposed augmentation. The underscored values show when training with the SalDACov version of the proposed augmentation achieved a P-value lower than 0.05 when compared with training with the DACov version of the proposed augmentation, and the null hypothesis was rejected.

In the first evaluation, the image generation considered the maximum salience distance between the image produced by the GAN and the lesion image from the dataset. The lesions were positioned on the same lung side as the original image from the dataset. As highlighted in Table 6.17, compared to training the segmentation model without any augmentation, this configuration of the proposed augmentation method showcased statistical significance across all five evaluated datasets when assessed with augmentation probabilities of 35% and 40%. With probabilities ranging from 15% to 30%, the proposed augmentation achieved statistical significance in three datasets: MosMed, Ricord1a, and Zenodo. Furthermore, the proposed augmentation also achieved statistical difference in the MedSeg dataset when evaluated with probabilities 45% and 50%.

Table 6.17: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the maximum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in the same lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show when training with the salience version of the proposed augmentation achieved a P-value lower than 0.05 when compared with training with the random version of the proposed augmentation, and the null hypothesis was rejected.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8642	0.8090	0.8862	0.8233	0.8200	0.7564	0.8536	0.7878	0.9091	0.8507
	Stylegan	0.8609	0.8062	0.8868	0.8236	0.8196	0.7565	0.8503	0.7837	0.9087	0.8503
0.1	Stargan	0.8662	0.8109	0.8874	0.8247	0.8211	0.7574	0.8610	0.7962	0.9100	0.8519
	Stylegan	0.8653	0.8099	0.8881	0.8248	0.8202	0.7564	0.8586	0.7930	0.9087	0.8502
0.15	Stargan	0.8621	0.8069	0.8878	0.8247	0.8257	0.7630	0.8736	0.8107	0.9113	0.8537
	Stylegan	0.8668	0.8112	0.8881	0.8253	0.8237	0.7599	0.8655	0.8014	0.9102	0.8523
0.2	Stargan	0.8621	0.8073	0.8880	0.8250	0.8221	0.7590	0.8677	0.8039	0.9105	0.8525
	Stylegan	0.8643	0.8091	0.8898	0.8270	0.8234	0.7604	0.8665	0.8028	0.9109	0.8530
0.25	Stargan	0.8628	0.8066	0.8875	0.8247	0.8253	0.7624	0.8736	0.8107	0.9110	0.8533
	Stylegan	0.8647	0.8100	0.8894	0.8264	0.8205	0.7578	0.8683	0.8049	0.9112	0.8533
0.3	Stargan	0.8654	0.8091	0.8894	0.8265	0.8248	0.7619	0.8717	0.8085	0.9105	0.8529
	Stylegan	0.8609	0.8048	0.8904	0.8276	0.8259	0.7617	0.8668	0.8031	0.9101	0.8522
0.35	Stargan	0.8690	0.8130	0.8895	0.8270	0.8275	0.7642	0.8745	0.8119	0.9111	0.8532
	Stylegan	0.8644	0.8090	0.8907	0.8283	0.8239	0.7606	0.8681	0.8047	0.9112	0.8535
0.4	Stargan	0.8676	0.8122	0.8894	0.8266	0.8250	0.7612	0.8732	0.8104	0.9109	0.8532
	Stylegan	0.8622	0.8055	0.8907	0.8281	0.8250	0.7619	0.8689	0.8053	0.9112	0.8534
0.45	Stargan	0.8643	0.8088	0.8896	0.8271	0.8273	0.7640	0.8755	0.8132	0.9118	0.8540
	Stylegan	0.8641	0.8075	0.8915	0.8291	0.8257	0.7623	0.8701	0.8069	0.9108	0.8529
0.5	Stargan	0.8641	0.8097	0.8904	0.8283	0.8275	0.7638	0.8734	0.8108	0.9114	0.8539
	Stylegan	0.8646	0.8086	0.8904	0.8278	0.8268	0.7633	0.8697	0.8066	0.9108	0.8528

In the second evaluation of the proposed augmentation, presented in Table 6.18, the images were generated with the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. The lesions were positioned on the same lung side as the original image from the dataset. Compared to training the segmentation model without any augmentation, this variation of the proposed augmentation method yielded statistical significance across all five evaluated datasets when assessed with a probability of 5% and achieved statistical significance in four out of the five evaluated datasets when evaluated with probabilities ranging from 10% to 50%. Moreover, this configuration demonstrated promising results in the Ricord1a dataset, showing a notable improvement of 3% when evaluated with probabilities of 35%, 45%, and 50%, and with probabilities 5%, 15%, 35%, and 50%, achieved the higher values on Ricord1a dataset when compared with the generic data augmentation techniques presented in Subsection 6.2.2 and the DACov version of the proposed augmentation presented in Subsection 6.3.3.2.

In the third evaluation of the proposed augmentation, presented in Table 6.19, the images were generated with a random salience distance between the image produced by the GAN and the lesion image from the dataset, and the lesions were positioned on the same lung side as the original image from the dataset. Compared to training the segmentation model without

Table 6.18: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the minimum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in the same lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The F-scores highlighted in blue, and the IoUs highlighted in red indicate the metrics where the salience version of the proposed augmentation achieved higher values compared to both the generic data augmentation techniques presented in Sub-section 6.2.2 and the random version of the proposed augmentation presented in Sub-section 6.3.3.2. The underscored values show when training with the salience version of the proposed augmentation achieved a P-value lower than 0.05 when compared with training with the random version of the proposed augmentation, and the null hypothesis was rejected.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8680	0.8125	0.8900	0.8280	0.8278	0.7638	**0.8748**	**0.8121**	0.9116	0.8542
	Stylegan	0.8664	0.8096	0.8886	0.8260	0.8213	0.7582	0.8691	0.8054	0.9100	0.8521
0.1	Stargan	0.8640	0.8080	0.8902	0.8277	0.8228	0.7593	0.8704	0.8071	0.9105	0.8529
	Stylegan	0.8665	0.8106	0.8914	0.8294	0.8253	0.7618	0.8745	0.8123	0.9117	0.8544
0.15	Stargan	0.8643	0.8091	0.8930	0.8308	0.8265	0.7631	0.8748	0.8124	0.9117	0.8541
	Stylegan	0.8652	0.8099	0.8911	0.8290	0.8302	0.7668	**0.8781**	**0.8162**	0.9120	0.8547
0.2	Stargan	0.8652	0.8098	0.8914	0.8289	0.8255	0.7622	0.8742	0.8116	0.9120	0.8545
	Stylegan	0.8664	0.8108	0.8901	0.8281	0.8245	0.7604	0.8750	0.8128	0.9121	0.8547
0.25	Stargan	0.8650	0.8093	0.8917	0.8296	0.8291	0.7655	0.8790	0.8171	0.9127	0.8556
	Stylegan	0.8630	0.8082	0.8902	0.8281	0.8239	0.7614	0.8774	0.8159	0.9124	0.8550
0.3	Stargan	0.8633	0.8079	0.8908	0.8293	0.8259	0.7622	0.8739	0.8116	0.9118	0.8543
	Stylegan	0.8629	0.8078	0.8908	0.8283	0.8250	0.7613	0.8779	0.8162	0.9135	0.8563
0.35	Stargan	0.8652	0.8096	0.8900	0.8280	0.8278	0.7646	0.8779	0.8162	0.9132	0.8563
	Stylegan	0.8638	0.8084	0.8930	0.8310	0.8307	0.7674	**0.8811**	**0.8201**	0.9131	0.8561
0.4	Stargan	0.8621	0.8070	0.8909	0.8288	0.8305	0.7672	0.8780	0.8163	0.9123	0.8549
	Stylegan	0.8611	0.8063	0.8910	0.8290	0.8253	0.7621	0.8761	0.8139	0.9118	0.8542
0.45	Stargan	0.8636	0.8086	0.8910	0.8287	0.8273	0.7636	0.8774	0.8155	0.9126	0.8553
	Stylegan	0.8638	0.8082	0.8918	0.8301	0.8265	0.7633	0.8804	0.8193	0.9137	0.8568
0.5	Stargan	0.8638	0.8074	0.8916	0.8291	0.8304	0.7675	0.8783	0.8167	0.9133	0.8562
	Stylegan	0.8626	0.8076	0.8945	0.8329	0.8306	0.7675	**0.8805**	**0.8195**	0.9130	0.8562

augmentation, this variant of the proposed augmentation method yielded statistical significance in four of the five evaluated datasets when assessed with probabilities ranging from 20% to 50%.

Among these initial three evaluations of the proposed augmentation method, with the lesions positioned on the same lung side of the original image from the dataset, the most promising outcomes emerged from the approach where images were generated considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. Notably, the results obtained on the Ricord1a dataset surpassed those achieved in this dataset with the DACov version of the proposed augmentation, as presented in Subsection 6.3.3.2, as well as the results obtained with generic data augmentation techniques detailed in Subsection 6.2.2. These findings underscore the effectiveness and superiority of the SalDACov version of the proposed augmentation method, particularly in enhancing performance across challenging datasets like Ricord1a. The ability to surpass results obtained with the DACov version of the proposed augmentation and generic augmentation techniques highlights this novel augmentation approach's significant impact and potential in the COVID-19 CT segmentation problem.

Table 6.19: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the a random saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in the same lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show when training with the salience version of the proposed augmentation achieved a P-value lower than 0.05 when compared with training with the random version of the proposed augmentation, and the null hypothesis was rejected.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8657	0.8106	0.8891	0.8269	0.8224	0.7582	0.8624	0.7980	0.9098	0.8518
	Stylegan	0.8648	0.8088	0.8885	0.8259	0.8231	0.7596	0.8676	0.8038	0.9104	0.8524
0.1	Stargan	0.8629	0.8077	0.8894	0.8272	0.8194	0.7566	0.8666	0.8027	0.9103	0.8521
	Stylegan	0.8636	0.8080	0.8900	0.8280	0.8255	0.7622	0.8716	0.8086	0.9116	0.8537
0.15	Stargan	0.8633	0.8078	0.8889	0.8267	0.8245	0.7609	0.8706	0.8073	0.9104	0.8525
	Stylegan	0.8643	0.8091	0.8882	0.8254	0.8190	0.7556	0.8600	0.7949	0.9100	0.8518
0.2	Stargan	0.8645	0.8086	0.8883	0.8253	0.8235	0.7597	0.8665	0.8025	0.9095	0.8514
	Stylegan	0.8676	0.8110	0.8907	0.8282	0.8199	0.7579	0.8714	0.8083	0.9105	0.8527
0.25	Stargan	0.8639	0.8077	0.8891	0.8265	0.8268	0.7626	0.8761	0.8139	0.9116	0.8543
	Stylegan	0.8633	0.8074	0.8902	0.8282	0.8318	0.7675	0.8747	0.8126	0.9109	0.8536
0.3	Stargan	0.8652	0.8092	0.8903	0.8283	0.8272	0.7638	0.8761	0.8142	0.9124	0.8548
	Stylegan	0.8662	0.8096	0.8892	0.8269	0.8251	0.7614	0.8723	0.8094	0.9108	0.8530
0.35	Stargan	0.8631	0.8072	0.8919	0.8298	0.8303	0.7667	0.8765	0.8144	0.9119	0.8545
	Stylegan	0.8654	0.8101	0.8900	0.8278	0.8190	0.7560	0.8736	0.8114	0.9116	0.8538
0.4	Stargan	0.8635	0.8083	0.8903	0.8281	0.8294	0.7660	0.8770	0.8153	0.9119	0.8549
	Stylegan	0.8641	0.8079	0.8920	0.8300	0.8250	0.7615	0.8755	0.8135	0.9113	0.8539
0.45	Stargan	0.8648	0.8091	0.8892	0.8271	0.8234	0.7607	0.8762	0.8142	0.9128	0.8555
	Stylegan	0.8656	0.8099	0.8924	0.8302	0.8284	0.7636	0.8761	0.8140	0.9118	0.8544
0.5	Stargan	0.8654	0.8099	0.8909	0.8286	0.8302	0.7660	0.8789	0.8173	0.9122	0.8552
	Stylegan	0.8660	0.8097	0.8928	0.8309	0.8258	0.7625	0.8765	0.8145	0.9108	0.8535

In the evaluation presented in Table 6.20, image generation considered the maximum salience distance between the image produced by the GAN and the lesion image from the dataset, with lesions positioned on the opposite lung side compared to the original image from the dataset. However, this configuration yielded the least favorable results, achieving statistical significance only in the MosMed, Ricord1a, and Zenodo datasets compared to training the segmentation model without augmentation. Specifically, in the MedSeg dataset, statistical significance was attained solely with probabilities of 30% and 45%. This configuration failed to achieve statistical significance in the CC-CCII dataset across all evaluated probabilities.

