

November 2020

Efficient Neural Architecture Search with Multiobjective Evolutionary Optimization

Maria Gabriela Baldeón Calisto
University of South Florida

Follow this and additional works at: <https://digitalcommons.usf.edu/etd>



Part of the [Industrial Engineering Commons](#)

Scholar Commons Citation

Baldeón Calisto, Maria Gabriela, "Efficient Neural Architecture Search with Multiobjective Evolutionary Optimization" (2020). *USF Tampa Graduate Theses and Dissertations*.
<https://digitalcommons.usf.edu/etd/8511>

This Dissertation is brought to you for free and open access by the USF Graduate Theses and Dissertations at Digital Commons @ University of South Florida. It has been accepted for inclusion in USF Tampa Graduate Theses and Dissertations by an authorized administrator of Digital Commons @ University of South Florida. For more information, please contact digitalcommons@usf.edu.

Efficient Neural Architecture Search with Multiobjective Evolutionary Optimization

by

Maria Gabriela Baldeón Calisto

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Industrial Engineering
Department of Industrial and Management Systems Engineering
College of Engineering
University of South Florida

Major Professor: Susana K. Lai-Yuen, Ph.D.
Grisselle Centeno, Ph.D.
Tapas Das, Ph.D.
Mingyang Li, Ph.D.
Lu Lu, Ph.D.
Ismail Uysal, Ph.D.

Date of Approval:
November 3, 2020

Keywords: Hyperparameter Optimization, Auto Machine Learning, Medical Image Analysis, Image Segmentation

Copyright © 2020, Maria Gabriela Baldeón Calisto

Dedication

This dissertation is dedicated to my beloved husband Nicolas and daughter Valentina whose unconditional love, motivation, and continuous support has been the cornerstone for obtaining my degree. Thank you for making my life beautiful and worth living, your love is the greatest gift of my life.

I also want to dedicate this dissertation to my parents who have raised me to become the person that I am today. Thank you for instilling in me the thirst for knowledge and teaching me to never give up.

Acknowledgments

First and foremost, I would like to thank my advisor Dr. Susana Lai-Yuen for her guidance, understanding and support throughout this academic path. Her valuable advice and insightful comments have helped me grow professionally and personally. I also want to thank Dr. Tapas Das for his encouragement and support when I needed it the most. In addition, I want to thank all my dissertation committee members for their feedback and suggestions to enrich my work. Finally, I would like to thank the Fulbright Commission and Senescyt for funding my Ph.D. studies

Table of Contents

List of Tables	iii
List of Figures.....	vi
Abstract.....	viii
Chapter 1: Introduction.....	1
1.1 Research Objectives	4
1.2 Intellectual Merit and Broader Impact.....	6
1.3 Outline of the Dissertation.....	7
Chapter 2: Literature Review.....	8
2.1 Medical Image Segmentation Techniques.....	8
2.2 Convolutional Neural Networks for Medical Image Segmentation	9
2.3 Neural Architecture Search (NAS).....	10
2.4 Multiobjective NAS	11
2.5 NAS for Medical Image Segmentation	11
2.6 Current Research Challenges	12
Chapter 3: AdaResU-Net-Multiobjective Adaptive Convolutional Neural Network for Medical Image Segmentation	14
3.1 Note to Reader.....	14
3.2 Introduction	14
3.3 Methods.....	14
3.3.1 AdaResU-Net Architecture	15
3.3.2 Hyperparameter Representation	16
3.3.3 Learning Framework.....	17
3.4 Experimental Setup	22
3.4.1 Image Pre-processing	23
3.4.2 Learning Framework Application	23
3.4.2.1 Evolution of AdaResU-Net Candidates with the MEA Algorithm.....	23
3.4.2.2 Selection of the Optimal Architecture	25
3.4.2.3 Evaluation and Comparison of the Model	25
3.5 Results	27
3.5.1 Prostate Segmentation.....	27
3.5.2 Endocardium Segmentation	30
3.5.3 Effect of the Number of Training Images on the MEA Learning Framework	31
3.5.4 Computation Time Analysis	34
3.5.5 Extension of the AdaResU-Net Model to 3D Segmentation.....	36
3.6 Discussion	37
Chapter 4: AdaEn-Net-An Ensemble of Adaptive 2D-3D Fully Convolutional Networks for Medical Image Segmentation	40
4.1 Note to Reader.....	40
4.2 Introduction	40
4.3 Methods.....	41
4.3.1 Phase I: 2D-3D FCN Model Fitting	42
4.3.1.1 FCN Structure	42
4.3.1.2 MEA Algorithm.....	45

4.3.2 Phase II: 2D-3D Ensemble Network.....	52
4.4 Experiments.....	54
4.4.1 Prostate MR Image Segmentation Challenge (PROMISE12).....	54
4.4.1.1 Dataset and Pre-processing.....	54
4.4.1.2 Implementation of the AdaEn-Net.....	54
4.4.1.3 Benchmark Results.....	57
4.4.2 Automated Cardiac Diagnosis Challenge (ACDC).....	59
4.4.2.1 Dataset and Pre-processing.....	59
4.4.2.2 Implementation of the AdaEn-Net.....	60
4.4.2.3 Benchmark Results.....	63
4.5 Discussion.....	66
Chapter 5: EMONAS-Net-Efficient Multiobjective Neural Architecture Search Framework Using Surrogate-Assisted Evolutionary Algorithm.....	69
5.1 Introduction.....	69
5.2 Methods.....	70
5.2.1 Search Space.....	70
5.2.1.1 Micro Search Space.....	71
5.2.1.2 Macro Search Space.....	73
5.2.1.3 Decision Variables in the Search Space.....	73
5.2.2 Search Algorithm.....	75
5.2.2.1 Problem Statement.....	75
5.2.2.2 Guiding the Search with Selection Probabilities.....	76
5.2.2.3 Random Forest Surrogate Model.....	79
5.2.2.4 SaMEA Algorithm.....	81
5.3 Experiments.....	86
5.3.1 Prostate MR Image Dataset (PROMISE12 Challenge).....	87
5.3.1.1 Dataset Details.....	87
5.3.1.2 Implementation Details.....	87
5.3.1.3 Benchmark Results.....	90
5.3.1.4 Search Efficiency.....	91
5.3.2 Hippocampus MR Dataset (Medical Segmentation Decathlon).....	94
5.3.2.1 Dataset Details.....	94
5.3.2.2 Implementation Details.....	94
5.3.2.3 Benchmark Results.....	96
5.3.3 Cardiac MR Dataset (ACDC Challenge).....	98
5.3.3.1 Dataset Details.....	98
5.3.3.2 Implementation Details.....	98
5.3.3.3 Benchmark Results.....	100
5.4 Discussion.....	103
Chapter 6: Conclusions.....	106
References.....	109
Appendix A: Copyright Permission.....	119

List of Tables

Table 1:	Hyperparameter search space for generating new AdaResU-Net architecture configurations.....	24
Table 2:	Parameters for the proposed MEA algorithm.....	24
Table 3:	Optimal set of hyperparameters on the Prostate MRI dataset for the AdaResU-Net architecture with the proposed MEA learning framework (AdaResU-Net MEA) and with the Bayesian Optimization approach (AdaResU-Net BO)	28
Table 4:	Evaluation metrics of the AdaResU-Net with the proposed MEA learning framework (*) and three other architectures on the Prostate MRI test set	28
Table 5:	P-values of paired t-test between two models on the Prostate MRI test set, * denotes a statistically significant difference with alpha value of 0.05 ($p < 0.05$) and ** a statistically significant difference with alpha value of 0.01 ($p < 0.01$).	29
Table 6:	Optimal set of hyperparameters on the Heart MRI dataset for the AdaResU-Net architecture with the proposed MEA learning framework (AdaResU-Net MEA) and with the Bayesian Optimization approach (AdaResU-Net BO).	31
Table 7:	Evaluation metrics of the AdaResU-Net with the proposed MEA learning framework and three other architectures on the Heart MRI test set.	31
Table 8:	P-values of paired t-test between two models on the Heart MRI test set, * denotes a statistically significant difference with alpha value of 0.05 ($p < 0.05$) and ** a statistically significant difference with alpha value of 0.01 ($p < 0.01$).	31
Table 9:	Optimal hyperparameter values of the AdaResU-Net architecture with varying dataset sizes.	33
Table 10:	Performance metrics on the Heart test set of the optimal AdaResU-Net architectures with varying dataset size	33
Table 11:	Time, in seconds, it takes the AdaResU-Net to adapt to the prostate MRI dataset and heart MRI dataset through the proposed MEA learning framework and using the Gaussian Process Bayesian optimization	35
Table 12:	Computation time, in seconds, the AdaResU-Net MEA, AdaResU-Net BO, ResU-Net and U-Net take to train and segment an image.....	35
Table 13:	Mean Dice coefficient of the proposed model in the 3D test images from the prostate dataset and competing state-of-the-art models	36
Table 14:	Mean Dice coefficient of the proposed model in the 3D test images from the cardiac dataset and competing state-of-the-art models.....	36

Table 15:	Set of hyperparameters and corresponding search space the MEA algorithm optimizes for the construction of the 2D FCN and 3D FCN	44
Table 16:	Parameters for the MEA algorithm to obtain the optimal architecture.....	52
Table 17:	Optimal hyperparameter values found with the MEA algorithm to construct the 2D FCN and 3D FCN for the prostate and cardiac datasets.	55
Table 18:	Evaluation metrics for the predicted prostate segmentation using the optimal 2D-3D FCN ensemble, 2D FCN, and 3D FCN.....	56
Table 19:	P-values for the one-tailed paired t-test between the 2D-3D FCN ensemble, 2D FCN, and 3D FCN for the prostate segmentation.....	56
Table 20:	Design type, overall score and rank score on the PROMISE12 challenge dataset.....	58
Table 21:	Mean Dice and 95% Hausdorff distance of the proposed method and five competing models on the PROMISE12 challenge dataset	58
Table 22:	Mean average boundary distance and absolute relative volume difference of the proposed method and five competing models on the PROMISE12 challenge dataset.	58
Table 23:	Evaluation metrics for the predicted segmentation of the right ventricle cavity (RVC), left ventricle cavity (LVC), and left ventricle myocardium (LVM) using the optimal 2D-3D FCN ensemble, 2D FCN, and 3D FCN.....	62
Table 24:	P-values for the one-tailed paired t-test between the 2D-3D FCN ensemble, 2D FCN, and 3D FCN for the cardiac segmentation.....	62
Table 25:	Quantitative results in the segmentation of the Right Ventricle Cavity of the proposed method (AdaEn-Net) and top competing models on the ACDC challenge dataset.....	63
Table 26:	Quantitative results in the segmentation of the Left Ventricle Cavity of the proposed method (AdaEn-Net) and top competing models on the ACDC challenge dataset.....	64
Table 27:	Quantitative results in the segmentation of the Left Ventricle Myocardium of the proposed method (AdaEn-Net) and top competing models on the ACDC challenge dataset.....	64
Table 28:	Quantitative results on the ACDC dataset between the proposed method (AdaEn-Net) and a current reinforcement learning based algorithm for neural architecture search of medical image segmentation architectures	65
Table 29:	Set of possible convolutional operations in a node	72
Table 30:	The 10 decision variables that constitute the hyperparameter search space for generating the EMONAS-Net architecture and their corresponding search range	74
Table 31:	Summary of parameter values used to implement the SaMEA algorithm	85
Table 32:	Best hyperparameter values and size of the EMONAS-Net architecture on the Prostate, Hippocampus and Cardiac MRI datasets	88
Table 33:	Description of top competing AutoML approaches on the PROMISE12 challenge dataset	90

Table 34:	Evaluation metrics for the EMONAS-Net and top competing AutoML approaches on the PROMISE12 challenge dataset.	90
Table 35:	Segmentation accuracy metrics with the different search methods.....	92
Table 36:	Search efficiency metrics with the different search methods.....	93
Table 37:	Description for the EMONAS-Net and competing architectures on the hippocampus dataset.....	97
Table 38:	Evaluation metrics for the EMONAS-Net and competing automatically and manually designed architectures on the hippocampus dataset.....	97
Table 39:	Evaluation metrics for the segmentation of the Right Ventricle Cavity of the EMONAS-Net and top competing models on the ACDC challenge dataset.....	101
Table 40:	Evaluation metrics for the segmentation of the Left Ventricle Cavity of the EMONAS-Net and top competing models on the ACDC challenge dataset.....	102
Table 41:	Evaluation metrics for the segmentation of the Left Ventricle Myocardium of the EMONAS-Net and top competing models on the ACDC challenge dataset.....	102
Table 42:	Evaluation metrics on the ACDC challenge dataset between the EMONAS-Net framework and competing NAS approaches for medical image segmentation.	103

List of Figures

Figure 1:	Traditional neural architecture search process to automatically design a neural network.....	2
Figure 2:	Examples of segmentation of medical images: a) Cardiac cine MRI, b) Segmentation of the right ventricle cavity, left ventricle cavity and left ventricle myocardium, c) Prostate MRI, d) Segmentation of the Prostate.....	3
Figure 3:	A Fully Convolutional Network (FCN) comprised of 2 convolutional layers and 1 transpose convolutional layer.....	9
Figure 4:	Fixed AdaResU-Net architecture	15
Figure 5:	Example of a genotype decoded into an AdaResU-Net architecture.....	17
Figure 6:	Overview of the proposed MEA learning framework for the AdaResU-Net model	19
Figure 7:	Experimental process for the prostate and endocardium segmentation.....	23
Figure 8:	Tested AdaResU-Net architectures and their corresponding value in the criterion space.....	27
Figure 9:	Evaluation metrics of the AdaResU-Net with the MEA learning framework (AdaResU-Net MEA: rightmost green boxplot), AdaResU-Net with Bayesian optimization (AdaResU-Net BO: middle-right blue boxplot), ResU-Net (middle-left brown boxplot) and U-Net (leftmost olive boxplot) on the test set	28
Figure 10:	Segmentation results for different cases of the Prostate MRI test set	30
Figure 11:	Segmentation results for different cases of the Heart MRI test set	32
Figure 12:	Pareto front for the heart dataset with varying number of training images	33
Figure 13:	Overview of the proposed framework for the 2D-3D FCN ensemble construction	41
Figure 14:	Schematic view of the FCN architecture.....	43
Figure 15:	Example of two possible hyperparameter values and their corresponding decoded FCN architectures	46
Figure 16:	Validation multi-class Dice coefficient (MCDice) for the candidate architectures FCN ¹ and FCN ² during the training epochs	49
Figure 17:	Flowchart of the MEA algorithm	51

Figure 18:	Examples of prostate segmentation using the optimal 2D FCN, optimal 3D FCN, and 2D-3D FCN ensemble on the validation dataset	57
Figure 19:	Examples of cardiac segmentation using the optimal 2D FCN, optimal 3D FCN, and 2D-3D FCN ensemble on the validation dataset	62
Figure 20:	An overview of the micro and macro search space for the EMONAS-Net.....	71
Figure 21:	Example of a hyperparameter vector with the 10 decision variables that define the micro- and macro-structure of the EMONAS-Net and the decoded architecture	74
Figure 22:	Flowchart of the search process applied by the proposed SaMEA algorithm	81
Figure 23:	Trained architectures (green points) and approximate Pareto Front (blue points) found with the SaMEA algorithm	88
Figure 24:	Visualization of the segmentation results on the prostate dataset using the best EMONAS-Net architecture	89
Figure 25:	Configuration of the best EMONAS-Net architectures found with the proposed NAS framework	95
Figure 26:	Visualization of the segmentation results on the hippocampus dataset using the best EMONAS-Net architecture	96
Figure 27:	Visualization of the segmentation results on the cardiac dataset using the best EMONAS-Net architecture	99

Abstract

Deep neural networks have become very successful at solving many complex tasks such as image classification, image segmentation, and speech recognition. These models are composed of multiple layers that have the capacity to learn increasingly higher-level features, without prior handcrafted specifications. However, the success of a deep neural network relies on finding the proper configuration for the task in hand. Given the vast number of hyperparameters and the massive search space, manually designing or fine-tuning deep learning architectures requires extensive knowledge, time, and computational resources.

There is a growing interest in developing methods that automatically design a neural network's architecture, known as neural architecture search (NAS). NAS is usually modeled as a single-objective optimization problem where the aim is to find an architecture that maximizes the prediction's accuracy. However, most deep learning applications require accurate as well as efficient architectures to reduce memory consumption and enable their use in computationally-limited environments. This has led to the need to model NAS as a multiple objective problem that optimizes both the predictive performance and efficiency of the network. Furthermore, most NAS framework have focused on either optimizing the micro-structure (structure of the basic cell), or macro-structure (optimal number of cells and their connection) of the architecture. Consequently, manual engineering is required to find the topology of the non-optimized structure.

Although NAS has demonstrated great potential in automatically designing an architecture, it remains a computationally expensive and time-consuming process because it requires training and evaluating many potential configurations. Recent work has focused on improving the search time of NAS algorithms, but most techniques have been developed and applied only for single-objective optimization problems. Given that optimizing multiple objectives has a higher complexity and requires more iterations to approximate the Pareto Front, it is critical to investigate algorithms that decrease the search time of multiobjective NAS.

One critical application of deep learning is medical image segmentation. Segmentation of medical images provides valuable information for various critical tasks such as analyzing anatomical structures, monitoring disease

progression, and predicting patient outcomes. Nonetheless, achieving accurate segmentation is challenging due to the inherent variability in appearance, shape, and location of the region of interest (ROI) between patients and the differences in imaging equipment and acquisition protocols. Therefore, neural networks are usually tailored to a specific application, anatomical region, and image modality. Moreover, medical image data is often volumetric requiring expensive 3D operations that result in large and complex architectures. Hence, training and deploying them requires considerable storage and memory bandwidth that makes them less suitable for clinical applications.

To overcome these challenges, the main goal of this research is to automatically design accurate and efficient deep neural networks using multiobjective optimization algorithms for medical image segmentation. The proposed research consists of three major objectives: (1) to design a deep neural network that uses a multiobjective evolutionary based algorithm to automatically adapt to different medical image datasets while minimizing the model's size; (2) to design a self-adaptive 2D-3D Fully Convolutional network (FCN) ensemble that incorporates volumetric information and optimizes both the performance and the size of the architecture; and (3) to design an efficient multiobjective neural architecture search framework that decreases the search time while simultaneously optimizing the micro- and macro-structure of the neural architecture.

For the first objective, a multiobjective adaptive convolutional neural network named AdaResU-Net is presented for 2D medical image segmentation. The proposed AdaResU-Net is comprised of a fixed architecture and a learning framework that adjusts the hyperparameters to a particular training dataset using a multiobjective evolutionary based algorithm (MEA algorithm). The MEA algorithm evolves the AdaResU-Net network to optimize both the segmentation accuracy and model size. In the second objective, a self-adaptive ensemble of 2D-3D FCN named AdaEn-Net is proposed for 3D medical image segmentation. The AdaEn-Net is comprised of a 2D FCN that extracts intra-slice and long-range 2D context, and a 3D FCN architecture that exploits inter-slice and volumetric information. The 2D and 3D FCN architectures are automatically fitted for a specific medical image segmentation task by simultaneously optimizing the expected segmentation error and size of the network using the MEA algorithm. Finally, for the third objective, an efficient multiobjective neural architecture search framework named EMONAS is presented for 3D medical image segmentation. EMONAS has two main components, a novel search space that includes the hyperparameters that define the micro- and macro-structure of the architecture, and a Surrogate-assisted multiobjective evolutionary based algorithm (SaMEA algorithm) that efficiently searches for the best hyperparameter values using a Random Forest surrogate and guiding selection probabilities.

The broader impact of the proposed research is as follows: (1) automating the design of deep neural networks' architecture and hyperparameters to improve the performance and efficiency of the models; and (2) increase the accessibility of deep learning to a broader range of organizations and people by reducing the need of expert knowledge and GPU time when automatically designing deep neural networks. In the medical area, the proposed models aim to improve the automatic extraction of data from medical images to potentially enhance diagnosis, treatment planning and survival prediction of various diseases such as cardiac disease and prostate cancer. Although the proposed techniques are applied to medical image segmentation tasks, they can also be implemented in other applications where accurate and resource-efficient deep neural networks are needed such as autonomous navigation, augmented reality and internet-of-things.

Chapter 1: Introduction

Deep learning methods have become very successful at solving a variety of complex tasks such as image classification [1] and segmentation [2], speech recognition [3], and machine translation [4]. The main reason of their success is their capability to automatically extract low, mid, and high level features from the input data in an end-to-end fashion. However, the performance of a neural network is highly dependent on the configuration of its architecture and hyperparameters. Extensive work has focused on manually designing network components for the task in hand. Given the complexity of current architectures and the vast hyperparameter search space, manually designing a neural network resembles a black-box optimization process that requires extensive experience, time, and computational resources. Furthermore, this lengthy process does not guarantee the convergence to an optimal or good solution.

To address this problem, there has been an increasing focus on developing methods that automatically design deep neural networks through the application of optimization algorithms, also known as neural architecture search (NAS). The optimization algorithms applied in NAS are usually based on reinforcement learning [5, 6], Bayesian optimization [7], evolutionary algorithms [8], and gradient-based methods [9, 10]. Evolutionary algorithms are a good choice for designing neural networks because they rely on an unconstrained optimization process that allows the use of stochastic gradient descent during training, can represent flexible architecture configurations, and can apply parallelization techniques to increase convergence speed. Moreover, multiobjective evolutionary algorithms (MOEAs) have been developed and successfully applied to solve problems where multiple and competing objectives need to be optimized simultaneously.

Although NAS has demonstrated great potential in automatically designing deep neural networks, it remains a computationally expensive and time consuming process requiring even thousands of GPU days to obtain state-of-the-art architectures [11, 8, 6]. NAS usually consists of three steps as shown in Figure 1. First, the hyperparameters of a candidate architecture are sampled from the search space. This architecture is then trained and its performance evaluated for the specific task. Using this information, the search strategy is updated to guide the search to more promising architectures. The steps are repeated iteratively until the performance measure converges to a desired threshold, or a maximum number of iterations has been reached. To decrease the computational burden, recent works

have proposed embedding the hyperparameter search space from a discrete to a continuous space [9, 10], reusing learned weights [5, 12], using a one-shot architecture search [7], or applying a surrogate function to estimate an architecture’s validation loss [13, 14]. Nevertheless, these techniques were developed for NAS methods that only optimize the model’s accuracy and do not consider additional objective functions or constraints that might be imposed by the particular application. Furthermore, most NAS frameworks have focused on either optimizing the micro-structure (structure of the basic cell), or macro-structure (optimal number of cells and their connection) of the architecture. Consequently, manual engineering is required to find the topology of the non-optimized structure.

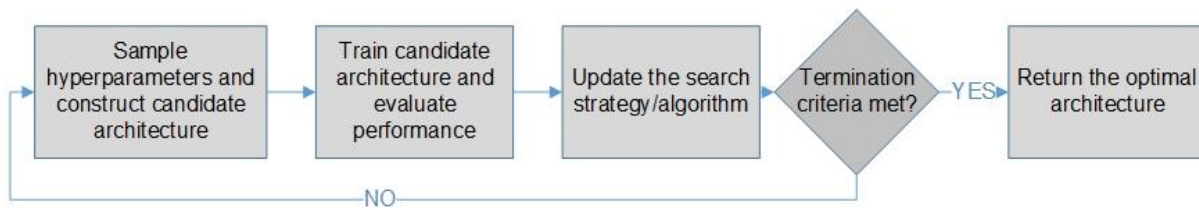


Figure 1 Traditional neural architecture search process to automatically design a neural network.

Since the deep learning breakthrough in 2012, state-of-the-art networks have greatly increased in size and complexity to achieve better performance. Hence, training and deploying them requires considerable GPU memory, computational resources and time. However, deep neural networks are usually over-parametrized, have significant redundancy in the learned weights, and are highly inefficient at exploiting their whole learning capacity [15, 16]. Recent works [17, 18, 19, 20] have shown that networks can be significantly reduced in size without a loss of accuracy. Therefore, there is a growing interest in developing multiobjective NAS methods that automatically design architectures that optimize both the predictive performance and size of the networks to reduce unnecessary computational usage and enable their use in many computationally-limited environments such as autonomous navigation, augmented reality and internet-of-things.