In the evaluation presented in Table 6.21, image generation considered the minimum salience distance between the image produced by the GAN and the lesion image from the dataset, with lesions positioned on the opposite lung side from the original image in the dataset. Remarkably, when compared to training the segmentation model without augmentation, this variant of the proposed augmentation method exhibited statistical significance across all five evaluated datasets when assessed with a probability of 50%. Additionally, it achieved statistical significance in four out of the five evaluated datasets when evaluated with probabilities ranging from 5% to 45%. This evaluation yielded promising results in the Ricord1a dataset, showcasing a notable improvement of 3% when assessed with probabilities of 35%, 40%, and

Table 6.20: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the maximum saliency distance between the image generated by the GAN. The lesions are placed in the opposite lung side from the original image from the dataset. p stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show when training with the salience version of the proposed augmentation achieved a P-value lower than 0.05 when compared with training with the random version of the proposed augmentation, and the null hypothesis was rejected.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8600	0.8045	0.8872	0.8245	0.8203	0.7563	0.8567	0.7910	0.9088	0.8505
	Stylegan	0.8633	0.8084	0.8864	0.8234	0.8215	0.7576	0.8558	0.7899	0.9094	0.8510
0.1	Stargan	0.8675	0.8125	0.8882	0.8254	0.8177	0.7539	0.8618	0.7970	0.9111	0.8533
	Stylegan	0.8635	0.8080	0.8878	0.8253	0.8206	0.7577	0.8600	0.7950	0.9091	0.8508
0.15	Stargan	0.8627	0.8072	0.8871	0.8243	0.8223	0.7588	0.8612	0.7965	0.9094	0.8509
	Stylegan	0.8660	0.8100	0.8873	0.8248	0.8217	0.7589	0.8607	0.7957	0.9099	0.8515
0.2	Stargan	0.8632	0.8077	0.8879	0.8252	0.8227	0.7590	0.8600	0.7949	0.9100	0.8521
	Stylegan	0.8640	0.8081	0.8879	0.8259	0.8229	0.7591	0.8637	0.7990	0.9100	0.8518
0.25	Stargan	0.8656	0.8098	0.8887	0.8257	0.8244	0.7616	0.8722	0.8093	0.9114	0.8541
	Stylegan	0.8651	0.8092	0.8880	0.8255	0.8225	0.7587	0.8640	0.7996	0.9098	0.8519
0.3	Stargan	0.8649	0.8094	0.8908	0.8276	0.8250	0.7612	0.8720	0.8090	0.9112	0.8535
	Stylegan	0.8645	0.8089	0.8912	0.8285	0.8238	0.7608	0.8679	0.8042	0.9109	0.8532
0.35	Stargan	0.8635	0.8086	0.8887	0.8255	0.8239	0.7606	0.8714	0.8082	0.9097	0.8518
	Stylegan	0.8642	0.8088	0.8894	0.8267	0.8228	0.7594	0.8716	0.8087	0.9104	0.8523
0.4	Stargan	0.8655	0.8096	0.8882	0.8258	0.8274	0.7634	0.8727	0.8100	0.9115	0.8535
	Stylegan	0.8649	0.8098	0.8914	0.8285	0.8257	0.7629	0.8708	0.8076	0.9114	0.8536
0.45	Stargan	0.8662	0.8105	0.8916	0.8291	0.8242	0.7615	0.8740	0.8114	0.9121	0.8546
	Stylegan	0.8666	0.8101	0.8888	0.8262	0.8247	0.7613	0.8715	0.8082	0.9113	0.8535
0.5	Stargan	0.8643	0.8085	0.8885	0.8258	0.8284	0.7654	0.8736	0.8109	0.9120	0.8544
	Stylegan	0.8638	0.8072	0.8883	0.8255	0.8248	0.7619	0.8699	0.8069	0.9112	0.8534

45%. Furthermore, with probabilities 10%, 40%, and 45%, the salience based on the proposed augmentation achieved higher values on the Ricord1a dataset when compared with the generic data augmentation techniques and the DACov version of the proposed augmentation presented. With a probability of 10%, the salience augmentation achieved the highest F-score on the Zenodo dataset.

In the subsequent evaluation, presented in Table 6.22, image generation considered a random salience distance between the image produced by the GAN and the lesion image from the dataset, with lesions positioned on the opposite lung side compared to the original image in the dataset. Compared to training the segmentation model without any augmentation, this variant of the proposed augmentation method demonstrated statistical significance in the MosMed, Ricord1a, and Zenodo datasets when applied with probabilities ranging from 5% to 15%. When applied with a probability of 20%, the augmentation achieved statistical significance in the CC-CCII, MosMed, Ricord1a, and Zenodo datasets. Moreover, the augmentation method achieved statistical significance in the MedSeg, MosMed, Ricord1a, and Zenodo datasets with higher evaluated probabilities.

Similar to the initial three evaluations, in these three evaluations of the proposed augmentation method, with the lesions positioned on the opposite lung side from the original

Table 6.21: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the minimum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in the opposite lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The F-scores highlighted in blue, and the IoUs highlighted in red indicate the metrics where the salience version of the proposed augmentation achieved higher values compared to both the generic data augmentation techniques presented in Sub-section 6.2.2 and the random version of the proposed augmentation presented in Sub-section 6.3.3.2. The underscored values show when training with the salience version of the proposed augmentation achieved a P-value lower than 0.05 when compared with training with the random version of the proposed augmentation, and the null hypothesis was rejected.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8659	0.8106	0.8894	0.8266	0.8214	0.7576	0.8630	0.7983	0.9105	0.8525
	Stylegan	0.8643	0.8089	0.8885	0.8258	0.8230	0.7597	0.8706	0.8073	0.9101	0.8519
0.1	Stargan	0.8669	0.8115	0.8894	0.8264	0.8226	0.7598	0.8729	0.8099	0.9107	0.8529
	Stylegan	0.8669	0.8112	0.8915	0.8296	0.8276	0.7647	**0.8773**	**0.8152**	**0.9125**	**0.8552**
0.15	Stargan	0.8638	0.8075	0.8904	0.8285	0.8294	0.7660	0.8752	0.8126	0.9113	0.8539
	Stylegan	0.8638	0.8073	0.8898	0.8278	0.8258	0.7620	0.8773	0.8152	0.9123	0.8550
0.2	Stargan	0.8669	0.8118	0.8917	0.8296	0.8239	0.7601	0.8751	0.8127	0.9120	0.8543
	Stylegan	0.8670	0.8118	0.8927	0.8312	0.8287	0.7652	0.8784	0.8166	0.9133	0.8560
0.25	Stargan	0.8658	0.8102	0.8914	0.8300	0.8251	0.7613	0.8713	0.8081	0.9124	0.8550
	Stylegan	0.8612	0.8057	0.8896	0.8270	0.8263	0.7634	0.8784	0.8169	0.9118	0.8544
0.3	Stargan	0.8656	0.8098	0.8927	0.8308	0.8293	0.7649	0.8790	0.8176	0.9133	0.8563
	Stylegan	0.8657	0.8095	0.8892	0.8273	0.8288	0.7650	0.8760	0.8140	0.9115	0.8545
0.35	Stargan	0.8665	0.8107	0.8922	0.8306	0.8312	0.7680	0.8801	0.8187	0.9139	0.8573
	Stylegan	0.8665	0.8103	0.8926	0.8311	0.8296	0.7660	0.8795	0.8181	0.9129	0.8556
0.4	Stargan	0.8652	0.8096	0.8927	0.8306	0.8282	0.7646	0.8785	0.8170	0.9127	0.8553
	Stylegan	0.8632	0.8077	0.8925	0.8306	0.8361	0.7725	**0.8814**	**0.8205**	0.9142	0.8575
0.45	Stargan	0.8668	0.8109	0.8927	0.8303	0.8299	0.7672	**0.8830**	**0.8222**	0.9138	0.8573
	Stylegan	0.8591	0.8049	0.8894	0.8269	0.8254	0.7619	0.8776	0.8156	0.9129	0.8557
0.5	Stargan	0.8654	0.8104	0.8918	0.8297	0.8300	0.7665	0.8794	0.8182	0.9138	0.8569
	Stylegan	0.8662	0.8106	0.8922	0.8302	0.8267	0.7644	0.8776	0.8160	0.9119	0.8545

image from the dataset, the most promising outcomes were observed when images were generated considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. Furthermore, the results obtained on the Ricord1a dataset also surpassed those achieved in this dataset with the DACov version of the proposed augmentation, as presented in Subsection 6.3.3.2, as well as the results obtained with generic data augmentation techniques detailed in Subsection 6.2.2. However, this version achieved less promising results than maintaining the lesions positioned on the same lung side from the original image from the dataset.

In the evaluation, presented in Table 6.23, image generation involved considering the maximum salience distance between the image generated by the GAN and the lesion image from the dataset, with lesions randomly positioned on either lung side compared to the original image in the dataset. Remarkably, compared to training the segmentation model without any augmentation, this variant of the proposed augmentation method achieved statistical significance in the MedSeg, MosMed, Ricord1a, and Zenodo datasets when applied with a probability higher than 25%

Table 6.22: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the a random saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in the opposite lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The underscored values show when training with the salience version of the proposed augmentation achieved a P-value lower than 0.05 when compared with training with the random version of the proposed augmentation, and the null hypothesis was rejected.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8620	0.8072	0.8896	0.8271	0.8257	0.7633	0.8678	0.8044	0.9104	0.8527
	Stylegan	0.8644	0.8090	0.8877	0.8252	0.8211	0.7575	0.8554	0.7893	0.9081	0.8497
0.1	Stargan	0.8647	0.8097	0.8881	0.8253	0.8258	0.7621	0.8648	0.8005	0.9108	0.8530
	Stylegan	0.8662	0.8107	0.8888	0.8261	0.8220	0.7583	0.8644	0.8002	0.9092	0.8509
0.15	Stargan	0.8628	0.8071	0.8889	0.8266	0.8282	0.7641	0.8695	0.8060	0.9111	0.8532
	Stylegan	0.8644	0.8092	0.8893	0.8271	0.8240	0.7611	0.8658	0.8016	0.9095	0.8509
0.2	Stargan	0.8657	0.8102	0.8901	0.8274	0.8258	0.7633	0.8716	0.8085	0.9117	0.8541
	Stylegan	0.8701	0.8149	0.8891	0.8262	0.8236	0.7601	0.8723	0.8093	0.9101	0.8524
0.25	Stargan	0.8606	0.8050	0.8895	0.8267	0.8252	0.7620	0.8738	0.8112	0.9113	0.8536
	Stylegan	0.8651	0.8084	0.8904	0.8277	0.8224	0.7579	0.8690	0.8058	0.9102	0.8520
0.3	Stargan	0.8650	0.8102	0.8901	0.8281	0.8280	0.7645	0.8718	0.8088	0.9112	0.8536
	Stylegan	0.8628	0.8068	0.8910	0.8284	0.8264	0.7635	0.8737	0.8113	0.9111	0.8531
0.35	Stargan	0.8647	0.8088	0.8909	0.8287	0.8294	0.7663	0.8774	0.8154	0.9129	0.8558
	Stylegan	0.8621	0.8066	0.8906	0.8281	0.8220	0.7582	0.8747	0.8122	0.9107	0.8529
0.4	Stargan	0.8656	0.8093	0.8917	0.8293	0.8262	0.7625	0.8761	0.8139	0.9128	0.8555
	Stylegan	0.8657	0.8111	0.8906	0.8285	0.8235	0.7596	0.8734	0.8109	0.9118	0.8542
0.45	Stargan	0.8632	0.8078	0.8915	0.8293	0.8296	0.7661	0.8787	0.8172	0.9137	0.8567
	Stylegan	0.8655	0.8092	0.8934	0.8315	0.8212	0.7584	0.8747	0.8124	0.9117	0.8541
0.5	Stargan	0.8641	0.8085	0.8900	0.8274	0.8295	0.7659	0.8777	0.8157	0.9122	0.8549
	Stylegan	0.8653	0.8092	0.8918	0.8297	0.8212	0.7576	0.8750	0.8131	0.9123	0.8549

Table 6.24 displays the evaluation results of the proposed augmentation method, with images generated considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset. The lesions were randomly placed on either lung side compared to the original image in the dataset. Compared to training the segmentation model without any augmentation, this variant of the proposed augmentation achieved statistical significance in the MedSeg, MosMed, Ricord1a, and Zenodo datasets when applied with a probability of 5% and over 15%. Additionally, statistical significance was attained in the CC-CCII dataset when the augmentation was applied with a probability of 30%. Furthermore, this evaluation yielded an improvement of 3% on the Ricord1a dataset when evaluated with a probability of 30%. Furthermore, with probabilities 20%, 25%, and 30%, the SalDACov version of the proposed augmentation achieved higher values on the Ricord1a dataset compared to the generic data augmentation techniques and the DACov version of the proposed augmentation presented.

In the following evaluation, presented in Table 6.25, image generation involved considering a random salience distance between the image generated by the GAN and the lesion image from the dataset, with lesions randomly positioned on either lung side compared to the original image in the dataset. Compared to training the segmentation model without augmentation, this

Table 6.23: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the maximum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in a random lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation).

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8629	0.8070	0.8864	0.8235	0.8216	0.7575	0.8594	0.7942	0.9098	0.8516
	Stylegan	0.8645	0.8090	0.8873	0.8241	0.8173	0.7536	0.8603	0.7956	0.9096	0.8514
0.1	Stargan	0.8648	0.8089	0.8863	0.8235	0.8177	0.7546	0.8616	0.7969	0.9107	0.8526
	Stylegan	0.8624	0.8073	0.8870	0.8242	0.8233	0.7595	0.8626	0.7982	0.9090	0.8506
0.15	Stargan	0.8654	0.8104	0.8882	0.8253	0.8233	0.7604	0.8669	0.8030	0.9100	0.8521
	Stylegan	0.8642	0.8080	0.8875	0.8244	0.8192	0.7558	0.8619	0.7973	0.9097	0.8512
0.2	Stargan	0.8638	0.8087	0.8885	0.8257	0.8235	0.7603	0.8686	0.8046	0.9099	0.8517
	Stylegan	0.8655	0.8087	0.8893	0.8263	0.8211	0.7579	0.8625	0.7982	0.9093	0.8511
0.25	Stargan	0.8651	0.8097	0.8874	0.8245	0.8278	0.7638	0.8726	0.8098	0.9111	0.8533
	Stylegan	0.8656	0.8093	0.8903	0.8278	0.8278	0.7626	0.8698	0.8067	0.9114	0.8536
0.3	Stargan	0.8639	0.8088	0.8907	0.8285	0.8275	0.7642	0.8732	0.8107	0.9113	0.8535
	Stylegan	0.8650	0.8094	0.8897	0.8265	0.8236	0.7599	0.8712	0.8083	0.9115	0.8537
0.35	Stargan	0.8647	0.8095	0.8905	0.8280	0.8249	0.7613	0.8701	0.8069	0.9105	0.8526
	Stylegan	0.8628	0.8077	0.8898	0.8269	0.8213	0.7582	0.8704	0.8073	0.9116	0.8539
0.4	Stargan	0.8641	0.8084	0.8900	0.8280	0.8243	0.7609	0.8717	0.8088	0.9108	0.8532
	Stylegan	0.8625	0.8068	0.8890	0.8261	0.8235	0.7606	0.8685	0.8049	0.9110	0.8532
0.45	Stargan	0.8640	0.8089	0.8913	0.8288	0.8256	0.7623	0.8712	0.8082	0.9114	0.8536
	Stylegan	0.8643	0.8080	0.8912	0.8287	0.8280	0.7649	0.8716	0.8085	0.9115	0.8536
0.5	Stargan	0.8624	0.8069	0.8911	0.8290	0.8236	0.7606	0.8715	0.8085	0.9116	0.8542
	Stylegan	0.8647	0.8090	0.8920	0.8293	0.8239	0.7600	0.8680	0.8043	0.9098	0.8520

variant of the proposed augmentation method achieved statistical significance in the MedSeg, MosMed, Ricord1a, and Zenodo datasets when applied with a probability of 20% and probabilities exceeding 30%. Additionally, when applied with a probability of 25%, the augmentation method achieved statistical significance in the CC-CCII, MosMed, Ricord1a, and Zenodo datasets.