Among the many deep learning applications, a rapidly growing field of research is deep learning for medical image analysis, particularly for medical image segmentation. The segmentation of 3D medical images consists in detecting and identifying the boundaries of structures such as organs, bones and tumors, and is a critical task to assist in clinical diagnosis, disease detection, and pathology research. Reliable automatic segmentation methods have been widely studied because manual annotation is time-consuming, subjective and error-prone. Nevertheless, achieving accurate segmentation is challenging due to the inherent variability in appearance, shape, and location of the region of interest (ROI) between patients and the differences in imaging equipment and acquisition protocols. Therefore, models are usually tailored to a specific application, anatomical region, and image modality.

Technically, image segmentation is defined as the process of separating an image into multiple regions, where pixels from the same region share common characteristics such as color, texture or contrast [21]. In medical imaging, these regions can correspond to hard and soft tissues such as bones, organs, and muscles. For example, in Fig. 2(a) and (b), a cardiac cine MRI and the segmentation of the right ventricle cavity (red region), left ventricle cavity (blue region) and left ventricle myocardium (green region) are presented. Fig. 2(c) and (d) shows an MRI from the prostate and its corresponding segmentation in color green.

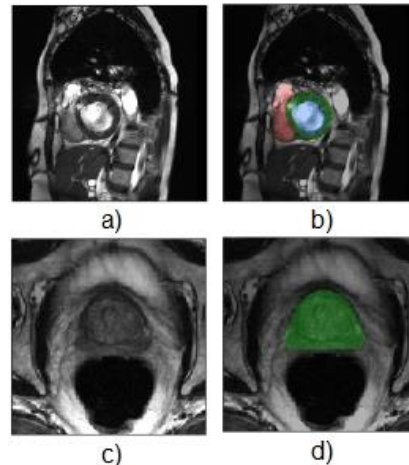


Figure 2 Examples of segmentation of medical images: a) Cardiac cine MRI, b) Segmentation of the right ventricle cavity, left ventricle cavity and left ventricle myocardium, c) Prostate MRI, d) Segmentation of the Prostate.

In contrast with most image segmentation tasks that are performed on natural images, medical image data is often volumetric requiring the methods to consider the entire volume to perform the segmentation in 3D. Two main types of networks have been proposed for handling volumetric medical image. The first type of models are 2D CNNs that segment each 2D slice independently and then concatenate the results along the third dimension [2, 22, 23]. Although these models are able to capture rich information in one plane and perform significantly well on anisotropic datasets or images that exhibit a severe misalignment along the third dimension, they do not fully exploit the spatial correlation along the z-axis affecting the segmentation accuracy. The second type of networks are 3D CNNs that replace 2D convolutions with 3D convolutions and directly process volumetric information [24, 25, 26]. Nevertheless, in this approach CNNs require a substantial number of parameters to capture representative features, suffer from high computational cost, and consume considerable GPU memory. Recent work has focused on hybrid 2D-3D CNNs to combine the strengths of 2D and 3D CNNs [27, 28]. However, these 2D-3D architectures remain considerably big and comparable in size and memory consumption to other 3D CNNs.

For these reasons, this dissertation will focus on three transcendental areas for the advancement of automatic neural architecture design for medical image segmentation: 1) to design a segmentation model that can automatically and accurately adapt to different datasets while minimizing the model's size; 2) to design an ensemble of deep neural networks that incorporates volumetric data and automatically adapts to a dataset while optimizing the model's accuracy and size; and 3) to design an efficient multiobjective neural architecture search framework that decreases the search time and simultaneously optimizes the micro- and macro-structure of the neural architecture

1.1 Research Objectives

The main goal of this research is to automatically design accurate and efficient deep neural networks using multiobjective evolutionary optimization algorithms for medical image segmentation. Although this research work focuses on designing architectures for medical image segmentation, the proposed underlying algorithms can also be applied to identify the best and most efficient neural architectures for other applications. Furthermore, the proposed self-adaptive network can be used for the segmentation of other types of images such as cell, natural, or aerial imagery. The proposed research consists of three major objectives as described below:

- *To design a deep neural network architecture for 2D MRI segmentation that automatically adapts to different medical image datasets while minimizing model's size.* A multiobjective adaptive convolutional neural network, called AdaResU-Net, is proposed for medical image segmentation. The AdaResU-Net is comprised of a fixed architecture and a learning framework that adjusts the hyperparameters to a particular training dataset. The fixed architecture combines the favorable structure of the U-Net [22] with a residual learning framework for a more efficient training. The proposed learning framework uses a multiobjective evolutionary based algorithm (MEA algorithm) to evolve the AdaResU-Net networks with different hyperparameters subject to segmentation accuracy and model size as objective functions. Experimental results show that the AdaResU-Net is able to adapt to different datasets and to achieve better performance than the state of the art U-Net, while having less than 30% the number of trainable parameters. Additionally, the MEA algorithm generates configurations that are smaller and perform better or similar than configurations obtained through a Bayesian optimization approach.
- *To design an ensemble of deep neural networks that automatically adapts to a 3D medical image dataset by incorporating volumetric data and optimizing the accuracy and size of the network.* An adaptive 2D-3D ensemble of fully convolutional networks (FCN), which incorporates volumetric information while

optimizing both the performance and model's size, is proposed for medical image segmentation. The presented adaptive ensemble, called AdaEn-Net, has two main components: a 2D FCN that extracts intra-slice and long-range 2D contexts, and a 3D FCN architecture that exploits inter-slice and volumetric information. The 2D and 3D FCN architectures have an encoder-decoder structure that is automatically fitted for a specific medical image dataset using the proposed MEA algorithm. During the search process, the proposed algorithm determines the optimal number of residual blocks, kernel sizes, and number of filters while minimizing the expected segmentation error. Thus, simultaneously optimizes the performance, width and depth of the network. The final segmentation takes advantage of the 2D FCN and 3D FCN by combining the results through an ensemble network. The AdaEn-Net is evaluated on two publically available medical image segmentation challenges. In both benchmarks, the AdaEn-Net achieved a rank within the top performing algorithms in the leaderboard. It achieves comparable performance to manually-designed architectures and surpasses the accuracy of automatically-designed architectures, while being considerably smaller in size.

- *To design an efficient multiobjective neural architecture search framework that decreases the search time and simultaneously optimizes the micro- and macro-structure of the neural architecture.* An efficient multiobjective NAS framework called EMONAS-Net is proposed for 3D medical image segmentation. EMONAS-Net has two key components: a novel search space that considers the configuration of the micro- and macro-structure of the segmentation architecture, and a surrogate-assisted multiobjective evolutionary based algorithm (SaMEA algorithm) that improves the efficiency of the architectural hyperparameter search. In relation to the search space, the micro-structure is represented by a directed acyclical graph and the SaMEA algorithm determines the connections between the nodes and the most appropriate convolutional operation for each node. For the design of the macro-structure, the search space includes the hyperparameters that define the depth and width of the architecture. The second component of the framework is the SaMEA algorithm that searches for architectures that minimize the segmentation error and number of parameters of the network. The convergence and search time of the algorithm is reduced by implementing two strategies. First, information produced during the initial stages of the evolutionary search is used to increase the probability of selecting the best performing hyperparameter values and most promising subproblems. Secondly, a Random Forest model is applied as an inexpensive surrogate function to estimate a candidate's

architecture performance and decrease the training time. The EMONAS-Net framework is tested on three medical image segmentations tasks from publically available datasets, where it is able to find networks that perform better or similar to other NAS methods while being significantly smaller and requiring considerably less computational time for the architecture search.

1.2 Intellectual Merit and Broader Impact

The proposed research aims to address major challenges in deep neural network design and hyperparameter optimization. In terms of the engineering contribution, the proposed framework can automatically search for the best and most efficient neural architecture for a particular dataset while increasing our understanding of the relationship between hyperparameter selection and neural network's performance. The presented AdaResU-Net, AdaEn-Net and EMONAS-Net have shown to adapt to distinct medical datasets, succeeding in the segmentation of 2D and 3D complex structures like the prostate, heart (left ventricle cavity, right ventricle cavity, and left ventricle myocardium) and hippocampus (anterior and posterior parts). Therefore, the proposed models can also be used in other non-medical fields that require the segmentation of single and multiple irregular and complex shaped figures. Example applications include aerial imagery segmentation, video understanding, object recognition for self-driving cars, content-based image retrieval and robot navigation.

The MEA and SaMEA algorithms, proposed to adapt the network's architecture to segment different medical images, provides a general method that solves the challenging hyperparameter optimization problem in deep neural network design. Thus, it can also be applied to optimize the hyperparameters and architecture of other types of deep neural networks. Moreover, the proposed algorithms reframe the commonly used single objective hyperparameter optimization problem to a multiobjective problem, where a model's generalization error and other objectives related to a network's efficiency can be minimized simultaneously. Given the proliferation of deep learning on embedded applications and mobile devices, the algorithms presented in this research provide a solution for incorporating additional constraints in terms of size, memory and power consumption that might be imposed by the hardware or software devices.

Finally, a transcendental contribution of the MEA and SaMEA algorithm is making deep learning more accessible to a broader range of organizations and individuals, which is also known as democratization of artificial intelligence (AI). The proposed algorithms help democratize AI by reducing the need for expert knowledge when designing deep neural networks. Moreover, considering that deploying most NAS algorithms still requires high

computational resources, the efficient SaMEA algorithm further improves the accessibility and applicability of deep learning technology by significantly reducing the convergence and search time.

In the medical field, structure segmentation is a crucial step in computer-aided diagnosis. Even though various algorithms that automatically segment medical images have been presented, it still remains a challenge to design methods that have the sufficient accuracy and consistent results to use in clinical practice. The proposed research aims to develop fully automated and adaptable algorithms that segment anatomical structures with high accuracy and robustness. This can potentially help clinicians and researchers obtain valuable information from images faster, thus improving various critical tasks such as clinical diagnosis, radiation therapy planning, and treatment response prediction.

1.3 Outline of the Dissertation

The remaining content of this dissertation is organized as follows:

- In Chapter 2, a summary of traditional and automated methods developed for medical image segmentation is introduced with a focus on convolutional neural networks. A review of single objective and multiobjective NAS methods is also provided, with an emphasis on NAS applied to medical image segmentation. Finally, the current research challenges addressed by this research are introduced.
- In Chapter 3, the multiobjective adaptive convolutional neural network for 2D medical image segmentation named AdaResU-Net is described.
- In Chapter 4 an ensemble of adaptive 2D-3D fully convolutional networks for 3D medical image segmentation, named AdaEn-Net, is presented.
- In Chapter 5 the EMONAS-Net, an efficient multiobjective neural architecture search framework using a surrogate-assisted evolutionary based algorithm for 3D medical image segmentation, is presented.
- In Chapter 6 a summary of the conclusion of this dissertation are presented.

Chapter 2: Literature Review

This chapter provides a review of major methods developed for the segmentation of medical images and approaches to automatically design neural network architectures for a specific problem. First, an introduction to traditional and automated medical image segmentation techniques is presented. Subsequently, a description of convolutional neural networks is provided with a focus on common architectures for image segmentation tasks. Then, single objective and multiobjective neural architecture search methods are introduced, with an emphasis on the most recent approaches for the construction of medical image segmentation architectures. Finally, the current challenges that will be addressed in this research are discussed.

2.1 Medical Image Segmentation Techniques

Various methods have been proposed for the segmentation of medical images. Classical methods use either intensity or texture-based features. Methods that use intensity level based features include amplitude segmentation based on histogram features [29], edge based segmentation and grouping [30], and region based segmentation [31]. The main limitations of these models are the difficulty in selecting a proper thresholding value, their sensitivity to the presence of artifacts or noise in the image, and their overall performance leading to under-segmented or over-segmented regions [32]. Some of these deficiencies can be rectified by using machine learning techniques to select the optimal segmentation criteria or combining edge based segmentation with a region based method such as proposed by Gevers et al. in [33].

Texture-based segmentation methods focus on the spatial arrangement and tone of pixels. The aim is to divide the image into regions that have similar texture properties. Commonly used approaches are the co-occurrence matrix method [34], Fourier filter method [35], and fractal texture description method [36]. Texture-based methods have shown to provide better results in medical image segmentation than intensity based methods [32]. However, the major challenges in texture segmentation are determining the types of textures that exist in an image, specifying the regions of the image where the texture is localized, and defining the most relevant descriptors and features for the texture identification.

Recently, artificial intelligence (AI) techniques have been increasingly and successfully used for automatic image segmentation. These methods can be divided into unsupervised and supervised methods. In unsupervised learning the input variables, or in this case the medical images, do not have a corresponding output variable or ground truth. Thus, the objective is to learn the underlying distribution in the data. Methods based on the K-means algorithm have been the most popular, which aim to construct decision boundaries by grouping data points into clusters [37, 38].

On the other hand, in supervised learning the input variables have a fixed output variable or ground truth. The algorithm must learn to map from the input to the output variables. The most well-known and used method in this category are artificial neural networks, specifically convolutional neural networks. Various works have been presented to segment anatomical structures in medical images such as brain tumors [39], liver [40] and prostate [41].

2.2 Convolutional Neural Networks for Medical Image Segmentation

Convolutional Neural Networks (CNNs) are a type of artificial neural networks that have proven to be very effective in computer vision problems because of their capability to recognize patterns and learn from structured data. One important application of CNNs is semantic segmentation, which is the task of assigning a class label to every pixel in the image to provide segments with semantic meaning. In [2], Long et al. presented the first CNN for end-to-end image segmentation called Fully Convolutional Network (FCN). The FCN added de-convolutional layers on top of a classification network to produce a per pixel classification result. FCNs usually have an encoder-decoder structure as shown in Fig 3. The encoder path is composed of convolutional layers, while the decoder path has transpose convolutional layers. A convolutional layer has several trainable filters that convolve on an image to extract relevant features into a feature map. This operation commonly decreases the size of the feature map. Transpose convolutional layers also have trainable filters, but are applied with a fractional stride to increase the size of the feature map.

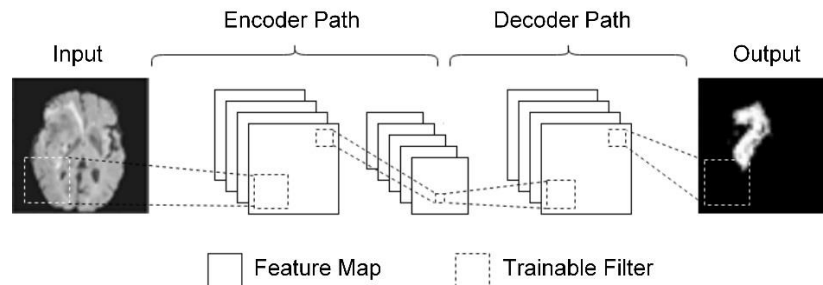


Figure 3 A Fully Convolutional Network (FCN) comprised of 2 convolutional layers and 1 transpose convolutional layer.

CNNs have been increasingly used for medical image segmentation with the U-Net [22] being the most well-known architecture. Inspired in the FCN, the U-Net is a U-shaped encoder-decoder network that has an equal number

of convolutional and transpose convolutional layers. Long skip connections are located between opposing layers in the encoder and decoder path to share information. Derived from the U-Net and FCN architectures, other networks have proposed a similar encoder-decoder structure such as the DCAN [42], SegNet [43], and fully convolutional DenseNets [44, 45].

In an effort to incorporate volumetric information, researchers have proposed 2.5D FCNs that input a stack of adjacent slices in the channel dimension [46], and triplanar CNNs that integrate the prediction of three 2D networks trained on orthogonal planes [47]. Volumetric information has also been integrated by using recurrent neural networks (RNNs). In [48], a 2D FCN is combined with a convolutional LSTM to extract intra-slice and inter-slice context. Finally, 3D FCNs have been proposed by replacing 2D convolutions with 3D convolutions [24, 26, 49]. Although 3D FCNs directly process volumetric information, they require a substantial number of parameters to capture representative features, suffer from high computational cost, and consume considerable GPU memory.

To reduce the computational burden of 3D CNNs, recent work has focused on hybrid 2D-3D CNNs to combine the strengths of 2D and 3D CNNs. In [27], Li et al. proposed a 2D-3D architecture, where a very deep 2D network obtains a coarse slice-wise segmentation and a 3D counterpart processes the results to exploit inter-slice information. In [28], Mlynarski et al. presented a model that uses three 2D CNNs to process axial, coronal and sagittal slices and a 3D CNN that combines the results and produces the final segmentation. However, the proposed architectures remain considerably big and comparable in size and memory consumption to other 3D CNNs.

2.3 Neural Architecture Search (NAS)

Neural architecture search (NAS) methods aim to automatically design neural network architectures for a specific problem. NAS can be considered a subfield of auto machine learning (AutoML) and has a significant overlap with hyperparameter optimization and meta-learning [50]. NAS has three main components: the search space, the optimization method, and the evaluation method. The search space defines the set of feasible architectures that can be represented. Global search spaces have been proposed [51, 6], in which the whole neural network is defined as a directed acyclic graph (DAG). The nodes in the DAG corresponds to an operation and the edges the forward pass of feature maps. Motivated by the observation that effective manually-designed architectures repeat fixed modules, works have also proposed cell-based search spaces [52, 53]. In this search space, the subspace is confined to represent smaller DAGs that are stacked in a predetermined template to form the entire architecture.

The optimization method in NAS determines how the search space is explored and the optimal solution obtained. The most commonly used methods include reinforcement learning based algorithms [51, 6], evolutionary based algorithms [54, 55], Bayesian optimization [7], and gradient-based methods [56, 10]. Finally, the evaluation strategy measures the performance of the candidate architecture found with the optimization method. The simplest approach is fully training the network and using the validation performance as a metric [8], but this demands high computational resources. Less expensive proxy-metrics have been proposed such as partial training [57], training on a subset of the data [58], or using images with lower resolution [59]. NAS-specific evaluation methods have also been proposed to reduce convergence time such as network morphism [60], weight-sharing [5], and hypernetworks [61].

2.4 Multiobjective NAS

Although it is essential to design networks that have a high accuracy, in various real-world applications it is also important to have models that optimize other objectives such as model size, inference time or energy consumption. Multiobjective NAS is an approach where these additional objectives can be considered when automatically designing an architecture. In [62] Kim et al. employed the classic NSGA-II algorithm to design CNNs for image classification that maximize accuracy and minimize inference speed. Their method optimized the type of layer and number of outputs per layer in the architecture. Dong et al. [63] proposed a surrogate model-based approach that adapts the progressive search strategy presented by Liu et. al [13] for multiobjective NAS. The proposed method optimizes the configuration of the basic cells in a CondenseNet based network. Furthermore, a Recurrent Neural Network is chosen as the surrogate function. Elsken et al. [64] presented a Lamarckian evolutionary algorithm that generates child architectures warm started with the predictive performance of their parents. It also uses the information provided by the “cheap” to evaluate objective functions, such as model size, to select child architectures that will fill the gaps on the Pareto Front. In [65] Lu et al. proposed a NSGA-II based algorithm to handle multiobjective NAS problems and employed a Bayesian Network to exploit promising solutions discovered in the search.

2.5 NAS for Medical Image Segmentation

NAS has been primarily applied to search for CNN architectures for image classification and for recurrent neural networks (RNN) for language modeling. Directly applying previously developed NAS methods for medical image segmentation is not feasible because the search space differs significantly and the approaches can be computationally intractable as the architecture search must operate on high resolution imagery. There has been limited work on using NAS for medical image segmentation. In [66], Mortazi and Bagci proposed a policy gradient

reinforcement learning based method to search for the kernel size and number of feature channels on the convolutional layers of a 2D densely connected encoder-decoder baseline. In [67], Weng et al. presented three types of primitive operation sets to construct the encoder and decoder cells for a 2D U-Net backbone network. The cell configurations are updated using the DARTS [10] differential search strategy. Zhu et al. [68] proposed a differentiable NAS to select between 2D, 3D and pseudo-3D convolutions for the fixed encoder and decoder cells of a FCN baseline architecture. Similarly, Kim et al. [69] presented a differentiable NAS framework to optimize the encoder, decoder, reduction and expansion cells of a 3D U-Net template architecture. Finally, in [70] the reinforcement learning based algorithm proposed by ENAS [5] is applied to find the optimal macro configuration of a U-Net baseline architecture, in which the input patch size, pooling operations, activation function, convolutional dilation rate and skip connections points are included as hyperparameters.

2.6 Current Research Challenges

The proposed research aim to address the following challenges: 1) the difficulty of designing both accurate and efficient deep neural networks for a specific problem; 2) the long search times and high computational resources required to automatically optimize the architecture of a deep neural network; and 3) the complexity of automatically searching for CNN architectures in the task of 3D medical image segmentation.

First, deep neural networks have increased in size and complexity to achieve an accurate performance. Hence designing an architecture for a specific dataset requires extensive expertise, time, and computational resources. Current models have many design parameters (i.e., number and type of layers, activation functions, number of filters, and size of kernels) that determine the accuracy of the model. Setting the correct values for these parameters is a non-trivial task because of the massive search space and exponential number of viable architectures. Given that it is infeasible to test all possible configurations, the design process has relied mostly on manual optimization. This can result in neural networks that are over-parametrized and highly inefficient at exploiting their full potential. As the application of deep learning is moving towards mobile and embedded platforms, it has become increasingly important to not only design accurate but also efficient models. Therefore, designing a neural network has to be modeled as multiobjective optimization problem, where the performance as well as the efficiency need to be optimized simultaneously.

Automatically designing neural networks through the application of optimization algorithms is a growing research area that aims to help people find good performing architectures. However, methods for neural architecture

search are usually computationally expensive and time-consuming, requiring multiple GPUs and thousands of GPU hours to converge to a good solution [8, 6]. Therefore, it could be unfeasible to replicate the proposed methods on other problems that require high memory usage, such as image segmentation or high-resolution 3D image processing, or in settings with limited computational resources. Methods that satisfactorily balance speed and configuration quality are needed. Furthermore, as most techniques developed to speed the convergence of NAS methods have been designed and applied for single-objective optimization, it is necessary to design techniques for the more complicated and time-consuming problem of multiobjective NAS.

Finally, the development of highly complex and diverse FCN's architectures has led to a dramatic improvement in the automated segmentation of medical images. However, it remains extremely difficult to adapt a FCN to a new segmentation task. Medical images have specific characteristics compared to other types of imagery that need be considered when adapting an architecture. First, medical images are significantly affected by the acquisition protocol and specific imaging modality. For example, common problems in MRI include partial volume effect, noise caused by the sensors and electronic system, presence of artifacts (gradient, motion, and wrap around) and intensity inhomogeneity. Moreover, because the process of annotating medical images is expensive and time-consuming, there is usually a limited number of training images with ground truth. Therefore, FCNs are prone to overfitting and having a poor performance on unseen datasets. Finally, medical image data is often multi-dimensional and multi-modal requiring the methods to consider the entire volume to perform the segmentation in 3D. Incorporating volumetric information into FCNs substantially increases the number of parameters required, the computational cost, and the need for GPU memory. Hence, to develop deep neural architecture search methods for medical applications, it is important to consider the large variability across image datasets, the limited number of images available for training, and the volumetric information contained in the medical data.

Chapter 3: AdaResU-Net-Multiobjective Adaptive Convolutional Neural Network for Medical Image Segmentation

3.1 Note to Reader

This chapter has been previously published in *Neurocomputing*, 2019 (see reference [71]), and has been reproduced with permission from Elsevier Publishing. The copyright permission for reuse can be found in Appendix A.

3.2 Introduction

In this chapter, a new convolutional neural network for 2D magnetic resonance image (MRI) segmentation that is able to automatically adapt to new datasets while minimizing the network size is presented. The proposed architecture, called AdaResU-Net, has a fixed basic structure that combines the favorable symmetry and skip connections of the well-known U-net [22] with a residual framework [72] as the building block to improve information propagation and speed the training process. The AdaResU-Net has an unfixed set of hyperparameters that are automatically selected through a learning framework that uses a multiobjective evolutionary based algorithm (MEA algorithm). The MEA algorithm evolves models subject to two objective functions: segmentation accuracy and number of trainable parameters. The advantage of formulating the AdaResU-Net in this manner is that it reduces the time necessary to adapt the model to new datasets by changing the unfixed hyperparameters. To evaluate the performance of the proposed method, experiments were conducted on prostate and heart MRI from publically available datasets. Results show that the AdaResU-Net achieves better segmentation performance with less than 30% the number of trainable parameters than the U-Net. Additionally, the MEA algorithm generated configurations that are smaller and equally or better than configurations generated with a Bayesian optimization approach.