In these three last evaluations of the proposed augmentation method, with the lesions positioned on a random lung side from the original image from the dataset, the most promising outcomes were also observed when images were generated considering the minimum salience distance between the image generated by the GAN and the lesion image from the dataset, with the results obtained on the Ricord1a dataset surpassing those achieved in this dataset with the DACov version of the proposed augmentation and the results obtained with generic data augmentation techniques.

The proposed augmentation showed promising results based on the nine experiments conducted, particularly on the Ricord1a dataset. Like the generic augmentation, a 50% probability yielded the best outcomes. Furthermore, similar to the DACov version of the proposed augmentation, there was a minimal disparity in the results from images generated using Stargan and Stylegan. It is also worth noting that augmentation using images generated based on the minimum salience distance between the GAN-generated image and the lesion image from the dataset produced the best results compared to using images based on maximum and random

Table 6.24: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering the minimum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in a random lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The F-scores highlighted in blue, and the IoUs highlighted in red indicate the metrics where the salience version of the proposed augmentation achieved higher values compared to both the generic data augmentation techniques presented in Sub-section 6.2.2 and the random version of the proposed augmentation presented in Sub-section 6.3.3.2.

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8668	0.8112	0.8915	0.8294	0.8250	0.7617	0.8747	0.8122	0.9116	0.8543
	Stylegan	0.8590	0.8032	0.8884	0.8263	0.8199	0.7567	0.8692	0.8057	0.9114	0.8539
0.1	Stargan	0.8673	0.8116	0.8896	0.8273	0.8233	0.7601	0.8721	0.8092	0.9119	0.8542
	Stylegan	0.8681	0.8127	0.8900	0.8278	0.8260	0.7625	0.8751	0.8125	0.9117	0.8544
0.15	Stargan	0.8644	0.8089	0.8893	0.8272	0.8286	0.7652	0.8774	0.8155	0.9116	0.8542
	Stylegan	0.8636	0.8083	0.8905	0.8278	0.8228	0.7592	0.8779	0.8162	0.9115	0.8538
0.2	Stargan	0.8670	0.8115	0.8914	0.8297	0.8298	0.7654	0.8787	0.8169	0.9125	0.8553
	Stylegan	0.8645	0.8091	0.8912	0.8295	0.8317	0.7677	**0.8794**	**0.8177**	0.9128	0.8555
0.25	Stargan	0.8649	0.8097	0.8906	0.8282	0.8273	0.7637	0.8759	0.8135	0.9117	0.8543
	Stylegan	0.8637	0.8078	0.8918	0.8298	0.8323	0.7686	**0.8793**	**0.8177**	0.9132	0.8563
0.3	Stargan	0.8670	0.8110	0.8914	0.8295	0.8278	0.7649	**0.8810**	**0.8195**	0.9126	0.8556
	Stylegan	0.8646	0.8094	0.8899	0.8279	0.8241	0.7603	0.8770	0.8152	0.9123	0.8549
0.35	Stargan	0.8656	0.8099	0.8888	0.8263	0.8248	0.7613	0.8761	0.8142	0.9119	0.8542
	Stylegan	0.8649	0.8096	0.8913	0.8294	0.8263	0.7618	0.8762	0.8144	0.9116	0.8542
0.4	Stargan	0.8641	0.8088	0.8928	0.8309	0.8301	0.7668	0.8776	0.8159	0.9132	0.8561
	Stylegan	0.8657	0.8092	0.8918	0.8300	0.8251	0.7612	0.8774	0.8158	0.9121	0.8549
0.45	Stargan	0.8629	0.8078	0.8928	0.8308	0.8271	0.7635	0.8785	0.8170	0.9130	0.8555
	Stylegan	0.8655	0.8094	0.8925	0.8307	0.8327	0.7689	0.8789	0.8173	0.9129	0.8557
0.5	Stargan	0.8640	0.8080	0.8890	0.8266	0.8238	0.7606	0.8763	0.8144	0.9120	0.8543
	Stylegan	0.8626	0.8066	0.8911	0.8288	0.8278	0.7648	0.8770	0.8152	0.9128	0.8556

distances. However, changing the lesion's position to the opposite lung or a random lung side from the original image did not differ from keeping it on the same lung side.

6.3.3.5 Evaluation - SalDACov Version Plus Traditional Data Augmentation Techniques

The SalDACov version of the proposed data augmentation method was combined with traditional data augmentation techniques and evaluated using the unified training approach detailed in Subsection 6.2.2. This evaluation continues the experiments presented in Subsections 6.2.2 and 6.3.3.4. However, unlike the previous evaluation, the augmentation techniques were applied only with a probability of 50%, as this probability yielded the most promising results in the prior evaluation. Tables 6.26, 6.27, and 6.28 present the results of the proposed data augmentation applied in combination with the traditional data augmentation techniques.

The proposed augmentation used images generated with the minimum salience distance between the image produced by the GAN and the lesion image from the dataset due to the best outcomes presented in 6.3.3.4. The Wilcoxon Signed-Rank Test was utilized to perform statistical analysis between the segmentation model trained applying data augmentation technique and training the segmentation model without applying data augmentation. The green-highlighted

Table 6.25: Results of the salience version of the data augmentation evaluation when unifying the training sets. The images were generated considering a random saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in a random lung side from the original image from the dataset. *p* stands for probability. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation).

p	**Augmentation**	**CC-CCII**		**MedSeg**		**MosMed**		**Ricord1a**		**Zenodo**	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
0.05	Stargan	0.8635	0.8084	0.8884	0.8256	0.8224	0.7592	0.8622	0.7974	0.9102	0.8519
	Stylegan	0.8620	0.8060	0.8882	0.8255	0.8223	0.7578	0.8579	0.7922	0.9090	0.8507
0.1	Stargan	0.8666	0.8102	0.8906	0.8284	0.8205	0.7566	0.8655	0.8013	0.9093	0.8511
	Stylegan	0.8626	0.8067	0.8869	0.8241	0.8214	0.7572	0.8605	0.7955	0.9093	0.8512
0.15	Stargan	0.8662	0.8109	0.8881	0.8258	0.8184	0.7550	0.8626	0.7980	0.9107	0.8525
	Stylegan	0.8663	0.8102	0.8909	0.8282	0.8242	0.7606	0.8683	0.8051	0.9103	0.8520
0.2	Stargan	0.8628	0.8067	0.8904	0.8283	0.8308	0.7670	0.8724	0.8094	0.9123	0.8548
	Stylegan	0.8671	0.8121	0.8907	0.8281	0.8250	0.7612	0.8732	0.8105	0.9113	0.8535
0.25	Stargan	0.8692	0.8129	0.8892	0.8269	0.8281	0.7649	0.8760	0.8139	0.9114	0.8539
	Stylegan	0.8646	0.8082	0.8888	0.8260	0.8236	0.7615	0.8754	0.8131	0.9115	0.8541
0.3	Stargan	0.8655	0.8098	0.8909	0.8283	0.8269	0.7637	0.8754	0.8133	0.9119	0.8548
	Stylegan	0.8642	0.8086	0.8902	0.8278	0.8270	0.7637	0.8764	0.8146	0.9112	0.8535
0.35	Stargan	0.8636	0.8087	0.8916	0.8293	0.8303	0.7672	0.8792	0.8178	0.9136	0.8566
	Stylegan	0.8659	0.8094	0.8901	0.8274	0.8260	0.7633	0.8752	0.8129	0.9122	0.8548
0.4	Stargan	0.8644	0.8090	0.8911	0.8284	0.8292	0.7648	0.8764	0.8143	0.9125	0.8553
	Stylegan	0.8638	0.8088	0.8916	0.8296	0.8266	0.7634	0.8761	0.8142	0.9117	0.8543
0.45	Stargan	0.8640	0.8077	0.8914	0.8297	0.8292	0.7652	0.8787	0.8170	0.9132	0.8562
	Stylegan	0.8685	0.8131	0.8915	0.8288	0.8293	0.7660	0.8768	0.8150	0.9120	0.8547
0.5	Stargan	0.8674	0.8126	0.8902	0.8276	0.8288	0.7651	0.8767	0.8147	0.9121	0.8548
	Stylegan	0.8669	0.8117	0.8909	0.8287	0.8240	0.7610	0.8756	0.8136	0.9122	0.8549

values in the following tables indicate data augmentation techniques resulting in P-values below 0.05, thereby rejecting the null hypothesis, signifying a statistical disparity and superior outcomes to those without data augmentation.

The F-scores highlighted in blue indicate the best F-score value, and the IoUs highlighted in red indicate the best IoU values. The Wilcoxon Signed-Rank Test was also utilized to perform statistical analysis between training the segmentation model with the combination of the proposed augmentation and traditional augmentation techniques and training the segmentation model with just the traditional augmentation techniques. The underscored values presented in Tables 6.26, 6.27, and 6.28 are the traditional augmentation techniques that combined with the proposed augmentation resulted in a P-value lower than 0.05, thereby rejecting the null hypothesis, suggesting that there is a statistical difference and the achieved results with the combination of the proposed augmentation and the traditional augmentation technique achieved superior outcomes compared to use just the tradition augmentation technique.

As presented in Tables 6.26, 6.27, and 6.28, the SalDACov version of the proposed augmentation, combined with traditional data augmentation techniques, achieved statistical significance across all five datasets and obtained better F-scores in the MosMed, Ricord1a, and Zenodo datasets compared to applying traditional techniques alone. In Table 6.26, the proposed augmentation used images with the lesions positioned on the same lung side as in the original dataset image. For the images generated by Stargan, the proposed augmentation achieved the

highest F-scores when combined with Rotate on the MosMed dataset, with Gaussian Blur on the Ricord1a dataset, and Optical Distortion on the Zenodo dataset. For the images generated by Stylegan, the proposed augmentation achieved the highest F-scores when combined with Piecewise Affine on the MosMed dataset, Sharpen on the Ricord1a dataset, and Median Blur on the Zenodo dataset.

Table 6.26: Results of the proposed data augmentation applied in combination with the traditional data augmentation techniques in the unified training set. The images were generated considering the minimum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in the same lung side from the original image from the dataset. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The blue-colored values indicate the best F-score values, and the red-colored values indicate the best IoU values. The underscored values show the techniques in which the proposed augmentation combined with the traditional data augmentation techniques achieved a P-value lower than 0.05 when compared to just applying the generic data augmentation techniques presented on Subsection 6.2.2, and the null hypothesis was rejected.

GAN	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
Stargan	CLAHE	0.8556	0.8000	0.8917	0.8296	0.8323	0.7682	0.8815	0.8195	0.9121	0.8551
	Coarse Dropout	0.8675	0.8123	0.8957	0.8353	0.8410	0.7779	0.8816	0.8208	0.9148	0.8584
	Elastic Transform	0.8789	0.8274	**0.9001**	**0.8407**	0.8528	0.7897	0.8753	0.8130	0.9159	0.8597
	Emboss	0.8632	0.8091	0.8936	0.8328	0.8271	0.7625	0.8789	0.8171	0.9139	0.8570
	Flip	0.8709	0.8160	0.8965	0.8355	0.8483	0.7862	0.8683	0.8045	0.9114	0.8532
	Gaussian Blur	0.8642	0.8084	0.8943	0.8335	0.8273	0.7643	**0.8825**	**0.8212**	0.9155	0.8595
	Grid Distortion	0.8759	0.8241	0.8993	0.8395	0.8505	0.7867	0.8760	0.8137	0.9159	0.8596
	Grid Dropout	0.8674	0.8114	0.8928	0.8318	0.8371	0.7738	0.8785	0.8166	0.9122	0.8548
	Image Compression	0.8655	0.8105	0.8936	0.8323	0.8306	0.7669	0.8817	0.8206	0.9138	0.8573
	Median Blur	0.8654	0.8099	0.8930	0.8319	0.8225	0.7599	0.8797	0.8183	0.9155	0.8595
	Optical Distortion	0.8742	0.8194	0.8964	0.8363	0.8412	0.7790	0.8793	0.8177	**0.9168**	**0.8610**
	Piecewise Affine	0.8755	0.8232	0.8978	0.8379	0.8493	0.7867	0.8745	0.8119	0.9148	0.8581
	Posterize	0.8621	0.8065	0.8938	0.8319	0.8323	0.7687	0.8783	0.8164	0.9142	0.8575
	RBC	0.8586	0.8026	0.8931	0.8313	0.8326	0.7691	0.8773	0.8152	0.9123	0.8547
	Random Crop	0.8629	0.8073	0.8922	0.8300	0.8304	0.7673	0.8791	0.8176	0.9125	0.8553
	Random Gamma	0.8655	0.8096	0.8926	0.8308	0.8330	0.7694	0.8797	0.8179	0.9131	0.8561
	Random Snow	0.8666	0.8103	0.8950	0.8333	0.8416	0.7782	0.8785	0.8160	0.9140	0.8572
	Rotate	**0.8792**	**0.8255**	0.8989	0.8388	**0.8551**	**0.7912**	0.8666	0.8028	0.9116	0.8535
	Sharpen	0.8643	0.8087	0.8963	0.8349	0.8290	0.7647	0.8803	0.8189	0.9156	0.8595
	Shift Scale Rotate	0.8771	0.8251	0.8990	0.8394	0.8539	0.7902	0.8712	0.8080	0.9157	0.8590
Stylegan	CLAHE	0.8593	0.8034	0.8926	0.8305	0.8318	0.7689	0.8822	0.8203	0.9125	0.8552
	Coarse Dropout	0.8660	0.8110	0.8957	0.8351	0.8422	0.7790	0.8802	0.8190	0.9141	0.8578
	Elastic Transform	0.8772	0.8259	**0.9013**	**0.8415**	0.8518	0.7895	0.8752	0.8127	0.9154	0.8591
	Emboss	0.8671	0.8122	0.8941	0.8331	0.8264	0.7631	0.8787	0.8174	0.9140	0.8572
	Flip	0.8727	0.8186	0.8989	0.8379	0.8469	0.7847	0.8668	0.8030	0.9110	0.8524
	Gaussian Blur	0.8625	0.8070	0.8948	0.8339	0.8258	0.7626	0.8819	0.8207	0.9145	0.8581
	Grid Distortion	0.8763	0.8241	0.9001	0.8403	0.8564	0.7932	0.8765	0.8145	0.9158	0.8596
	Grid Dropout	0.8652	0.8097	0.8928	0.8314	0.8348	0.7717	0.8773	0.8156	0.9120	0.8548
	Image Compression	0.8661	0.8106	0.8937	0.8324	0.8312	0.7681	0.8803	0.8193	0.9142	0.8574
	Median Blur	0.8634	0.8084	0.8949	0.8339	0.8318	0.7696	0.8829	0.8216	**0.9161**	**0.8602**
	Optical Distortion	0.8736	0.8199	0.8994	0.8391	0.8333	0.7708	0.8809	0.8195	0.9160	0.8601
	Piecewise Affine	0.8769	0.8245	0.8999	0.8398	**0.8574**	**0.7947**	0.8777	0.8158	0.9156	0.8593
	Posterize	0.8671	0.8105	0.8958	0.8340	0.8300	0.7667	0.8797	0.8180	0.9143	0.8576
	RBC	0.8591	0.8024	0.8976	0.8360	0.8357	0.7724	0.8785	0.8162	0.9125	0.8554
	Random Crop	0.8624	0.8066	0.8899	0.8281	0.8276	0.7643	0.8782	0.8167	0.9126	0.8551
	Random Gamma	0.8651	0.8083	0.8909	0.8287	0.8277	0.7635	0.8791	0.8176	0.9124	0.8549
	Random Snow	0.8717	0.8150	0.8928	0.8310	0.8411	0.7771	0.8776	0.8149	0.9135	0.8565
	Rotate	**0.8808**	**0.8271**	0.8993	0.8394	0.8553	0.7925	0.8664	0.8026	0.9129	0.8551
	Sharpen	0.8646	0.8089	0.8949	0.8334	0.8354	0.7711	**0.8851**	0.8242	0.9151	0.8592
	Shift Scale Rotate	0.8790	0.8267	0.8989	0.8391	0.8569	0.7935	0.8726	0.8096	0.9144	0.8574

Table 6.27: Results of the proposed data augmentation applied in combination with the traditional data augmentation techniques in the unified training set. The images were generated considering the minimum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in the opposite lung side from the original image from the dataset. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The blue-colored values indicate the best F-score values, and the red-colored values indicate the best IoU values. The underscored values show the techniques in which the proposed augmentation combined with the traditional data augmentation techniques achieved a P-value lower than 0.05 when compared to just applying the generic data augmentation techniques presented on Subsection 6.2.2, and the null hypothesis was rejected.

GAN	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
Stargan	CLAHE	0.8540	0.7987	0.8912	0.8285	0.8302	0.7670	0.8790	0.8174	0.9120	0.8548
	Coarse Dropout	0.8654	0.8115	0.8935	0.8329	0.8356	0.7724	0.8791	0.8177	0.9139	0.8573
	Elastic Transform	0.8782	0.8268	0.8996	0.8400	0.8502	0.7878	0.8758	0.8138	0.9162	0.8599
	Emboss	0.8667	0.8126	0.8943	0.8332	0.8339	0.7697	0.8827	0.8220	0.9150	0.8586
	Flip	0.8723	0.8181	0.8944	0.8330	0.8488	0.7866	0.8675	0.8037	0.9107	0.8520
	Gaussian Blur	0.8628	0.8082	0.8957	0.8354	0.8255	0.7617	0.8819	0.8208	0.9150	0.8589
	Grid Distortion	0.8762	0.8239	**0.9004**	**0.8409**	0.8507	0.7878	0.8755	0.8132	**0.9163**	0.8600
	Grid Dropout	0.8634	0.8086	0.8943	0.8325	0.8389	0.7754	0.8764	0.8147	0.9122	0.8547
	Image Compression	0.8665	0.8108	0.8943	0.8326	0.8342	0.7714	0.8803	0.8191	0.9152	0.8589
	Median Blur	0.8642	0.8094	0.8945	0.8334	0.8329	0.7693	0.8815	0.8203	**0.9163**	**0.8607**
	Optical Distortion	0.8723	0.8190	0.8965	0.8361	0.8355	0.7727	0.8784	0.8170	0.9158	0.8598
	Piecewise Affine	0.8772	0.8255	0.8983	0.8383	0.8524	0.7903	0.8761	0.8137	0.9149	0.8584
	Posterize	0.8642	0.8086	0.8952	0.8337	0.8287	0.7653	0.8779	0.8158	0.9145	0.8579
	RBC	0.8599	0.8039	0.8951	0.8333	0.8359	0.7712	0.8758	0.8134	0.9125	0.8552
	Random Crop	0.8651	0.8089	0.8917	0.8299	0.8283	0.7654	0.8786	0.8168	0.9134	0.8564
	Random Gamma	0.8634	0.8082	0.8911	0.8292	0.8293	0.7654	0.8798	0.8181	0.9134	0.8563
	Random Snow	0.8675	0.8117	0.8930	0.8316	0.8412	0.7772	0.8752	0.8126	0.9142	0.8575
	Rotate	**0.8812**	**0.8285**	0.8988	0.8391	0.8578	0.7942	0.8667	0.8030	0.9125	0.8546
	Sharpen	0.8619	0.8067	0.8949	0.8341	0.8304	0.7668	**0.8838**	**0.8226**	0.9151	0.8592
	Shift Scale Rotate	0.8773	0.8251	0.8986	0.8394	**0.8600**	**0.7968**	0.8742	0.8114	0.9157	0.8593
Stylegan	CLAHE	0.8594	0.8022	0.8939	0.8314	0.8329	0.7696	0.8797	0.8184	0.9133	0.8563
	Coarse Dropout	0.8662	0.8117	0.8963	0.8360	0.8376	0.7740	0.8798	0.8185	0.9146	0.8580
	Elastic Transform	0.8793	0.8278	**0.9006**	**0.8413**	0.8486	0.7862	0.8763	0.8144	0.9157	0.8594
	Emboss	0.8640	0.8094	0.8939	0.8328	0.8277	0.7647	0.8803	0.8192	0.9136	0.8570
	Flip	0.8721	0.8173	0.8972	0.8364	0.8457	0.7832	0.8656	0.8016	0.9106	0.8519
	Gaussian Blur	0.8657	0.8112	0.8955	0.8348	0.8231	0.7597	0.8802	0.8188	0.9135	0.8571
	Grid Distortion	0.8782	0.8265	0.9004	0.8405	0.8537	0.7903	0.8757	0.8136	0.9154	0.8589
	Grid Dropout	0.8639	0.8084	0.8924	0.8313	0.8348	0.7719	0.8762	0.8143	0.9120	0.8548
	Image Compression	0.8682	0.8130	0.8935	0.8320	0.8367	0.7734	0.8819	0.8212	0.9150	0.8586
	Median Blur	0.8627	0.8073	0.8936	0.8325	0.8281	0.7637	0.8808	0.8195	0.9151	0.8588
	Optical Distortion	0.8716	0.8175	0.8964	0.8359	0.8385	0.7762	0.8800	0.8188	0.9156	0.8594
	Piecewise Affine	0.8800	0.8286	0.8982	0.8380	0.8538	0.7903	0.8768	0.8146	0.9154	0.8588
	Posterize	0.8642	0.8095	0.8954	0.8339	0.8321	0.7684	0.8777	0.8161	0.9141	0.8571
	RBC	0.8604	0.8044	0.8949	0.8334	0.8329	0.7690	0.8781	0.8163	0.9136	0.8564
	Random Crop	0.8658	0.8098	0.8926	0.8307	0.8312	0.7680	0.8800	0.8190	0.9123	0.8555
	Random Gamma	0.8674	0.8112	0.8923	0.8302	0.8302	0.7667	0.8809	0.8200	0.9138	0.8571
	Random Snow	0.8669	0.8112	0.8937	0.8319	0.8388	0.7751	0.8750	0.8127	0.9131	0.8563
	Rotate	**0.8799**	**0.8259**	0.8997	0.8401	**0.8597**	**0.7969**	0.8679	0.8043	0.9126	0.8548
	Sharpen	0.8641	0.8085	0.8962	0.8349	0.8364	0.7734	**0.8844**	**0.8236**	**0.9160**	**0.8602**
	Shift Scale Rotate	0.8777	0.8253	0.8990	0.8394	0.8471	0.7849	0.8728	0.8098	0.9148	0.8582

In Table 6.27, the proposed augmentation used images with the lesions positioned on the opposite lung side from the original dataset image. For the images generated by Stargan, the proposed augmentation achieved the highest F-scores when combined with Shift Scale Rotate on the MosMed dataset, with Sharpen on the Ricord1a dataset, and with Grid Distortion and Image Compression on the Zenodo dataset. For the images generated by Stylegan, the proposed

augmentation achieved the highest F-scores when combined with Random Snow on the MosMed dataset and Sharpen on the Ricord1a and Zenodo datasets.

Table 6.28: Results of the proposed data augmentation applied in combination with the traditional data augmentation techniques in the unified training set. The images were generated considering the minimum saliency distance between the image generated by the GAN and the lesion image from the dataset. The lesions are placed in a random lung side from the original image from the dataset. The values highlighted in green show the data augmentation techniques in which the P-value achieved values lower than 0.05, and thus the null hypothesis was rejected (i.e., there is a statistical difference and the results achieved are better than without data augmentation). The blue-colored values indicate the best F-score values, and the red-colored values indicate the best IoU values. The underscored values show the techniques in which the proposed augmentation combined with the traditional data augmentation techniques achieved a P-value lower than 0.05 when compared to just applying the generic data augmentation techniques presented on Subsection 6.2.2, and the null hypothesis was rejected.