3.3 Methods

The AdaResU-Net model is composed of a fixed architecture and a learning framework that selects the set of hyperparameters that best fit a particular dataset. The hyperparameter search space is confined to the learning rate, number of filters, kernel size, dropout probability, and activation function. The proposed MEA learning framework uses a MOEA/D based algorithm with PBI approach to evolve a population of candidate AdaResU-Net networks

subject to two objective functions: segmentation accuracy and model’s size. After a specified number of generations, the approximate Pareto Front is obtained. The Pareto optimal solutions that compose this front correspond to the AdaResU-Net architectures (with different hyperparameters) that have achieved the best values in the objective functions. Since the main objective is to obtain a good segmentation, the architecture that maximizes the segmentation accuracy is selected as optimal and used for testing. In the following sub-sections, we describe the AdaResU-Net fixed architecture, how the hyperparameters are represented in the genotype for the evolution process, and the MEA learning framework that adapts the architecture to the specific dataset.

3.3.1 AdaResU-Net Architecture

The proposed fixed AdaResU-Net architecture is shown in Figure 4, which is based on the ResU-Net architecture presented in our previous work [73]. Similar to the U-net, the AdaResU-Net consists of a down-sampling path on the left and an up-sampling path on the right. However, the basic building blocks of the AdaResU-Net are the residual learning frameworks as presented in [72], each one having a total of 3 padded convolutional layers. The residual blocks in the contracting path are followed by a max-pooling operation with stride 2 that progressively decreases the size of the feature map. The expansive path uses an up-sampling and convolutional layer to increase the size of the feature map until it reaches the size of the original input. To improve the quality and detail of the segmented image, long residual connections are placed between blocks of the same size from the up-sampling and down-sampling path. The last convolutional layer in the network has a fixed filter of size 1x1 and a sigmoid activation function.

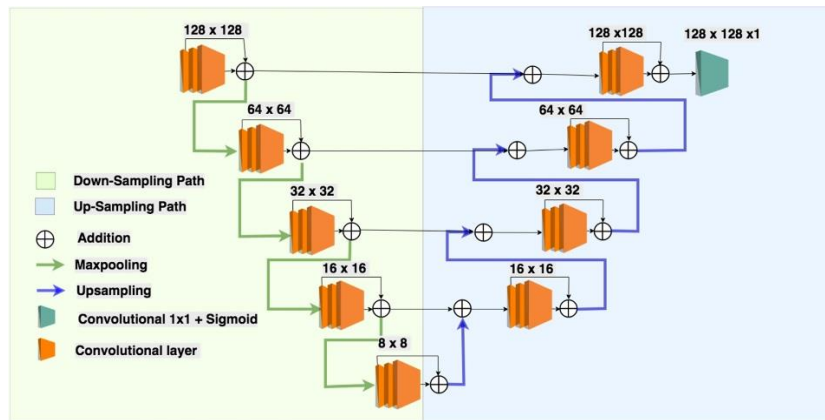


Figure 4 Fixed AdaResU-Net architecture. Each box represents a dropout or convolutional layer. The arrows represent the max pooling, up sampling and residual operations. The size of the input in each convolutional block is shown on top of each block.

The advantages of using the residual block as the basic unit in the architecture is that it favors information flow and facilitates parameter optimization by allowing the gradient to easily back propagate through the connections.

Moreover, the residual learning framework has no additional parameters, which means that all the improvements are obtained without adding computational complexity to the model.

Differently from our previously designed ResU-Net architecture [73], the proposed AdaResU-Net architecture includes dropout layers before the residual blocks to prevent the co-adaptation of feature detectors [74]. Then, the size of the features maps is reduced to decrease computational cost. Finally, the hyperparameters described in Section 3.3.2 have been left unset for the MEA learning framework to select automatically based on the specific dataset.

3.3.2 Hyperparameter Representation

In the proposed MEA learning framework, we focus on fine-tuning the hyperparameters associated with the model and only consider the learning rate from the hyperparameters associated with the gradient based-optimization. These were selected because our primary interest is to find the optimal architecture for the segmentation of a particular set of medical images. The learning rate has been included because it is considered the most important hyperparameter in the training process [75] and it usually needs to be tuned when the input images or architecture configuration changes. Having constructed the fixed AdaResU-Net architecture by combining the successful U-Net structure with the residual learning framework as the basic unit, the rest of the unset hyperparameters make up the search space. Hence, the hyperparameter space consists of the learning rate, dropout probability, activation function, number of filters, and the kernel size for the three convolutional layers that make up a residual block.

The genotype is represented by an array of size 7 where every gene encodes each of the hyperparameters mentioned above. The learning rate and dropout probability are continuous variables; however, we perform a log-space search for the learning rate and discretize the dropout probability by using only multiples of 0.05. In this way, we foment the algorithm to make a broader search in the hyperparameter's range by choosing values with different magnitudes. For the dropout probability, the same probability will be used in all layers of the network. The activation functions are encoded as integer numbers and the same activation is applied to all convolutional layers. For the number of filters, the architecture follows a rule: the number of filters are doubled in each stage of the down-sampling path and halved in each stage of the up-sampling path. Therefore, in the genotype, the number of filters in the whole architecture is represented only by the number of filters in the first residual block and the number of filters in the other blocks are computed according to the stated rule. For example, if the number of filters in the genotype is 32, the actual

number of filters in the AdaResU-Net architecture are [32, 64, 128, 256, 512] in the down-sampling blocks and [256, 128, 64, 32] in the up-sampling blocks.

As described in Section 3.3.1, the AdaResU-Net has 9 residual frameworks with 3 convolutional layers each. The kernel size hyperparameter has 3 components in the genotype that refer to the kernel size of the 3 convolutional layers in the residual framework. Hence, the kernel size of the convolutional layers within a residual framework can be different but the block remains the same throughout the network. An example of a genotype and the corresponding decoded AdaResU-Net architecture is shown in Figure 5.

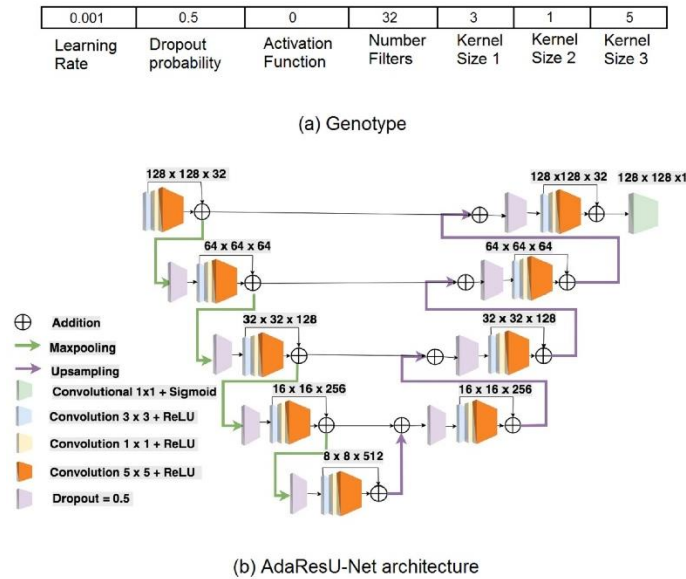


Figure 5 Example of a genotype decoded into an AdaResU-Net architecture. a) The genotype, where each gene refers to a hyperparameter. b) The resultant AdaResU-Net network after using the genotype from c). Different colors in the convolutional box represent a different kernel size

Adopting these rules for the number of filters and kernel size when generating the architecture's configuration reduces the search space and ensures the feasibility of all genotypes and architectures trained. This is necessary if we want to reduce the time it takes the MEA learning framework to approximate the Pareto Front. Moreover, state-of-the-art architectures in medical image segmentation use similar rules for specifying the number of filters per layer and kernel size [22, 41, 49].

3.3.3 Learning Framework

The proposed MEA learning framework aims to obtain the AdaResU-Net architecture configuration that maximizes the segmentation accuracy while minimizing the model's size. The segmentation accuracy is measured through the Dice similarity coefficient (DSC) defined as:

$$DSC = \frac{2 \sum_1^K \hat{y}_i(\theta) y_i}{\sum_1^K \hat{y}_i(\theta) + \sum_1^K y_i} \quad (1)$$

where y_i and $\hat{y}_i(\theta)$ are the pixels values in the ground truth and predicted segmentation, respectively, K is the total number of pixels in the image and θ represents the trainable parameters in the neural network. $\hat{y}_i(\theta)$ considers θ because the predicted pixel values depend on the learned weights of the CNN. The Dice coefficient is used as it is able to deal with the imbalance between background and foreground pixels usually present when segmenting medical images. Thus, maximizing the segmentation accuracy is equivalent to minimizing I -DSC. We call I -DSC the Dice loss function.

For the second objective of minimizing the model's size, we measure the number of trainable parameters in the network. The optimization model is formulated as follows:

$$\min_x FV(x) = \left(f_1(x) = 1 - \frac{2 \sum_1^K \hat{y}_i(\theta) y_i}{\sum_1^K \hat{y}_i(\theta) + \sum_1^K y_i}, f_2(x) = |\theta| \right)^T$$

subject to $x \in \Omega$ (2)

$|\cdot|$ represents the cardinality operator, and Ω the hyperparameter search space. The decision variable x is the genotype (or hyperparameter values) for the AdaResU-Net fixed architecture.

Figure 6 shows the overview of the proposed MEA learning framework. First, the method is initialized by generating an initial population of candidate AdaResU-Net architectures of size N . This is done by randomly selecting the genotypes from the hyperparameter space and decoding them into network configurations. The candidates are trained using the backpropagation algorithm and the ADAM optimizer [76] with the learning rate value from the genotype. Then, the MEA algorithm is applied to evolve the initial population for a specified number of generations. In each generation, the MEA generates the genotype of new individuals from the current optimal solutions and decodes them into candidate AdaResU-Net that are trained through backpropagation. Each individual is evaluated by the two objective functions presented in Equation (2) and the set of optimal solutions is updated through the PBI approach. At termination, the MEA algorithm will provide the solutions that approximate the Pareto Front, or in other words, the network configurations that give the best tradeoffs between the two objective functions. The solution that has the smaller Dice loss is selected as optimal and used to construct the AdaResU-Net that is evaluated on the test set. It is important to mention that even though we did not select the AdaResU-Net with the minimum number of trainable parameters, considering the model's size as an objective function in the algorithm forces the MEA algorithm to generate and select solutions that minimize the number of trainable parameters.

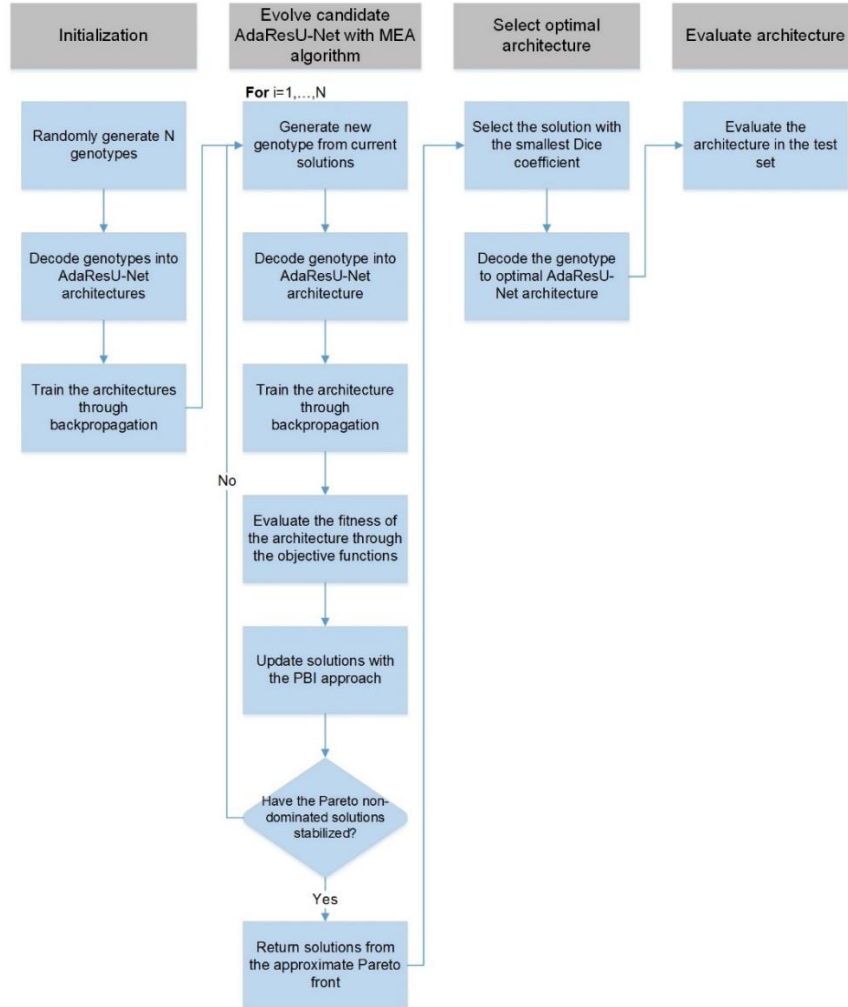


Figure 6 Overview of the proposed MEA learning framework for the AdaResU-Net model.