GAN	Augmentation	CC-CCII		MedSeg		MosMed		Ricord1a		Zenodo	
		F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU	F-score	IoU
	No Augmentation	0.8636	0.8087	0.8881	0.8253	0.8185	0.7547	0.8599	0.7947	0.9096	0.8514
Stargan	CLAHE	0.8573	0.8014	0.8924	0.8302	0.8303	0.7668	0.8793	0.8175	0.9118	0.8543
	Coarse Dropout	0.8689	0.8133	0.8954	0.8344	0.8363	0.7735	0.8793	0.8179	0.9142	0.8576
	Elastic Transform	0.8769	0.8251	**0.9010**	**0.8415**	0.8535	0.7914	0.8754	0.8132	0.9159	0.8596
	Emboss	0.8711	0.8162	0.8949	0.8340	0.8284	0.7647	0.8806	0.8194	0.9150	0.8583
	Flip	0.8717	0.8175	0.8960	0.8351	0.8487	0.7860	0.8681	0.8045	0.9106	0.8523
	Gaussian Blur	0.8645	0.8089	0.8936	0.8331	0.8285	0.7647	0.8800	0.8185	0.9145	0.8584
	Grid Distortion	**0.8781**	**0.8256**	0.8999	0.8399	0.8555	0.7928	0.8769	0.8147	**0.9162**	0.8599
	Grid Dropout	0.8642	0.8080	0.8928	0.8312	0.8311	0.7686	0.8750	0.8128	0.9115	0.8536
	Image Compression	0.8639	0.8084	0.8918	0.8302	0.8283	0.7641	0.8791	0.8177	0.9131	0.8565
	Median Blur	0.8626	0.8072	0.8949	0.8342	0.8286	0.7651	0.8812	0.8200	0.9152	0.8590
	Optical Distortion	0.8748	0.8206	0.8970	0.8365	0.8337	0.7708	0.8792	0.8177	0.9158	0.8597
	Piecewise Affine	0.8767	0.8247	0.9000	0.8401	**0.8600**	**0.7983**	0.8771	0.8151	0.9161	0.8598
	Posterize	0.8623	0.8071	0.8946	0.8331	0.8332	0.7690	0.8771	0.8154	0.9132	0.8564
	RBC	0.8578	0.8023	0.8985	0.8372	0.8387	0.7752	0.8788	0.8167	0.9128	0.8556
	Random Crop	0.8669	0.8110	0.8938	0.8319	0.8336	0.7708	0.8809	0.8199	0.9128	0.8561
	Random Gamma	0.8652	0.8089	0.8930	0.8308	0.8308	0.7672	0.8803	0.8191	0.9130	0.8559
	Random Snow	0.8698	0.8142	0.8933	0.8313	0.8370	0.7745	0.8749	0.8120	0.9137	0.8568
	Rotate	0.8777	0.8248	0.8984	0.8386	0.8578	0.7937	0.8657	0.8017	0.9118	0.8538
	Sharpen	0.8638	0.8088	0.8957	0.8342	0.8298	0.7665	**0.8840**	**0.8227**	0.9160	**0.8600**
	Shift Scale Rotate	0.8771	0.8246	0.8978	0.8385	0.8554	0.7922	0.8734	0.8105	0.9152	0.8585
Stylegan	CLAHE	0.8554	0.7999	0.8910	0.8281	0.8343	0.7704	0.8757	0.8137	0.9121	0.8546
	Coarse Dropout	0.8690	0.8138	0.8959	0.8353	0.8346	0.7714	0.8806	0.8193	0.9136	0.8573
	Elastic Transform	0.8794	0.8278	0.8999	0.8404	0.8496	0.7873	0.8751	0.8126	0.9158	0.8595
	Emboss	0.8654	0.8111	0.8936	0.8323	0.8301	0.7659	0.8793	0.8180	0.9140	0.8570
	Flip	0.8746	0.8206	0.8982	0.8376	0.8466	0.7851	0.8678	0.8043	0.9109	0.8523
	Gaussian Blur	0.8687	0.8141	0.8952	0.8346	0.8280	0.7654	0.8819	0.8209	0.9144	0.8584
	Grid Distortion	0.8783	0.8264	**0.9013**	**0.8414**	0.8582	0.7948	0.8768	0.8147	**0.9167**	**0.8607**
	Grid Dropout	0.8652	0.8092	0.8937	0.8324	0.8382	0.7749	0.8759	0.8134	0.9110	0.8534
	Image Compression	0.8682	0.8134	0.8948	0.8332	0.8344	0.7717	0.8819	0.8209	0.9132	0.8567
	Median Blur	0.8654	0.8105	0.8942	0.8335	0.8293	0.7669	0.8808	0.8196	0.9153	0.8594
	Optical Distortion	0.8722	0.8188	0.8990	0.8390	0.8361	0.7731	0.8807	0.8194	0.9159	0.8599
	Piecewise Affine	0.8779	0.8267	0.8972	0.8372	0.8514	0.7889	0.8750	0.8126	0.9146	0.8582
	Posterize	0.8633	0.8070	0.8952	0.8338	0.8313	0.7674	0.8765	0.8144	0.9130	0.8558
	RBC	0.8597	0.8034	0.8956	0.8342	0.8397	0.7762	0.8785	0.8167	0.9128	0.8559
	Random Crop	0.8650	0.8096	0.8918	0.8298	0.8264	0.7631	0.8794	0.8177	0.9135	0.8565
	Random Gamma	0.8653	0.8098	0.8929	0.8311	0.8280	0.7651	0.8820	0.8206	0.9139	0.8571
	Random Snow	0.8652	0.8090	0.8941	0.8323	0.8415	0.7777	0.8763	0.8137	0.9135	0.8563
	Rotate	0.8786	0.8257	0.8989	0.8389	0.8551	0.7921	0.8673	0.8036	0.9123	0.8545
	Sharpen	0.8651	0.8103	0.8941	0.8330	0.8308	0.7675	**0.8827**	**0.8217**	0.9155	0.8596
	Shift Scale Rotate	**0.8810**	**0.8288**	0.8991	0.8397	**0.8583**	**0.7955**	0.8740	0.8112	0.9154	0.8588

In Table 6.28, the proposed augmentation used images with the lesions positioned on a random lung from the original dataset image. For the images generated by Stargan, the

proposed augmentation achieved the highest F-scores when combined with Piecewise Affine on the MosMed dataset, with Sharpen on the Ricord1a dataset, and with Grid Distortion on the Zenodo dataset. For the images generated by Stylegan, the proposed augmentation achieved the highest F-scores when combined with Shift Scale Rotate on the MosMed dataset, Sharpen on the Ricord1a dataset, and Grid Distortion on the Zenodo dataset.

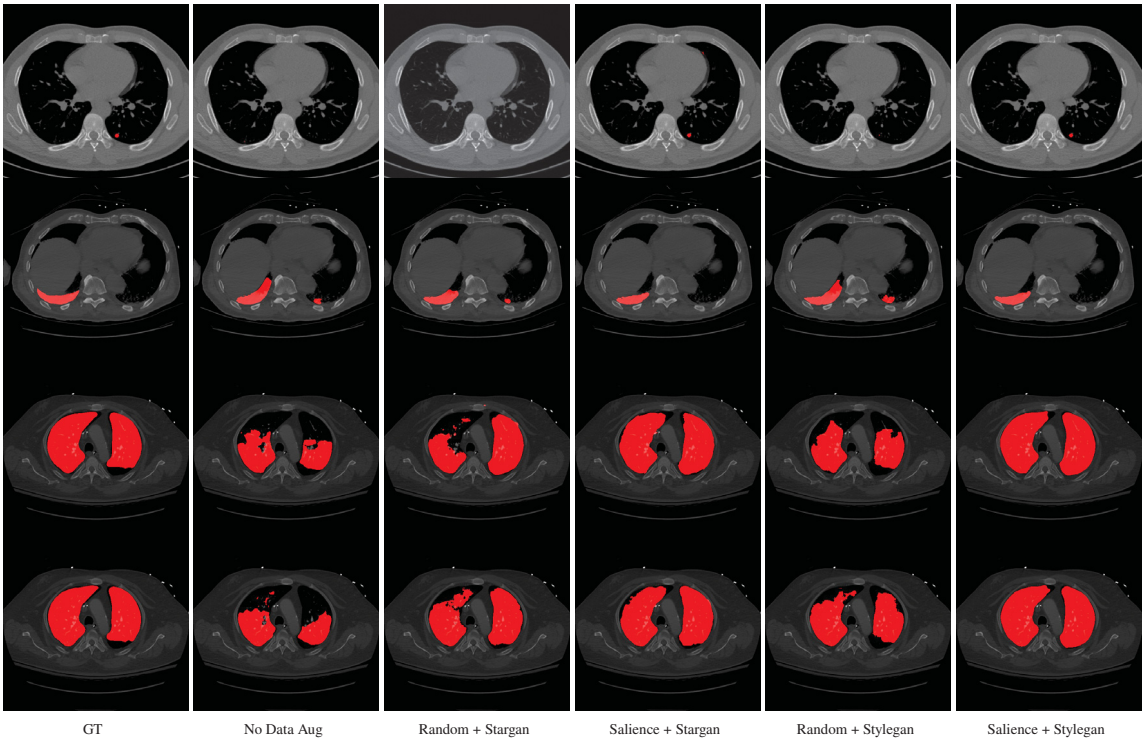


Figure 6.26: Qualitative comparison between the SalDACov version of the proposed data augmentation, the baseline without data augmentation, and the DACov version of the proposed data augmentation. With the SalDACov version of the proposed data augmentation, the segmentation network found the lesion region closer to the expected in the ground truth.

Moreover, the SalDACov version significantly improved mask quality in some cases, as illustrated in Figure 6.26. In the first row, the segmentation network failed to detect lesions without data augmentation and with the method proposed in (Krinski et al., 2023). This case is very challenging due to the small size of the lesion, with the segmentation network failing to identify the lesion when trained without data augmentation and when trained with the augmentation proposed by (Krinski et al., 2023). This case represents the initial stages of COVID-19, and, without data augmentation or with the method proposed by (Krinski et al., 2023), it would not be possible to provide an early diagnostic of positive for COVID-19, increasing the risks of virus spreading and highly increasing risks for the patient (Woloshin et al., 2020; Wikramaratna et al., 2020; Kanji et al., 2021).

In the second row, without data augmentation and the method proposed in (Krinski et al., 2023), the network showed a low recall, predicting a false positive area in the left lung. There was low precision in the third and fourth rows with many false negative pixels in the masks generated with no data augmentation and the method proposed by (Krinski et al., 2023), resulting in a low F-score on an image where the lesions nearly covered the entire lungs.

6.4 CONCLUSION

The first experiment performed in this work was an evaluation of 120 segmentation models on the COVID-19 CT segmentation problem. The findings of this first evaluation pointed out the combination of the RegNetx-002 as encoder and U-Net++ as decoder, a promising segmentation network to perform segmentation of COVID-19 CT images. The U-Net++ achieved the most promising results. Meanwhile, the RegNetx-002 is a small network with fewer parameters, requiring less training time.

The second experiment performed evaluated 20 augmentation techniques. Firstly, the augmentation techniques were performed with the segmentation model trained in each dataset individually. Then, all training sets were unified in one big training set, with the augmentation techniques being applied on top of this unified training set. The first evaluation did not clear the best augmentation techniques or the best probability to use when using data augmentation on the COVID-19 CT segmentation problem. These questions were clarified only in the second evaluation, with the results pointing to spatial transform augmentation techniques like Elastic Transform, Flip, Grid Distortion, Piecewise Affine, Rotate, and Shift Scale Rotate as the most promising techniques. Furthermore, the results indicated the higher probabilities as more promising when applying an augmentation technique in the COVID-19 CT segmentation problem.

The third experiment evaluated a new data augmentation technique in the COVID-19 CT segmentation problem. The new data augmentation used two GANs to generate healthy lung images, a segmentation model to generate lung masks to the healthy images, and two algorithms to add new lesions to the healthy images: an initial version that randomly adds lesions to the healthy images and an improved version which uses visual salience distance to add lesions to the healthy images. The SalDACov version of the proposed augmentation improved the results on the Ricord1a dataset and achieved higher results than the generic augmentation techniques in this dataset. Furthermore, this version of the proposed augmentation was evaluated with the generic augmentation, generating the highest results in three of five datasets.

7 CONCLUSION AND FUTURE WORKS

Since 2019 the world has struggled with the coronavirus pandemic (COVID-19), with millions of infections and deaths worldwide (Wang et al., 2020). In this scenario, automatic detection of COVID-19 infections in Computed Tomography (CT) scans has shown to be a great help for early diagnoses (Shi et al., 2021), with the Semantic Segmentation (Cao and Bao, 2020) of CT scans with deep learning-based approaches being widely explored since the COVID-19 outbreak (Shi et al., 2021; Narin et al., 2021). Furthermore, performing automatic segmentation of COVID-19 CT images assists doctors in diagnosing and quantifying COVID-19 lesions in a way to avoid human subjectivity (Zhang et al., 2022; Anthimopoulos et al., 2016). The spreading velocity of the virus increases the number of infected and causes a massive shortage of test kits for Reverse-Transcription Polymerase Chain Reaction (RT-PCR), the primary tool for diagnosing COVID-19. It is also worth mentioning that RT-PCR tests have high false negative rates (Ai et al., 2020). Thus, deep learning methods for COVID-19 CT segmentation have become an essential supplementary tool for RT-PCR tests (Zhang et al., 2022).

Aiming to perform the segmentation of COVID-19 CTs, many studies apply Deep Learning techniques and Deep Neural Networks, achieving impressive results in the task (Shi et al., 2021). Deep Neural Networks are widely applied in segmentation problems due to their great generalization capacity, learning to represent different classes of objects (Garcia-Garcia et al., 2018; Krinski et al., 2019). However, with new approaches being proposed quickly, an urgency aggravated by the global pandemic, the need for an extensive evaluation becomes apparent. To provide a standardized comparison of different architectures across multiple recently proposed datasets, the first step of this work was an extensive benchmark of multiple encoders and decoders with a total of 120 architectures evaluated. The models were trained and evaluated across five different CTs datasets: MedSeg (MedSeg, 2021), Zenodo (Jun et al., 2020), CC-CCII (Zhang et al., 2020), MosMed (Morozov et al., 2020), Ricord1a (Tsai et al., 2020). Each dataset was validated through a five-fold cross-validation strategy, totaling 3.000 experiments.

As expected, the comparison of different architectures revealed that there is no definitive answer for which encoder-decoder combination is the best to be applied in the approached problem. However, the evaluation provided a robust guideline for future works as encoder-decoder selection, setting of variables like the number of network parameters, and training time. In terms of decoders, the U-Net and U-Net++, widely applied in the literature, achieved impressive results. However, the LinkNet and the Multi-scale Attention Net (MA-Net) achieved very close results to them. Also, the Feature Pyramid Network (FPN) and Pyramid Scene Parsing Network (PSPNet) presented the worst results, appearing to be unsuitable for this problem. In terms of encoders, the performance varied depending on decoder combination and dataset applied. In general, they achieved close results, with none of them standing out.

Also, this analysis revealed some weak points to be approached. The major problem is the critical class imbalance within the datasets, which affects the learning process of deep neural networks. So, the next step of this work was an extensive analysis of how different data augmentation techniques improve the training of encoder-decoder neural networks on the COVID-19 CT segmentation. A total of 20 traditional different data augmentation techniques were evaluated with two training strategy. Firstly, the augmentation techniques were applied to the training set of each dataset separately, generating over 5.000 experiments. Than, the training subsets are merged into a larger set while the testing procedure remained unchanged, generating over 1.000 experiments. The impact of increasing the number of augmented images was also

evaluated with 10 probabilities of applying data augmentation: 0.05, 0.1, 0.15, 0.2, 0.25, 0.30, 0.35, 0.4, 0.45, and 0.5.

The first data augmentation experiments revealed distinct trends across datasets. In CC-CCII, MedSeg, MosMed, and Zenodo datasets, spatial transformations notably led to the highest F-scores, likely due to their ability to introduce variations in lesion region shapes. Conversely, spatial operations proved counterproductive in Ricord1a, characterized by images with minimal lung position and shape alterations. Instead, a color operation (Median Blur) achieved the highest F-score, indicating the dataset's heightened sensitivity to color alterations. However, despite these observations, the experiment yielded insignificant findings. Augmentation techniques, when compared to the baseline where the network was trained without augmentation, resulted in marginal improvements. Additionally, the slight differences observed in the effectiveness of augmentation techniques at varying probabilities further clouded the determination of an optimal probability in this context.

However, with the unified training strategy, the data augmentation techniques consistently proved more effective, yielding better F-scores and Intersection over Union (IoU) values across four out of five evaluated datasets, with the Ricord1a dataset standing out as an exception not achieving better results with the addition of data augmentation. Furthermore, augmenting data in the unified training set consistently yielded superior outcomes, clarifying that increasing the probability of the data augmentation techniques generates better results, opposing previously works from literature (Müller et al., 2020, 2021). Additionally, the results show that the best operations in both training strategies are those that change shape instead of colors, such as Grid Distortion, Optical Distortion, Flip, Piecewise Affine, and Shift Scale Rotate.

Beside the promising results achieved with the generic data augmentation techniques, these techniques generally did not perform so well, and the Semantic Segmentation of COVID-19 CTs has been a challenging problem for data augmentation. So, this work focus on studying and developing a domain-specific data augmentation that uses Visual Salience concepts. The proposed data augmentation technique employs a Generative Adversarial Network (GAN) model to produce new CT scans of healthy lungs and then combines existing labeled lesions with those new images to generate new samples and boost the segmentation performance. In a first implementation of the proposed augmentation, a matching algorithm randomly combines the new images generated by GANs with an image with COVID-19 lesions from the dataset. Then, in a second implementation, the random algorithm is replaced by a visual salience based algorithm. Furthermore, a segmentation model trained with lung labels from the datasets automatically generates the segmentation labels for the healthy CT images generated by both GANs. This step guides the augmentation algorithm to place the lesions from the dataset inside the lung areas of the generated healthy CT images.

Both version were evaluated through a 5-fold cross-validation strategy and 10 probabilities of applying data augmentation: 0.05, 0.1, 0.15, 0.2, 0.25, 0.30, 0.35, 0.4, 0.45, and 0.5. The DACov version of the proposed augmentation is limited to either flipping or not flipping the image from the dataset before adding it to the healthy image generated by the GAN, resulting in over 200 experiments. Meanwhile, nine variations of the proposed augmentation are evaluated based on saliency distance (maximum, minimum, and random) and the side of the lesion in the original dataset image (same side, opposite side, and random), resulting in over 900 experiments.

The DACov version of the proposed data augmentation technique demonstrated better F-scores in the Ricord1a dataset (previously identified as more susceptible to color alterations) when probabilities were set at 0.2, 0.25, 0.35, 0.4, 0.45, and 0.5. Also, it achieved competitive results compared to the traditional approaches on the CC-CCII, MedSeg, MosMed, and Zenodo datasets, which were previously identified as more susceptible to shape alterations. The SalDACov

version of the proposed augmentation method yielded statistical significance in four out of the five evaluated datasets compared to training the segmentation model without any augmentation. Moreover, it demonstrated promising results in the Ricord1a dataset and achieved higher values on the Ricord1a dataset compared with the generic data augmentation techniques and the random version.

In general, changing the position of the lesion from the original dataset (opposite side and random side) did not show a difference from maintaining it on the original side. Also, the salience augmentation achieved close results with the images generated by the Stargan and the Stylegan. However, when changing the saliency distance (maximum, minimum, and random) presented the highest difference, with the minimum salience distance presenting the best results. When the images were generated with the minimum salience distance between the image generated by the GAN and the lesion image from the dataset, and the lesions were positioned on the same lung side as the original image from the dataset, the proposed augmentation showed an improvement of 3% when evaluated with probabilities of 35%, 45%, and 50%, and with probabilities 5%, 15%, 35%, and 50%, achieved the higher values on Ricord1a dataset when compared with the generic data augmentation techniques and the DACov version of the proposed augmentation.

The proposed salience algorithm was combined with traditional data augmentation techniques, resulting in over 600 experiments. It achieved statistical significance across all five datasets and obtained better F-scores in the MosMed, Ricord1a, and Zenodo datasets compared to traditional techniques alone. For the images generated by the Stargan, the proposed augmentation achieved the highest F-scores when combined with Rotate on the MosMed dataset, Gaussian Blur on the Ricord1a dataset, and Optical Distortion on the Zenodo dataset. For the images generated by Stylegan, the proposed augmentation achieved the highest F-scores when combined with Piecewise Affine on the MosMed dataset, Sharpen on the Ricord1a dataset, and Median Blur on the Zenodo dataset.

Besides the new samples generated by the proposed augmentation helped to improve the results on other datasets, it is a limitation of the proposed technique that the healthy images and lesions are from the same dataset. This reduce the variability of the generated images. Furthermore, the segmentation model was trained with the set of datasets evaluated in this work. For new datasets it would be necessary to train the segmentation model again, which demands a high amount of time and computacional resources.

7.1 FUTURE WORK

Beside the findings of this work, the segmentation of COVID-19 CT images is an open problem with many study cases left for future works. Some of them are presented bellow:

- Evaluation of segmentation models with cross-datasets. This is an open evaluation in the literature and no study has evaluated the behavior of the segmentation models on cross-dataset training for the COVID-19 segmentation problem. Since the size of the datasets available are a limiting factor, cross-dataset could be another solution to mitigate this problem.

- Evaluation of ensemble of segmentation models. Ensemble of networks uses the classification of more than one network to generate the final classification or segmentation. By combining segmentation models, the final segmentation is generated through a voting system where the combination of the majority suppress flaws of individual models.

- Evaluation of mixing data augmentation techniques. This is another open evaluation left for further steps. The objective of this evaluation is to increase the number of images in the datasets by creating artificial data with different augmentation techniques and not restricted to just one augmentation.
- Evaluation of more recent GAN models for healthy CT images generation using the same methodology presented in this work.
- Evaluation of the proposed augmentation in others lungs segmentation problems. The proposed method can also be used to increase the number of training samples in other lung segmentation problems like Chronic Obstructive Pulmonary Disease (COPD), lung cancer, pneumonia, and interstitial lung disease.
- Evaluate the proposed augmentation with different salience features. There are several ways to represent visual salience, and other representations can be explored in future works to make the feature vector more robust.
- Evaluated the proposed augmentation with healthy images generated by GANs trained with other datasets and addition of COVID-19 lesions from other datasets.

REFERENCES

Ahmed, I., Chehri, A., and Jeon, G. (2022). A Sustainable Deep Learning-Based Framework for Automated Segmentation of COVID-19 Infected Regions: Using U-Net with an Attention Mechanism and Boundary Loss Function. *Electronics*, 11(15):2296.

Ai, T., Yang, Z., Hou, H., Zhan, C., Chen, C., Lv, W., Tao, Q., Sun, Z., and Xia, L. (2020). Correlation of Chest CT and RT-PCR Testing for Coronavirus Disease 2019 (COVID-19) in China: A Report of 1014 Cases. *Radiology*, 296(2):E32–E40.

Albelwi, S. and Mahmood, A. (2017). A Framework for Designing the Architectures of Deep Convolutional Neural Networks. *Entropy*, 19(6):242.

Alshomrani, S., Arif, M., and A. Al Ghamdi, M. (2023). A systematic literature review of deep learning algorithms for segmentation of the covid-19 infection. *Computers, Materials & Continua*, 75(3):5717–5742.

Amyar, A., Modzelewski, R., Li, H., and Ruan, S. (2020). Multi-task deep learning based CT imaging analysis for COVID-19 pneumonia: Classification and segmentation. *Computers in Biology and Medicine*, 126:104037.

Ankile, L. L., Heggland, M. F., and Krange, K. (2020). Deep Convolutional Neural Networks: A survey of the foundations, selected improvements, and some current applications.

Anthimopoulos, M., Christodoulidis, S., Ebner, L., Christe, A., and Mougiakakou, S. (2016). Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE Transactions on Medical Imaging*, 35(5):1207–1216.

Asad, M., Williams, H., Mandal, I., Ather, S., Deprest, J., D'hooge, J., and Vercauteren, T. (2023). *Adaptive Multi-scale Online Likelihood Network for AI-Assisted Interactive Segmentation*, page 564–574. Springer Nature Switzerland.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495.

Bertalmio, M., Bertozzi, A., and Sapiro, G. (2003). Navier-stokes, fluid dynamics, and image and video inpainting. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. IEEE Comput. Soc.

Bizopoulos, P., Vretos, N., and Daras, P. (2020). Comprehensive Comparison of Deep Learning Models for Lung and COVID-19 Lesion Segmentation in CT scans.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Buongiorno, R., Del Corso, G., Germanese, D., Colligiani, L., Python, L., Romei, C., and Colantonio, S. (2023). Enhancing covid-19 ct image segmentation: A comparative study of attention and recurrence in unet models. *Journal of Imaging*, 9(12):283.

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., and Kalinin, A. A. (2020). Albumentations: Fast and Flexible Image Augmentations. *Information*, 11(2):125.

Cai, G.-W., Liu, Y.-B., Feng, Q.-J., Liang, R.-H., Zeng, Q.-S., Deng, Y., and Yang, W. (2023). Semi-supervised segmentation of interstitial lung disease patterns from ct images via self-training with selective re-training. *Bioengineering*, 10(7):830.

Cao, F. and Bao, Q. (2020). A Survey On Image Semantic Segmentation Methods With Convolutional Neural Network. In *2020 International Conference on Communications, Information System and Computer Engineering (CISCE)*. IEEE.

Cao, K., Bi, L., Feng, D., and Kim, J. (2020). Improving PET-CT Image Segmentation via Deep Multi-modality Data Augmentation. In *Machine Learning for Medical Image Reconstruction*, pages 145–152. Springer International Publishing.

Chaurasia, A. and Culurciello, E. (2017). LinkNet: Exploiting encoder representations for efficient semantic segmentation. In *2017 IEEE Visual Communications and Image Processing (VCIP)*. IEEE.

Chen, C., Dou, Q., Chen, H., Qin, J., and Heng, P.-A. (2019). Synergistic Image and Feature Adaptation: Towards Cross-Modality Domain Adaptation for Medical Image Segmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):865–872.

Chen, H., Jiang, Y., Ko, H., and Loew, M. (2023). A teacher–student framework with fourier transform augmentation for covid-19 infection segmentation in ct images. *Biomedical Signal Processing and Control*, 79:104250.

Chen, M., Tu, C., Tan, C., Zheng, X., Wang, X., Wu, J., Huang, Y., Wang, Z., Yan, Y., Li, Z., Shan, H., Liu, J., and Huang, J. (2020a). Key to successful treatment of COVID-19: accurate identification of severe risks and early intervention of disease progression. *medRxiv*.

Chen, P., Liu, S., Zhao, H., and Jia, J. (2020b). GridMask Data Augmentation.

Chen, X., Yao, L., and Zhang, Y. (2020c). Residual attention u-net for automated multi-class segmentation of covid-19 chest ct images.

Choi, J., Lee, C., Lee, D., and Jung, H. (2021). SalfMix: A Novel Single Image-Based Data Augmentation Technique Using a Saliency Map. *Sensors*, 21(24):8444.

Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., and Choo, J. (2018). StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE.

Choi, Y., Uh, Y., Yoo, J., and Ha, J.-W. (2020). StarGAN v2: Diverse Image Synthesis for Multiple Domains. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Csurka, G., Volpi, R., and Chidlovskii, B. (2023). Semantic image segmentation: Two decades of research.

Dabouei, A., Soleymani, S., Taherkhani, F., and Nasrabadi, N. M. (2021). Supermix: Supervising the mixing data augmentation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 13789–13798. IEEE.

Das, S. P., Mitra, S., and Shankar, B. U. (2022). Collective intelligent strategy for improved segmentation of covid-19 from ct. *Expert Systems with Applications*.

Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.*, 7:1–30.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE.

Diniz, J. O. B., Quintanilha, D. B. P., Neto, A. C. S., da Silva, G. L. F., Ferreira, J. L., Netto, S. M. B., Araújo, J. D. L., Cruz, L. B. D., Silva, T. F. B., da S. Martins, C. M., Ferreira, M. M., Rego, V. G., Boaro, J. M. C., Cipriano, C. L. S., Silva, A. C., de Paiva, A. C., Junior, G. B., de Almeida, J. D. S., Nunes, R. A., Mogami, R., and Gattass, M. (2021). Segmentation and quantification of COVID-19 infections in CT using pulmonary vessels extraction and deep learning. *Multimedia Tools and Applications*, 80(19):29367–29399.

Dmitriev, K. and Kaufman, A. E. (2019). Learning Multi-Class Segmentations From Single-Class Datasets. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Dogan, A. D. A., Christensen, T. Q., Jensen, T. T., Juhl, C. B., Hilberg, O., Bladbjerg, E.-M., and Hess, S. (2024). Fdg-pet/ct-based respiration-gated lung segmentation and quantification of lung inflammation in copd patients. *BMC Research Notes*, 17(1).

Dong, N., Kampffmeyer, M., Liang, X., Wang, Z., Dai, W., and Xing, E. (2018). Unsupervised Domain Adaptation for Automatic Estimation of Cardiothoracic Ratio. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, pages 544–552. Springer International Publishing.

Dunford, R., Su, Q., and Tamang, E. (2014). The pareto principle. *The Plymouth Student Scientist*, 7:140–148.

Elharrouss, O., Subramanian, N., and Al-Maadeed, S. (2021). An encoder–decoder-based method for segmentation of covid-19 lung infection in ct images. *SN Computer Science*, 3(1).

Enshaei, N., Oikonomou, A., Rafiee, M. J., Afshar, P., Heidarian, S., Mohammadi, A., Plataniotis, K. N., and Naderkhani, F. (2022). COVID-rate: an automated framework for segmentation of COVID-19 lesions from chest CT images. *Scientific Reports*, 12(1):3212.

Erridge, P. (2006). The pareto principle. *British Dental Journal*, 201(7):419–419.

Fan, D.-P., Zhou, T., Ji, G.-P., Zhou, Y., Chen, G., Fu, H., Shen, J., and Shao, L. (2020a). Inf-Net: Automatic COVID-19 Lung Infection Segmentation From CT Images. *IEEE Transactions on Medical Imaging*, 39(8):2626–2637.

Fan, T., Wang, G., Li, Y., and Wang, H. (2020b). MA-Net: A Multi-Scale Attention Network for Liver and Tumor Segmentation. *IEEE Access*, 8:179656–179665.