The proposed MEA algorithm is based on the framework of MOEA/D [77] and is presented in Algorithm 1. A PBI decomposition approach is selected because it provides uniformly distributed solutions on the boundary regions of the Pareto Front when solving a complex problem and the number of weight vectors is not large [78]. This is important in our case because training a neural network is a highly non-convex optimization problem and the number of weight vectors used by the MEA learning framework is small to reduce computational cost. However, using the PBI approach requires the selection of a penalty factor to balance the convergence and diversity of the solutions. For the proposed AdaResU-Net neural network, we applied the MEA algorithm for 10 generations with penalty factor values of 0.5, 1, 5, 8 and 10 on the prostate dataset. The factor of 5 was selected because it provided a diverse set of solutions that had better performance in the objective functions. This penalty value has also been commonly used in other studies and provided good results [79, 80].

In many applications, the value of the objective functions can have different magnitudes. In these cases, a normalization method is necessary to approximate the objective function's value to a same scale and eliminate bias when selecting the non-dominated points. This normalization is very important in the proposed algorithm given that the Dice loss returns values between [0,1] while the number of trainable parameters in a network can easily grow to millions. Let M be the number of objective functions in the MOP, in this case 2, $f_m(x_g^i)$ is the value of the objective function m with individual i in generation g and x_g^i genotype. The normalized objective function of individual i can be defined as follows:

$$FV^i = \left[\frac{f_1(x_g^i) - z_1^{min}}{z_1^{max} - z_1^{min}}, \dots, \frac{f_M(x_g^i) - z_M^{min}}{z_M^{max} - z_M^{min}} \right] \quad (3)$$

where

$$\begin{aligned} z_m^{max} &= \max \{f_m(x_g^1), \dots, f_m(x_g^N)\} \text{ for } m \in \{1, 2, \dots, M\} \\ z_m^{min} &= \min \{f_m(x_g^1), \dots, f_m(x_g^N)\} \text{ for } m \in \{1, 2, \dots, M\} \end{aligned} \quad (4)$$

z_m^{max} and z_m^{min} approximates the maximum and minimum values that the objective function m can take, respectively. As discussed above, the Dice loss function already returns a value between [0, 1] and therefore there is no need to apply the normalization strategy. However, when quantifying the number of trainable parameters in an architecture, it is necessary to normalize the objective function value. The z_2^{max} and z_2^{min} for the second objective function can be computed at the initialization of the algorithm by considering the biggest and smallest candidate AdaResU-Net architecture that can be constructed from the hyperparameter search space.

In general, the PBI approach drives the solutions towards the ideal point z^* in the criterion space in a search direction defined by the weight vectors λ . In the proposed algorithm, the ideal point $z^* = \{z_1^*, \dots, z_M^*\}$ is computed as follows:

$$z_m^* = \alpha [\min\{f_m(x_g^1), \dots, f_m(x_g^N)\}] \text{ for each } m \in \{1, 2, \dots, M\} \quad (5)$$

where $\alpha \in [0,1]$ is a rectification parameter. The rectification parameter is added to keep the value of the ideal point smaller than the current minimum objective function value. In this way, if a new solution is encountered that has a smaller objective function than the current solutions, it will not get penalized when computing the PBI objective function. The weight vectors are computed as described in [77].

Finally, since some of the hyperparameters encoded in the genotype are discrete or categorical variables, a uniform crossover and uniform mutation are selected as genetic operators for reproduction.

Algorithm 1: MEA algorithm for evolving AdaResU-Net architectures**Input:**

- A MOP with M objective functions and a feasible set.
- N : population size in each generation.
- $\lambda^1, \lambda^2, \dots, \lambda^N$: N uniform distributed weight vectors of size M , where $\lambda^i = \{\lambda_1^i, \dots, \lambda_M^i\}$.
- T : Neighborhood size.
- A termination criteria: Maximum number of generations G or maximum run time T .

Output:

- NDS : N AdaResU-Net configurations (non-dominated solutions) that are part of the Pareto front.

Step 1 Initialization

Step 1.1 Initialize list with non-dominated solutions $NDS = \emptyset$, run time as $t=0$ and generation as $g=1$.

Step 1.2 For each weight vector λ^i $i \in \{1, 2, \dots, N\}$ determine the T closest weight vectors $[\lambda^{i1}, \lambda^{i2}, \dots, \lambda^{iT}]$ using the Euclidean distance as measure. Define the neighborhood of λ_i as the T closest weight vectors $B(i) = \{i_1 \dots i_T\}$.

Step 1.3 Generate an initial population x_1^1, \dots, x_1^N by randomly generating the genotypes from the hyperparameter space and decoding them into AdaResU-Net configurations. Train the N models using the backpropagation algorithm and set the value of the objective functions as $FV_1^i = [f_1(x_1^i), \dots, f_M(x_1^i)]$ for all $i \in \{1, 2, \dots, N\}$. Save the solutions generated in list L .

Step 1.4 Initialize the ideal point as $z^* = \{z_1^*, \dots, z_M^*\}$ where $z_m^* = \alpha [\min\{f_m(x_1^1), \dots, f_m(x_1^N)\}]$ for all $m \in \{1, 2, \dots, M\}$ and $\alpha \in [0, 1]$.

Step 1.5 Initialize the normalization points as $z_m^{max} = \max\{f_m(x_1^1), \dots, f_m(x_1^N)\}$ and $z_m^{min} = \min\{f_m(x_1^1), \dots, f_m(x_1^N)\}$ for all $m \in \{1, 2, \dots, M\}$.

Step 2 Update:

While $g < G$ or $t < T$ **do:**

Step 2.1: Normalize the values of the objective functions of each individual by setting $FV^i = \left[\frac{f_1(x_g^i) - z_1^{min}}{z_1^{max} - z_1^{min}}, \dots, \frac{f_M(x_g^i) - z_M^{min}}{z_M^{max} - z_M^{min}} \right]$ for all $i \in \{1, 2, \dots, N\}$.

For $i = 1 \dots N$ **do:**

Step 2.2 Uniform Crossover: Randomly select two indexes k and l from $B(i)$ and generate a new solution x_g^y from x_g^k and x_g^l by selecting with probability p_c the genes from x_g^k and with $(1 - p_c)$ the genes from x_g^l .

Step 2.3 Uniform Mutation: Apply with probability p_m mutation to the solution x_g^y . The number of genes mutated is randomly chosen between one and the total number of genes.

Step 2.4 If solution x_g^y is in L repeat Step 2.2 and Step 2.3. If not add the solution x_g^y to L .

Step 2.5 Train the model: Train the candidate AdaResU-Net architecture by backpropagation using the decoded hyperparameters from the solution x_g^y . Set the value of the objective functions for this model as $FV_g^y = [f_1(x_g^y), \dots, f_M(x_g^y)]$ using the performance measures of the candidate AdaResU-Net.

Step 2.6 Normalize the value of the objective functions of the solution x_g^y by setting FV_g^y

$$= \left[\frac{f_1(x_g^y) - z_1^{min}}{z_1^{max} - z_1^{min}}, \dots, \frac{f_M(x_g^y) - z_M^{min}}{z_M^{max} - z_M^{min}} \right].$$

Step 2.7 Update of Neighboring Solutions: For each index $j \in B(i)$, if $d_1^y + \Theta d_2^y \leq d_1^j + \Theta d_2^j$ replace x_g^j with x_g^y and FV_g^j with FV_g^y , where $d_1^y = \frac{\|(FV_g^y - z^*)^T \lambda^j\|}{\|\lambda^j\|}$, $d_2^y = \frac{\|FV_g^y - (z^* - d_1 \frac{\lambda^j}{\|\lambda^j\|})\|}{\|\lambda^j\|}$, $d_1^j = \frac{\|(FV_g^j - z^*)^T \lambda^j\|}{\|\lambda^j\|}$ and $d_2^j = \frac{\|FV_g^j - (z^* - d_1 \frac{\lambda^j}{\|\lambda^j\|})\|}{\|\lambda^j\|}$.

Step 2.8 Update list of non-dominated solutions NDS: If no non-dominated point in NDS dominates FV_g^y , add FV_g^y to NDS. Eliminate all non-dominated points in NDS dominated by FV_g^y .

Step 2.9 Update the ideal point: If $\alpha[f_m(x_g^Y)] < z_m^*$ then set $z_m^* = \alpha[f_m(x_g^Y)]$ for all $m \in \{1, 2, \dots, M\}$.

end for

Step 2.10 Update the normalization points and termination criteria parameters: Set the normalization points as $z_m^{max} = \max \{f_m(x_g^1), \dots, f_m(x_g^N)\}$ for $m \in \{1, 2, \dots, M\}$ and $z_m^{min} = \min \{f_m(x_g^1), \dots, f_m(x_g^N)\}$ for $m \in \{1, 2, \dots, M\}$. Set $t = \text{elapsed time}$ and $g = g + 1$.

end for
return NDS

3.4 Experimental Setup

We evaluated the AdaResU-Net model on two publically available MRI datasets. The first task is the segmentation of the prostate in MRI from the PROMISE12 challenge [81]. The dataset consisted of 50 3D transversal T2-weighted images with their corresponding ground truth. The 3D images were collected from four different medical centers, having a different size, resolution, intensity range and number of slices per patient. The second dataset comprised of short axis cardiac MRI sequences available in York university [82] from the Hospital of Sick Children in Toronto. A total of 33 subjects were scanned and each scan consisted of 20 frames with 8 to 15 slices. All slices have a size of 256×256 but the pixel resolution and spacing between slices differ. On this dataset the task is to segment the endocardium. The steps to perform the experiments on both dataset are the same and are shown in Figure 7.

Although the datasets are composed of 3D images, this study focused on the segmentation of 2D images as the main goal is to determine whether the proposed AdaResU-Net and the MEA learning framework can result in architectures that improve segmentation performance while minimizing the model's size. These techniques can be used in applications that require 2D segmentation in specific planes to extract metrics or a structure's shape to aid in medical diagnosis. In our previous work [83, 84], the pubic bone needed to be segmented from 2D MRI to extract reference points for assessing the severity of pelvic organ prolapse in women. Similarly in [85], we identified the sacral curve from dynamic MRI to facilitate the assessment of gynecological conditions. As the proposed AdaResU-

Net and MEA learning framework are demonstrated to be successful in segmenting 2D images, future work will address the modification of the AdaResU-Net architecture for 3D image segmentation.

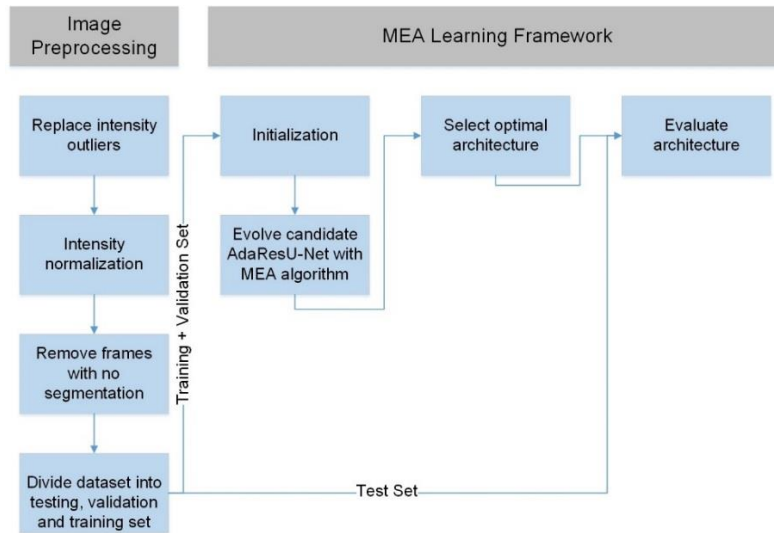


Figure 7 Experimental process for the prostate and endocardium segmentation.