Fang, H.-S., Sun, J., Wang, R., Gou, M., Li, Y.-L., and Lu, C. (2019). InstaBoost: Boosting Instance Segmentation via Probability Map Guided Copy-Pasting. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE.

Field, D. J., Hayes, A., and Hess, R. F. (1993). Contour integration by the human visual system: Evidence for a local "association field". *Vision Research*, 33(2):173–193.

Fix, E. and Hodges, J. L. (1989). Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review / Revue Internationale de Statistique*, 57(3):238.

Fung, D. L. X., Liu, Q., Zammit, J., Leung, C. K.-S., and Hu, P. (2021). Self-supervised deep learning model for COVID-19 lung CT image segmentation highlighting putative causal relationship among age, underlying disease and COVID-19. *Journal of Translational Medicine*, 19(1).

Gao, S.-H., Cheng, M.-M., Zhao, K., Zhang, X.-Y., Yang, M.-H., and Torr, P. (2021). Res2Net: A New Multi-Scale Backbone Architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(2):652–662.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., and Garcia-Rodriguez, J. (2017). A Review on Deep Learning Techniques Applied to Semantic Segmentation.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P., and Garcia-Rodriguez, J. (2018). A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, 70:41–65.

Gite, S., Mishra, A., and Kotecha, K. (2022). Enhanced lung image segmentation using deep learning. *Neural Computing and Applications*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, 63(11):139–144.

Gupta, A. U. and Singh Bhadauria, S. (2022). Multi level approach for segmentation of interstitial lung disease (ild) patterns classification based on superpixel processing and fusion of k-means clusters: Spfkmc. *Computational Intelligence and Neuroscience*, 2022:1–22.

Habili, N., Kwan, E., Li, W., Webers, C., Oorloff, J., Armin, M. A., and Petersson, L. (2023). A Hyperspectral and RGB Dataset for Building Façade Segmentation. In *Lecture Notes in Computer Science*, pages 258–267. Springer Nature Switzerland.

He, D.-C. and Wang, L. (1990). Texture unit, texture spectrum, and texture analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 28(4):509–512.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. (2020). AugMix: A Simple Method to Improve Robustness and Uncertainty under Data Shift. In *International Conference on Learning Representations*.

Heo, H.-J., Shin, U.-H., Lee, R., Cheon, Y., and Park, H.-M. (2023). Next-tdnn: Modernizing multi-scale temporal convolution backbone for speaker verification.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Hu, H., Shen, L., Guan, Q., Li, X., Zhou, Q., and Ruan, S. (2022). Deep co-supervision and attention fusion strategy for automatic COVID-19 lung infection segmentation on CT images. *Pattern Recognition*, 124:108452.

Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-Excitation Networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE.

Huang, C., Wang, Y., Li, X., Ren, L., Zhao, J., Hu, Y., Zhang, L., Fan, G., Xu, J., Gu, X., Cheng, Z., Yu, T., Xia, J., Wei, Y., Wu, W., Xie, X., Yin, W., Li, H., Liu, M., Xiao, Y., Gao, H., Guo, L., Xie, J., Wang, G., Jiang, R., Gao, Z., Jin, Q., Wang, J., and Cao, B. (2020). Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China. *The Lancet*, 395(10223):497–506.

Huang, G., Liu, Z., Maaten, L. V. D., and Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Huang, S., Wang, X., and Tao, D. (2021). Snapmix: Semantically proportional mixing for augmenting fine-grained data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(2):1628–1636.

Huang, X. and Belongie, S. (2017). Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE.

Ibrahim, M., Khalil, Y. A., Amirrajab, S., Sun, C., Breeuwer, M., Pluim, J., Elen, B., Ertaylan, G., and Dumontier, M. (2024). Generative ai for synthetic data across multiple medical modalities: A systematic review of recent developments and challenges.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-Image Translation with Conditional Adversarial Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Itti, L. (2007). Visual salience. *Scholarpedia*, 2(9):3327.

Jaccard, P. (1912). THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE.1. *New Phytologist*, 11(2):37–50.

Jaccard index (2021). "Jaccard index — Wikipedia, The Free Encyclopedia". Accessed: 2021-08-03.

Jaderberg, M., Simonyan, K., Zisserman, A., and kavukcuoglu, k. (2015). Spatial Transformer Networks. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Jiang, Y., Chen, H., Loew, M., and Ko, H. (2021). COVID-19 CT Image Synthesis With a Conditional Generative Adversarial Network. *IEEE Journal of Biomedical and Health Informatics*, 25(2):441–452.

Johns Hopkins University of Medicine (2024). Coronavirus Resource Center. `https://coronavirus.jhu.edu/`. Accessed: 2024-06-21.

Johnson, J. M. and Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1).

Joseph Raj, A. N., Zhu, H., Khan, A., Zhuang, Z., Yang, Z., Mahesh, V. G. V., and Karthik, G. (2021). Adid-unet—a segmentation model for covid-19 infection from lung ct scans. *PeerJ Computer Science*, 7:e349.

Jun, M., Cheng, G., Yixin, W., Xingle, A., Jiantao, G., Ziqi, Y., Minqing, Z., Xin, L., Xueyuan, D., Shucheng, C., Hao, W., Sen, M., Xiaoyu, Y., Ziwei, N., Chen, L., Lu, T., Yuntao, Z., Qiongjie, Z., Guoqiang, D., and Jian, H. (2020). COVID-19 CT Lung and Infection Segmentation Dataset.

Kaissis, G., Ziller, A., Passerat-Palmbach, J., Ryffel, T., Usynin, D., Trask, A., Lima, I., Mancuso, J., Jungmann, F., Steinborn, M.-M., Saleh, A., Makowski, M., Rueckert, D., and Braren, R. (2021). End-to-end privacy preserving deep learning on multi-institutional medical imaging. *Nature Machine Intelligence*, 3(6):473–484.

Kanji, J. N., Zelyas, N., MacDonald, C., Pabbaraju, K., Khan, M. N., Prasad, A., Hu, J., Diggle, M., Berenger, B. M., and Tipples, G. (2021). False negative rate of COVID-19 PCR testing: a discordant testing analysis. *Virology Journal*, 18(1).

Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2018). Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*.

Karras, T., Laine, S., and Aila, T. (2019). A Style-Based Generator Architecture for Generative Adversarial Networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2020). Analyzing and Improving the Image Quality of StyleGAN. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Khalifa, N. E. M., Manogaran, G., Taha, M. H. N., and Loey, M. (2021). A deep learning semantic segmentation architecture for COVID-19 lesions discovery in limited chest CT datasets. *Expert Systems*, 39(6).

Khan, A. U. and Borji, A. (2018). Analysis of hand segmentation in the wild. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE.

Kim, J.-H., Choo, W., and Song, H. O. (2020). Puzzle Mix: Exploiting Saliency and Local Statistics for Optimal Mixup.

Kisantal, M., Wojna, Z., Murawski, J., Naruniec, J., and Cho, K. (2019). Augmentation for small object detection. In *9th International Conference on Advances in Computing and Information Technology (ACITY 2019)*. Aircc Publishing Corporation.

Krichen, M. (2023). Convolutional neural networks: A survey. *Computers*, 12(8):151.

Krinski, B. A., Ruiz, D. V., Laroca, R., and Todt, E. (2023). DACov: a deeper analysis of data augmentation on the computed tomography segmentation problem. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, pages 1–18.

Krinski, B. A., Ruiz, D. V., Machado, G. Z., and Todt, E. (2019). Masking Salient Object Detection, a Mask Region-Based Convolutional Neural Network Analysis for Segmentation of Salient Objects. In *2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE)*. IEEE.

Krinski, B. A., Ruiz, D. V., and Todt, E. (2021). Spark in the Dark: Evaluating Encoder-Decoder Pairs for COVID-19 CT's Semantic Segmentation. In *2021 Latin American Robotics Symposium (LARS), 2021 Brazilian Symposium on Robotics (SBR), and 2021 Workshop on Robotics in Education (WRE)*. IEEE.

Krinski, B. A., Ruiz, D. V., and Todt, E. (2022). Light In The Black: An Evaluation of Data Augmentation Techniques for COVID-19 CT's Semantic Segmentation. In *Anais do XXII Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS 2022)*. Sociedade Brasileira de Computação - SBC.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

Kupyn, O., Martyniuk, T., Wu, J., and Wang, Z. (2019). DeblurGAN-v2: Deblurring (Orders-of-Magnitude) Faster and Better. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE.

Laroca, R., Araujo, A. B., Zanlorensi, L. A., Almeida, E. C. D., and Menotti, D. (2021). Towards Image-Based Automatic Meter Reading in Unconstrained Scenarios: A Robust and Efficient Approach. *IEEE Access*, 9:67569–67584.

Laroca, R., Cardoso, E., Lucio, D., Estevam, V., and Menotti, D. (2022a). On the cross-dataset generalization in license plate recognition. In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications.

Laroca, R., Santos, M., Estevam, V., Luz, E., and Menotti, D. (2022b). A First Look at Dataset Bias in License Plate Recognition. In *2022 35th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE.

Laroca, R., Zanlorensi, L. A., Estevam, V., Minetto, R., and Menotti, D. (2023). *Leveraging Model Fusion for Improved License Plate Recognition*, page 60–75. Springer Nature Switzerland.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, C.-Y., Xie, S., Gallagher, P., Zhang, Z., and Tu, Z. (2015). Deeply-Supervised Nets. In Lebanon, G. and Vishwanathan, S. V. N., editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 562–570, San Diego, California, USA. PMLR.

Li, B., Shi, Y., Qi, Z., and Chen, Z. (2018). A Survey on Semantic Segmentation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE.

Li, S., Cai, H., Qi, L., Yu, Q., Shi, Y., and Gao, Y. (2023). PLN: Parasitic-Like Network for Barely Supervised Medical Image Segmentation. *IEEE Transactions on Medical Imaging*, 42(3):582–593.

Li, W., He, C., Fang, J., Zheng, J., Fu, H., and Yu, L. (2019). Semantic Segmentation-Based Building Footprint Extraction Using Very High-Resolution Satellite Images and Multi-Source GIS Data. *Remote Sensing*, 11(4):403.

Li, Z., Liu, F., Yang, W., Peng, S., and Zhou, J. (2022). A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12):6999–7019.

Lin, T.-Y., Dollar, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). Feature Pyramid Networks for Object Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, pages 740–755. Springer International Publishing.

Liu, M.-Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., and Kautz, J. (2019). Few-Shot Unsupervised Image-to-Image Translation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE.

Liu, X., Liu, Y., Fu, W., and Liu, S. (2023). Sctv-unet: a covid-19 ct segmentation network based on attention mechanism. *Soft Computing*.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Ma, J., Wang, Y., An, X., Ge, C., Yu, Z., Chen, J., Zhu, Q., Dong, G., He, J., He, Z., Cao, T., Zhu, Y., Nie, Z., and Yang, X. (2021). Toward data-efficient learning: A benchmark for COVID-19 CT lung and infection segmentation. *Medical Physics*, 48(3):1197–1210.

Mahapatra, D. and Singh, A. (2021). CT Image Synthesis Using Weakly Supervised Segmentation and Geometric Inter-Label Relations For COVID Image Analysis.

Mahmud, T., Alam, M. J., Chowdhury, S., Ali, S. N., Rahman, M. M., Fattah, S. A., and Saquib, M. (2021a). CovTANet: A Hybrid Tri-Level Attention-Based Network for Lesion Segmentation, Diagnosis, and Severity Prediction of COVID-19 Chest CT Scans. *IEEE Transactions on Industrial Informatics*, 17(9):6489–6498.

Mahmud, T., Rahman, M. A., Fattah, S. A., and Kung, S.-Y. (2021b). Covsegnet: A multi encoder–decoder architecture for improved lesion segmentation of covid-19 chest ct scans. *IEEE Transactions on Artificial Intelligence*, 2(3):283–297.

MedSeg (2021). COVID-19 CT segmentation dataset. `http://medicalsegmentation.com/covid19/`. Accessed: 2021-05-03.

Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N., and Terzopoulos, D. (2021). Image Segmentation Using Deep Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Mirza, M. and Osindero, S. (2014). Conditional Generative Adversarial Nets.

Morozov, S., Andreychenko, A., Pavlov, N., Vladzymyrskyy, A., Ledikhova, N., Gombolevskiy, V., Blokhin, I., Gelezhe, P., Gonchar, A., and Chernina, V. (2020). MosMedData: Chest CT Scans with COVID-19 Related Findings Dataset. *ArXiv*.

Müller, D., Soto-Rey, I., and Kramer, F. (2020). Automated chest CT image segmentation of COVID-19 lung infection based on 3D U-Net. *arXiv preprint*, arXiv: 2007.04774.

Müller, D., Soto-Rey, I., and Kramer, F. (2021). Robust chest CT image segmentation of COVID-19 lung infection based on limited data. *Informatics in Medicine Unlocked*, 25:100681.

Narin, A., Kaya, C., and Pamuk, Z. (2021). Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks. *Pattern Analysis and Applications*, 24(3):1207–1220.

of Biomedical Imaging, N. I. and Bioengineering (2021). Computed Tomography (CT). [Online; accessed in 12/11/2021].

Oktay, O., Schlemper, J., Folgoc, L. L., Lee, M., Heinrich, M., Misawa, K., Mori, K., McDonagh, S., Hammerla, N. Y., Kainz, B., Glocker, B., and Rueckert, D. (2018). Attention U-Net: Learning Where to Look for the Pancreas.

Olsson, V., Tranheden, W., Pinto, J., and Svensson, L. (2021). Classmix: Segmentation-based data augmentation for semi-supervised learning. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, page 1368–1377. IEEE.

Pandey, P., Tyagi, A. K., Ambekar, S., and Prathosh, A. P. (2020). Unsupervised Domain Adaptation for Semantic Segmentation of NIR Images Through Generative Latent Search. In *Computer Vision – ECCV 2020*, pages 413–429. Springer International Publishing.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A. (2016). Context Encoders: Feature Learning by Inpainting. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Powers, D. M. W. (2008). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Mach. Learn. Technol.*, 2.