3.4.1 Image Pre-processing

Some of the challenges when segmenting structures on MRI are the intensity inhomogeneity, noise interference, and distinct resolution [86]. Therefore, we preprocess the images to standardize their appearance in a slice-wise fashion by first removing intensity outliers. Intensities outside the 1.8 interquartile range are replaced with the nearest value inside the accepted range. Then, the pixel intensities are normalized to a mean 0 and standard deviation of 1. Finally, we solely keep the frames where the area of interest is present since our main objective is to test the segmentation ability of the models.

The datasets are divided into training, validation and testing set. The testing set is used exclusively for the evaluation step. On the prostate MRI dataset, 10 cases with a total of 165 2D images were used for testing, 37 cases with 562 2D images for training, and 3 cases with 51 2D images for validation. For the heart MRI dataset, 1744 images from 7 patients were used for testing, 2997 images from 23 patients for training, and 270 images from 3 patients for validation.

3.4.2 Learning Framework Application

3.4.2.1 Evolution of AdaResU-Net Candidates with the MEA Algorithm

The MEA learning framework is implemented as described in Section 3.3. The ranges for the hyperparameters in the search space are presented in Table 1. For the MEA algorithm, the generation size was selected

based on the convergence of the population. We stopped the algorithm when the set of Pareto optimal solutions did not changed for at least 10 generations. The AdaResU-Net population evolved for 50 generations on the prostate MRI dataset and for 30 generations on the heart MRI dataset. Information about the other parameters for the MEA algorithm, which were kept the same on both datasets, are presented in Table 2. Since our interest is in finding one good architecture, we select a small population size. This decreases the computational time but reduces the diversity of the solutions. Therefore, we chose a large neighborhood, in relation with the population size, to foment the algorithm to explore new areas of the search space as well as selecting a high mutation probability.

Table 1 Hyperparameter search space for generating new AdaResU-Net architecture configurations.

Hyperparameter	Range
Learning rate	$[1 \times 10^{-7}, 1 \times 10^{-1}]$
Dropout probability	$[0, 0.7]$
Activation Function	['ReLU', 'elu']
Number of Filters	[4,8,16, 32, 64]
Kernel size convolutional layer 1	[1x1 ,3x3 ,5x5]
Kernel size convolutional layer 2	[1x1 ,3x3 ,5x5]
Kernel size convolutional layer 3	[1x1 ,3x3 ,5x5]

Table 2 Parameters for the proposed MEA algorithm.

Parameter	Value
Population size	8
Neighborhood size	4
Rectification parameter	0.80
Crossover probability	0.50
Mutation probability	0.4
Penalty factor	5
Epochs trained per candidate	50

Another important parameter is the number of epochs each candidate AdaResU-Net is trained for in the MEA learning framework. In the present work, we tested between 30 and 80 training epochs, and selected 50 training epochs as it provided a diverse set of Pareto optimal solutions that were similar to the architectures obtained with 70 and 80 epochs. This approach may bias the learning framework into discovering fast learning architectures that have a good performance. However, the learning framework can be modified to include other training termination criteria such as the validation Dice loss to allow slower learning architectures to be included in the Pareto optimal solutions. Early termination was applied for all architectures that did not improve after 20 epochs of training and immediately stopped if exploding gradients were encountered during backpropagation. In the latter cases, a validation Dice loss of 1000 was assigned to prevent those configurations from being part of the non-dominated solutions.

The AdaResU-Net is trained through the backpropagation algorithm with the ADAM optimizer. The parameter values are set to beta 1: 0.9, beta 2: 0.999 and epsilon: 1×10^8 as suggested in [76]. Due to memory limitations, a batch size of 1 is selected. The weights in the candidates and optimal neural networks were initialized from a normal distribution with mean 0 and standard deviation of 0.05. Any value outside the 2 standard deviations was eliminated and re-sampled. The AdaResU-Net was implemented using Python 3.6 and Keras library [87]. The experiments were carried on an NVIDIA GeForce GTX 1080 GPU, 3.60-GHz CPU and 16-GB RAM

To enlarge the dataset, we used the strategy of data augmentation. We included images with a 90° of rotation range, 0.5 scaling range, 0.4 horizontal and vertical translation range and horizontal flip. The augmentation was done “on-the-fly” before each batch training to reduce memory usage.

3.4.2.2 Selection of the Optimal Architecture

The MEA algorithm returns 8 genotypes that make up the approximate Pareto Front. The genotype that has the smaller Dice loss is selected as optimal and decoded into the corresponding AdaResU-Net.

3.4.2.3 Evaluation and Comparison of the Model

To test the AdaResU-Net model, we trained the optimal architecture for 450 additional epochs on the training dataset and selected the weights that gave the higher score on the validation set. The metrics used to evaluate the performance are the Dice similarity coefficient, sensitivity, 95 percentile Hausdorff distance (95% HD), and mean surface distance (MSD). Let $Y = \{y_1, y_2, \dots, y_n\}$ and $\hat{Y}(\theta) = \{\hat{y}_1(\theta), \hat{y}_2(\theta), \dots, \hat{y}_n(\theta)\}$ represent the two sets that contain the pixels from the ground truth image and predicted segmentation, respectively. The Dice similarity coefficient is defined as:

$$Dice(Y, \hat{Y}(\theta)) = \frac{2|Y^1 \cap \hat{Y}^1(\theta)|}{|Y^1| + |\hat{Y}^1(\theta)|} \quad (6)$$

where \cap represents the intersection and $|\cdot|$ the cardinality of the set. Y^1 represents the pixels that are part of the region of interest (ROI) in the ground truth and $\hat{Y}^1(\theta)$ the pixels from the predicted segmentation that are part of the ROI.

The sensitivity (true positive rate) is defined as:

$$Sensitivity(Y, \hat{Y}(\theta)) = \frac{|Y^1 \cap \hat{Y}^1(\theta)|}{|\hat{Y}^1(\theta)|} \quad (7)$$

The Hausdorff distance is defined as:

$$HD(Y, \hat{Y}(\theta)) = \max(h(Y, \hat{Y}(\theta)), h(\hat{Y}(\theta), Y)) \quad (8)$$

where

$$h(Y, \hat{Y}(\theta)) = \max_{y \in Y} \{ \min_{\hat{y}(\theta) \in \hat{Y}(\theta)} \{ \|y, \hat{y}(\theta)\| \} \} \quad (9)$$

$\|\cdot\|$ is the Euclidean distance function. To make this measurement less susceptible to small outlying regions, the 95 percentile Hausdorff distance is used which reports the 95 quantile distance instead of the maximal distance in equation (11). The mean surface distance is defined as:

$$MSD(Y, \hat{Y}(\theta)) = \frac{1}{|Y| + |\hat{Y}(\theta)|} \left[\sum_{y \in Y} \min_{\hat{y}(\theta) \in \hat{Y}(\theta)} \{ \|y, \hat{y}(\theta)\| \} + \sum_{\hat{y}(\theta) \in \hat{Y}(\theta)} \min_{y \in Y} \{ \|\hat{y}(\theta), y\| \} \right] \quad (10)$$

The Dice similarity coefficient and sensitivity help evaluate the overlap between the predicted segmentation and the ground truth. On the other hand, the Hausdorff distance and mean surface distance compare the spatial distribution of the predicted segmentation and ground truth, not solely focusing on the points located at the intersection between both.

To evaluate the performance of the optimal AdaResU-Net architecture and the proposed MEA learning framework, we compared it with the U-Net architecture [22], ResU-Net architecture [73], and resulting AdaResU-Net architecture after using a Bayesian optimization (BO) approach [88] to select the optimal set of hyperparameter values. The U-Net has been selected since it is a well-known neural network for medical image segmentation that has served as a benchmark for evaluating numerous segmentation architectures and has achieved state-of-the-art results in various segmentation tasks [89, 90, 91]. To keep the comparison fair, the learning rate and drop out probability of the U-Net architecture are fine-tuned through a random search algorithm. The ResU-Net is similar to AdaResU-Net, but the kernel sizes and number of layers in each residual block are fixed. By comparing these architectures, we can assess the impact of adding residual connections on segmentation capability, as well as the impact of the proposed MEA learning framework on the performance of the AdaResU-Net.

The U-Net and ResU-Net are trained for a total of 500 epochs. The weights that gave the best performance on the validation set are used for the testing step. The 500 epochs were selected after verifying that the models reached a stable validation accuracy and no significant improvement could be attained with further training.

Finally, the AdaResU-Net with MEA hyperparameter optimization is compared with the AdaResU-Net with Bayesian hyperparameter optimization (BO). A BO method with a Gaussian process surrogate function and expected improvement (EI) acquisition function was used. The EI is selected since it has shown to perform better than other acquisition functions and does not require a tuning parameter [88]. The BO algorithm is implemented using the open-source Bayesian optimization toolbox GPyOpt [92]. As with the AdaResU-Net learning framework, the architectures

tested during the optimization process are trained for 50 epochs (early termination is also implemented) and the hyperparameter search space and parameter ranges are the same as displayed in Table 1. The optimal hyperparameter values are used to construct the optimal Bayesian AdaResU-Net architecture, which is trained for 450 additional epochs and the weights with the lowest validation loss are used for testing. A paired-samples t-tests with 0.05 alpha level is applied to determine if the evaluation metrics between the models were significantly different.

3.5 Results

3.5.1 Prostate Segmentation

For the prostate dataset segmentation, the AdaResU-Net candidates tested with the MEA algorithm and the obtained Pareto Front are presented in Figure 8 a). A total of 8 AdaResU-Net architectures constitute the non-dominated frontier, where the selected configuration has 4.6×10^6 trainable parameters and loss of 0.45 on the validation set. The graph shows the tradeoff between the two objective functions, the models in the Pareto front that have a larger number of trainable parameters have a smaller Dice loss. However, while considering the whole criterion space, a larger capacity does not necessary minimize the Dice loss. There are many cases in which architecture configurations with lesser trainable parameters outperform models with more parameters. The optimal hyperparameters for the AdaResU-Net architecture are shown in Table 3 under the name of AdaResU-Net MEA.

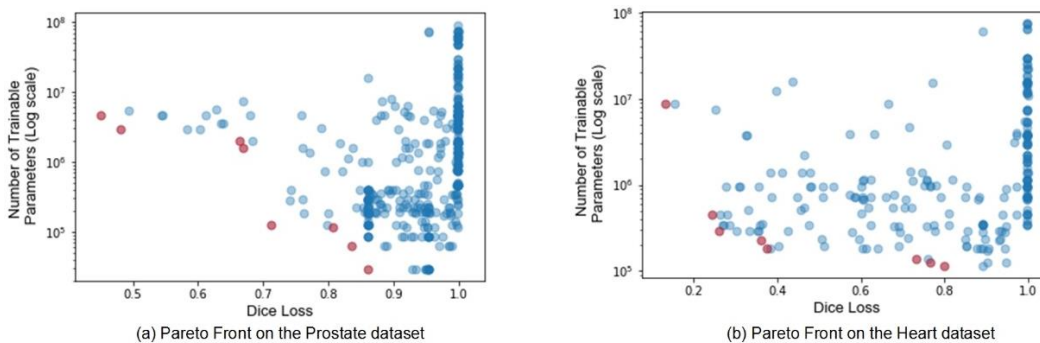


Figure 8 Tested AdaResU-Net architectures and their corresponding value in the criterion space. The Pareto front is formed by the points in color red a) Criterion space obtained for the prostate MRI dataset. b) Criterion space obtained for the heart MRI dataset.

In Table 4 and Figure 9 a), we present the evaluation metrics on the test set of the AdaResU-Net obtained with the proposed MEA learning framework (AdaResU-Net MEA), AdaResU-Net obtained with the Bayesian optimization (BO) approach (AdaResU-Net BO), ResU-Net, and U-Net. The p-values of the paired t-test between the four models is provided in Table 5. The p-values of the t-tests indicate that mean Dice coefficient, 95 percentile Hausdorff distance, and mean square distance are statistically equal for the AdaResU-Net MEA and AdaResU-Net

BO, while the latter has a higher mean sensitivity. In comparison with the other architectures, both AdaResU-Net models using the proposed MEA framework and Bayesian optimization achieve a significantly better mean in all evaluation metrics.