Primakov, S. P., Ibrahim, A., van Timmeren, J. E., Wu, G., Keek, S. A., Beuque, M., Granzier, R. W. Y., Lavrova, E., Scrivener, M., Sanduleanu, S., Kayan, E., Halilaj, I., Lenaers, A., Wu, J., Monshouwer, R., Geets, X., Gietema, H. A., Hendriks, L. E. L., Morin, O., Jochems, A., Woodruff, H. C., and Lambin, P. (2022). Automated detection and segmentation of non-small cell lung cancer computed tomography images. *Nature Communications*, 13(1).

Punn, N. S. and Agarwal, S. (2022). CHS-Net: A Deep Learning Approach for Hierarchical Segmentation of COVID-19 via CT Images. *Neural Processing Letters*, 54(5):3771–3792.

Qiblawey, Y., Tahir, A., Chowdhury, M. E. H., Khandakar, A., Kiranyaz, S., Rahman, T., Ibtehaz, N., Mahmud, S., Maadeed, S. A., Musharavati, F., and Ayari, M. A. (2021). Detection and Severity Classification of COVID-19 in CT Images Using Deep Learning. *Diagnostics*, 11(5):893.

Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollar, P. (2020). Designing Network Design Spaces. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Riaz, Z., Khan, B., Abdullah, S., Khan, S., and Islam, M. S. (2023). Lung tumor image segmentation from computer tomography images using mobilenetv2 and transfer learning. *Bioengineering*, 10(8):981.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Lecture Notes in Computer Science*, pages 234–241. Springer International Publishing.

Ruiz, D., Salomon, G., and Todt, E. (2020a). Can Giraffes Become Birds? An Evaluation of Image-to-image Translation for Data Generation. In *Anais do XI Computer on the Beach - COTB '20*. Universidade do Vale do Itajaí.

Ruiz, D. V., Krinski, B. A., and Todt, E. (2019). ANDA: A Novel Data Augmentation Technique Applied to Salient Object Detection. In *2019 19th International Conference on Advanced Robotics (ICAR)*. IEEE.

Ruiz, D. V., Krinski, B. A., and Todt, E. (2020b). IDA: Improved Data Augmentation Applied to Salient Object Detection. In *2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE.

Salama, W. M. and Aly, M. H. (2022). Framework for COVID-19 segmentation and classification based on deep learning of computed tomography lung images. *Journal of Electronic Science and Technology*, 20(3):100161.

Sanagavarapu, S., Sridhar, S., and Gopal, T. (2021). COVID-19 Identification in CLAHE Enhanced CT Scans with Class Imbalance using Ensembled ResNets. In *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*. IEEE.

Saood, A. and Hatem, I. (2021). COVID-19 lung CT image segmentation using deep learning methods: U-Net versus SegNet. *BMC Medical Imaging*, 21(1).

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.

Shi, F., Wang, J., Shi, J., Wu, Z., Wang, Q., Tang, Z., He, K., Shi, Y., and Shen, D. (2021). Review of Artificial Intelligence Techniques in Imaging Data Acquisition, Segmentation, and Diagnosis for COVID-19. *IEEE Reviews in Biomedical Engineering*, 14:4–15.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1).

Silva, T. (2020). An intuitive introduction to Generative Adversarial Networks (GANs). Accessed: 2021-09-13.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, pages 1–14. Computational and Biological Learning Society.

Summers, C. and Dinneen, M. J. (2019). Improved mixed-example data augmentation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, page 1262–1270. IEEE.

Sun, W., Chen, J., Yan, L., Lin, J., Pang, Y., and Zhang, G. (2022). COVID-19 CT image segmentation method based on swin transformer. *Frontiers in Physiology*, 13.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Thoma, M. (2016). A Survey of Semantic Segmentation.

Todt, E. (2005). *Visual Landmark Detection For Navigation in Outdoor Environments*. PhD thesis, Universitat Politècnica de Catalunya, Barcelona, Catalunya.

Tsai, E., Simpson, S., Lungren, M. P., Hershman, M., Roshkovan, L., Colak, E., Erickson, B. J., Shih, G., Stein, A., Kalpathy-Cramer, J., Shen, J., Hafez, M. A., John, S., Rajiah, P., Pogatchnik, B. P., Mongan, J. T., Altinmakas, E., Ranschaert, E., Kitamura, F. C., Topff, L., Moy, L., Kanne, J. P., and Wu, C. C. (2020). Medical Imaging Data Resource Center - RSNA International COVID Radiology Database Release 1a - Chest CT Covid+ (MIDRC-RICORD-1a).

Uddin, A. F. M. S., Monira, M. S., Shin, W., Chung, T., and Bae, S.-H. (2021). SaliencyMix: A Saliency Guided Data Augmentation Strategy for Better Regularization. In *International Conference on Learning Representations*.

Ümit Budak, Çıbuk, M., Cömert, Z., and Şengür, A. (2021). Efficient COVID-19 Segmentation from CT Slices Exploiting Semantic Segmentation with Integrated Attention Mechanism. *Journal of Digital Imaging*, 34(2):263–272.

USA, C. M. S. (2021). Deep Learning Innovation. [Online; accessed in 12/11/2021].

Vezakis, I. A., Georgas, K., Fotiadis, D., and Matsopoulos, G. K. (2024). Effisegnet: Gastrointestinal polyp segmentation through a pre-trained efficientnet-based network with a simplified decoder.

Vincenti, S., Villa, A., Crescenti, D., Crippa, E., Brunialti, E., Shojaei-Ghahrizjani, F., Rizzi, N., Rebecchi, M., Cas, M. D., Sole, A. D., Paroni, R., Mazzaferro, V., and Ciana, P. (2022). Increased Sensitivity of Computed Tomography Scan for Neoplastic Tissues Using the Extracellular Vesicle Formulation of the Contrast Agent Iohexol. *Pharmaceutics*, 14(12):2766.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., Vijaykumar, A., Bardelli, A. P., Rothberg, A., Hilboll, A., Kloeckner, A., Scopatz, A., Lee, A., Rokem, A., Woods, C. N., Fulton, C., Masson, C., Häggström, C., Fitzgerald, C., Nicholson, D. A., Hagen, D. R., Pasechnik, D. V., Olivetti, E., Martin, E., Wieser, E., Silva, F., Lenders, F., Wilhelm, F., Young, G., Price, G. A., Ingold, G.-L., Allen, G. E., Lee, G. R., Audren, H., Probst, I., Dietrich, J. P., Silterra, J., Webber, J. T., Slavič, J., Nothman, J., Buchner, J., Kulick, J., Schönberger, J. L., de Miranda Cardoso, J. V., Reimer, J., Harrington, J., Rodríguez, J. L. C., Nunez-Iglesias, J., Kuczynski, J., Tritz, K., Thoma, M., Newville, M., Kümmerer, M., Bolingbroke, M., Tartre, M., Pak, M., Smith, N. J., Nowaczyk, N., Shebanov, N., Pavlyk, O., Brodtkorb, P. A., Lee, P., McGibbon, R. T., Feldbauer, R., Lewis, S., Tygier, S., Sievert, S., Vigna, S., Peterson, S., More, S., Pudlik, T., Oshima, T., Pingel, T. J., Robitaille, T. P., Spura, T., Jones, T. R., Cera, T., Leslie, T., Zito, T., Krauss, T., Upadhyay, U., Halchenko, Y. O., and and, Y. V.-B. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272.

Vondrick, C., Pirsiavash, H., and Torralba, A. (2016). Generating Videos with Scene Dynamics. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Wang, C., Horby, P. W., Hayden, F. G., and Gao, G. F. (2020). A novel coronavirus outbreak of global health concern. *The Lancet*, 395(10223):470–473.

Wang, L. and He, D.-C. (1990). Texture classification using texture spectrum. *Pattern Recognition*, 23(8):905–910.

Wang, L., Zhou, H., Xu, N., Liu, Y., Jiang, X., Li, S., Feng, C., Xu, H., Deng, K., and Song, J. (2023). A general approach for automatic segmentation of pneumonia, pulmonary nodule, and tuberculosis in ct images. *iScience*, 26(7):107005.

Wikramaratna, P. S., Paton, R. S., Ghafari, M., and Lourenço, J. (2020). Estimating the false-negative test probability of SARS-CoV-2 by RT-PCR. *Eurosurveillance*, 25(50).

Winther, H. B., Gutberlet, M., Hundt, C., Kaireit, T. F., Alsady, T. M., Schmidt, B., Wacker, F., Sun, Y., Dettmer, S., Maschke, S. K., Hinrichs, J. B., Jambawalikar, S., Prince, M. R., Barr, R. G., and Vogel-Claussen, J. (2019). Deep semantic lung segmentation for tracking potential pulmonary perfusion biomarkers in chronic obstructive pulmonary disease (copd): The multi-ethnic study of atherosclerosis copd study. *Journal of Magnetic Resonance Imaging*, 51(2):571–579.

Woloshin, S., Patel, N., and Kesselheim, A. S. (2020). False Negative Tests for SARS-CoV-2 Infection — Challenges and Implications. *New England Journal of Medicine*, 383(6):e38.

Woo, S., Park, J., Lee, J.-Y., and Kweon, I. S. (2018). CBAM: Convolutional Block Attention Module. In *Computer Vision – ECCV 2018*, pages 3–19. Springer International Publishing.

Wu, J., Zhang, C., Xue, T., Freeman, W. T., and Tenenbaum, J. B. (2016). Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. In *Proceedings of*

the 30th International Conference on Neural Information Processing Systems, NIPS'16, page 82–90, Red Hook, NY, USA. Curran Associates Inc.

Xie, S., Girshick, R., Dollar, P., Tu, Z., and He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Xu, Z., Cao, Y., Jin, C., Shao, G., Liu, X., Zhou, J., Shi, H., and Feng, J. (2020). Gasnet: Weakly-supervised framework for covid-19 lesion segmentation.

Yakubovskiy, P. (2020). Segmentation Models Pytorch. `https://github.com/qubvel/segmentation_models.pytorch`.

Yang, R. and Zhang, S. (2024). Enhancing retinal vascular structure segmentation in images with a novel design two-path interactive fusion module model.

Yang, Y., Chen, J., Wang, R., Ma, T., Wang, L., Chen, J., Zheng, W.-S., and Zhang, T. (2021). Towards Unbiased Covid-19 Lesion Localisation And Segmentation Via Weakly Supervised Learning. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE.

Yao, H., gen Wan, W., and Li, X. (2022). A deep adversarial model for segmentation-assisted COVID-19 diagnosis using CT images. *EURASIP Journal on Advances in Signal Processing*, 2022(1).

Yazdekhasty, P., Zindari, A., Nabizadeh-ShahreBabak, Z., Khadivi, P., Karimi, N., and Samavi, S. (2021). Segmentation of Lungs COVID Infected Regions by Attention Mechanism and Synthetic Data.

Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., and Choe, J. (2019). CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, page 6022–6031. IEEE.

Zarate, O., Zaldívar, D., Cuevas, E., and Perez, M. (2023). Enhancing pneumonia segmentation in lung radiographs: A jellyfish search optimizer approach. *Mathematics*, 11(20):4363.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2017). mixup: Beyond Empirical Risk Minimization.

Zhang, J., Ding, X., Hu, D., and Jiang, Y. (2022). Semantic segmentation of COVID-19 lesions with a multiscale dilated convolutional network. *Scientific Reports*, 12(1).

Zhang, K., Liu, X., Shen, J., Li, Z., Sang, Y., Wu, X., Zha, Y., Liang, W., Wang, C., Wang, K., Ye, L., Gao, M., Zhou, Z., Li, L., Wang, J., Yang, Z., Cai, H., Xu, J., Yang, L., Cai, W., Xu, W., Wu, S., Zhang, W., Jiang, S., Zheng, L., Zhang, X., Wang, L., Lu, L., Li, J., Yin, H., Wang, W., Li, O., Zhang, C., Liang, L., Wu, T., Deng, R., Wei, K., Zhou, Y., Chen, T., Lau, J. Y.-N., Fok, M., He, J., Lin, T., Li, W., and Wang, G. (2020). Clinically Applicable AI System for Accurate Diagnosis, Quantitative Measurements, and Prognosis of COVID-19 Pneumonia Using Computed Tomography. *Cell*, 181(6):1423–1433.e11.

Zhang, Y., Liao, Q., Yuan, L., Zhu, H., Xing, J., and Zhang, J. (2021a). Exploiting Shared Knowledge From Non-COVID Lesions for Annotation-Efficient COVID-19 CT Lung Infection Segmentation. *IEEE Journal of Biomedical and Health Informatics*, 25(11):4152–4162.

Zhang, Y. and Yang, Q. (2022). A Survey on Multi-Task Learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609.

Zhang, Z., Ni, X., Huo, G., Li, Q., and Qi, F. (2021b). Novel coronavirus pneumonia detection and segmentation based on the deep-learning method. *Annals of Translational Medicine*, 9(11):934–934.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid Scene Parsing Network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Zhao, X., Zhang, P., Song, F., Fan, G., Sun, Y., Wang, Y., Tian, Z., Zhang, L., and Zhang, G. (2021). D2A U-Net: Automatic segmentation of COVID-19 CT slices based on dual attention and hybrid dilated convolution. *Computers in Biology and Medicine*, 135:104526.

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. (2020). Random Erasing Data Augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):13001–13008.

Zhou, T., Canu, S., and Ruan, S. (2020). Automatic COVID-19 CT segmentation using U-Net integrated spatial and channel attention mechanism. *International Journal of Imaging Systems and Technology*, 31(1):16–27.

Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., and Liang, J. (2018). U-Net++: A Nested U-Net Architecture for Medical Image Segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 3–11. Springer International Publishing.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. (2021). A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, 109(1):43–76.