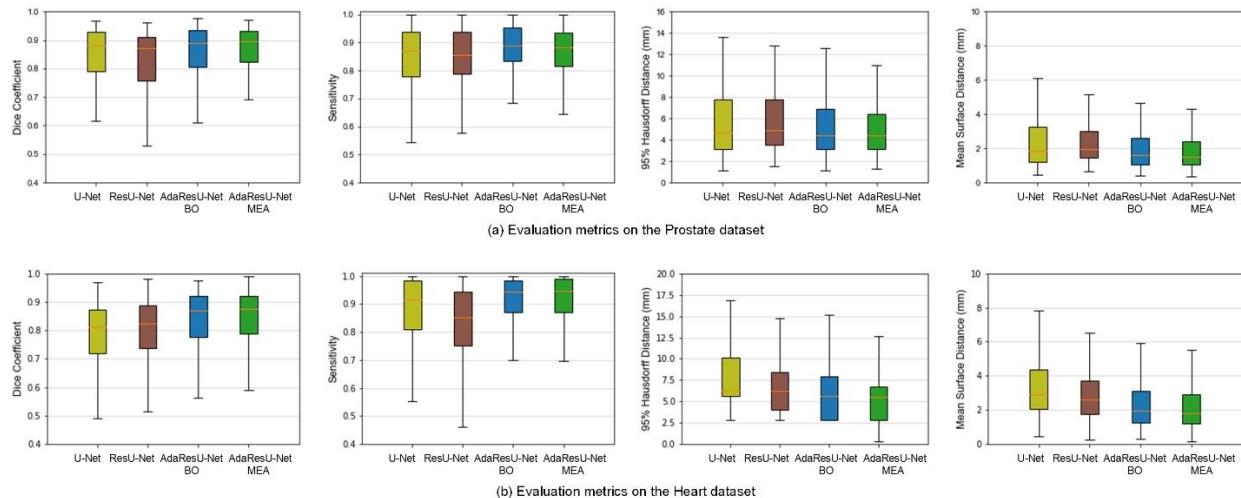


Figure 9 Evaluation metrics of the AdaResU-Net with the MEA learning framework (AdaResU-Net MEA: rightmost green boxplot), AdaResU-Net with Bayesian optimization (AdaResU-Net BO: middle-right blue boxplot), ResU-Net (middle-left brown boxplot) and U-Net (leftmost olive boxplot) on the test set. The metrics computed are Dice coefficient, sensitivity, 95 percentile Hausdorff distance and mean surface distance. a) Evaluation metrics for the prostate MRI dataset. b) Evaluation metrics for the heart MRI dataset.

Table 3 Optimal set of hyperparameters on the Prostate MRI dataset for the AdaResU-Net architecture with the proposed MEA learning framework (AdaResU-Net MEA) and with the Bayesian Optimization approach (AdaResU-Net BO).

Model	Dropout probability	Kernel 1	Kernel 2	Kernel 3	Activation Function	Number of Filters	Learning Rate
AdaResU-Net MEA	0.05	5	5	1	ReLU	16	2×10^{-5}
AdaResU-Net BO	0.35	5	1	1	ReLU	32	5×10^{-5}

Table 4 Evaluation metrics of the AdaResU-Net with the proposed MEA learning framework (*) and three other architectures on the Prostate MRI test set.

Model	Mean Dice Coefficient	Mean Sensitivity	95% HD (mm)	MSD (mm)	Number of Trainable Parameters
AdaResU-Net MEA	0.849 ± 0.136	0.859 ± 0.119	6.181 ± 5.728	2.271 ± 2.486	4.6×10^6
AdaResU-Net BO	0.848 ± 0.137	0.898 ± 0.105	6.382 ± 6.323	2.295 ± 2.516	8.1×10^6
ResU-Net	0.809 ± 0.149	0.828 ± 0.155	7.160 ± 7.116	2.949 ± 2.945	10.9×10^6
U-Net [22]	0.827 ± 0.146	0.840 ± 0.143	10.201 ± 15.435	3.244 ± 3.869	31.0×10^6

Table 5 P-values of paired t-test between two models on the Prostate MRI test set, * denotes a statistically significant difference with alpha value of 0.05 ($p < 0.05$) and ** a statistically significant difference with alpha value of 0.01 ($p < 0.01$).

Compared Models		Mean Dice Coefficient	Mean Sensitivity	95% HD	MSD
Model 1	Model 2				
AdaResU-Net MEA	AdaResU-Net BO	9.4×10^{-1}	$2.1 \times 10^{-9**}$	4.6×10^{-1}	7.4×10^{-1}
AdaResU-Net MEA	ResU-Net	$1.3 \times 10^{-6**}$	$1.4 \times 10^{-4**}$	$9.2 \times 10^{-3**}$	$2.3 \times 10^{-5**}$
AdaResU-Net MEA	U-Net	$3.9 \times 10^{-4**}$	$3.6 \times 10^{-3**}$	$6.4 \times 10^{-4**}$	$6.4 \times 10^{-4**}$
AdaResU-Net BO	ResU-Net	$1.5 \times 10^{-7**}$	$3.7 \times 10^{-14**}$	$1.6 \times 10^{-2*}$	$1.5 \times 10^{-5**}$
AdaResU-Net BO	U-Net	$1.9 \times 10^{-3**}$	$1.1 \times 10^{-11**}$	$1.2 \times 10^{-3**}$	$7.7 \times 10^{-4*}$
ResU-Net	U-Net	$1.4 \times 10^{-2*}$	1.8×10^{-1}	$1.1 \times 10^{-2*}$	3.2×10^{-1}

The optimal set of hyperparameters with the BO approach are displayed in Table 3 under the name AdaResU-Net BO. As it can be observed, the structural parameters such as the kernel sizes and activation function are similar to the ones selected with the proposed MEA learning framework. The main difference relies on the number of filters and dropout probability. The BO method converges to a bigger architecture and therefore must increase the dropout probability to prevent overfitting. Thus, the major improvement of our method in comparison with the BO approach is the reduction in the number of trainable parameters. The MEA learning framework was able to find an architecture 85% smaller than the U-Net, 57% smaller than the ResU-Net, and 44% smaller than the size of the AdaResU-Net BO, while providing a significantly better or similar segmentation accuracy. The decrease in the number of trainable parameters results in an important decrease in the training time, prediction time, and need of computational resources.

It can also be observed that the U-Net provides a better mean Dice Coefficient than the ResU-Net; however, it has the poorest performance in terms of the 95 percentile Hausdorff distance and mean surface distance. As shown in Figure 9 a), the U-Net has a high variability in the distance-based measures, showing a skew towards higher values. The images that cause this deviation have an overall acceptable definition of the prostate but there are also incorrect segmented areas outside the region of interest. This increases the spatial distance between the ground truth and predicted segmentation. The AdaResU-net and ResU-Net architectures do not have these troublesome segmentations, which we believe is due to the use of residual connections that allow a better transmission of information throughout the layers.

A visual illustration of the segmentation results for the prostate MRI are presented in Figure 10. Both AdaResU-Net models correctly identify and segment the complex shape of the prostate. Additionally, the contours obtained are continuous and smooth. The U-Net and ResU-Net, on the other hand, show under segmented areas in some images and do not capture small details of the prostate shape.

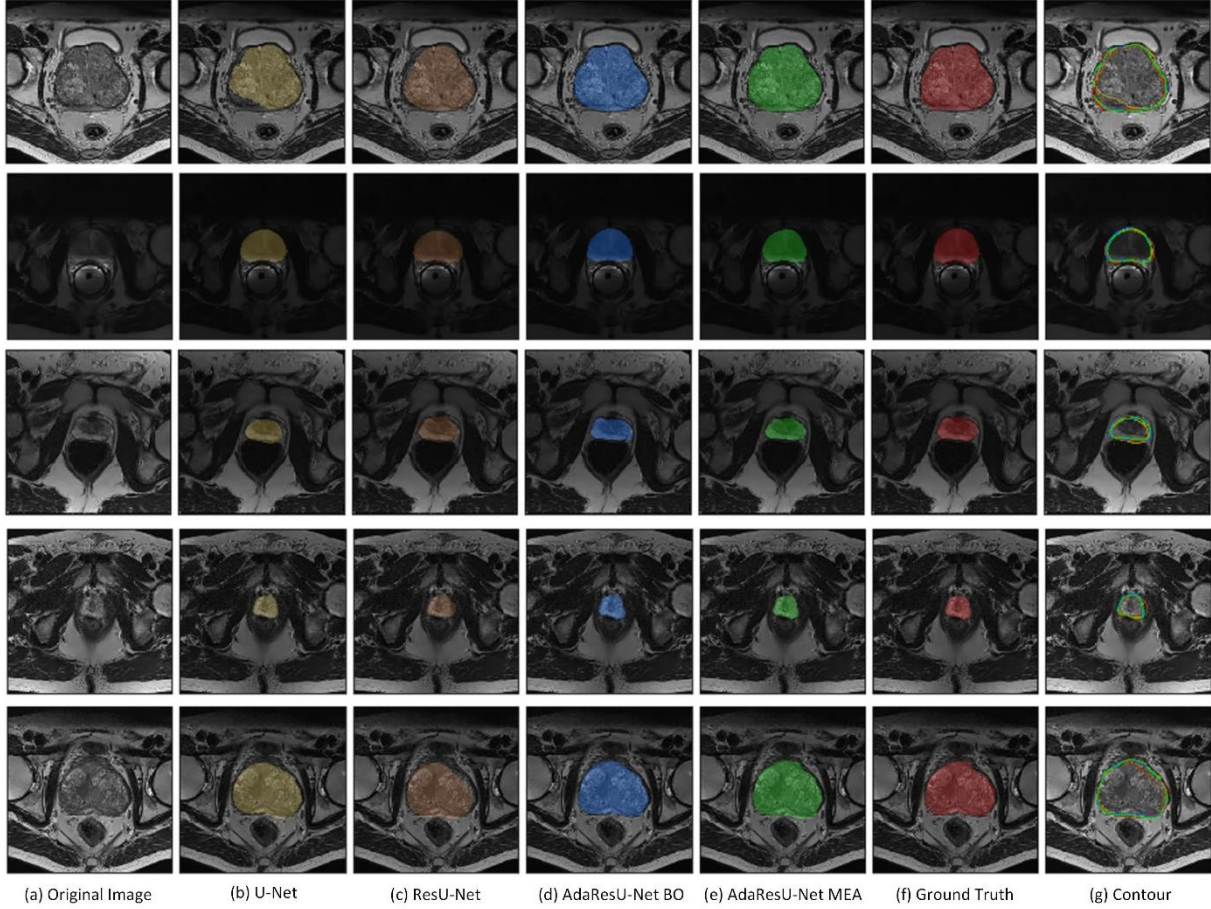


Figure 10 Segmentation results for different cases of the Prostate MRI test set. a) Input MR image b) Image segmented with the U-Net architecture c) Image segmented with the ResU-Net architecture d) Image segmented with the AdaResU-Net architecture with Bayesian optimization e) Image segmented with the AdaResU-Net architecture with MEA framework f) Ground truth g) Segmentation contours: ground truth (red), U-Net (olive), ResU-Net (brown), AdaResU-Net BO (blue) and AdaResU-Net MEA (green).

3.5.2 Endocardium Segmentation

The Pareto Front and criterion space for the heart dataset are shown in Figure. 8 b). Overall, the tested AdaResU-Net architectures achieved a smaller validation Dice loss in comparison with the prostate dataset. The selected optimal AdaResU-Net architecture has 8.8×10^6 trainable parameters with a 0.13 validation Dice loss. The optimal architecture for this dataset is presented in Table 6 for AdaResU-Net MEA and AdaResU-Net BO. Table 7 and Figure 9 b) present the quantitative evaluation of the AdaResU-Net MEA, AdaResU-Net BO, ResU-Net, and U-Net on the heart test set. Table 8 presents the p-values after applying a paired t-test between the 4 architectures. The difference between the four methods is statistically significant for all the metrics, except for the mean sensitivity in which the AdaResU-Net MEA and AdaResU-Net BO are statistically equal. The AdaResU-Net MEA has the best performance metrics, followed closely by the AdaResU-Net BO.

Table 6 Optimal set of hyperparameters on the Heart MRI dataset for the AdaResU-Net architecture with the proposed MEA learning framework (AdaResU-Net MEA) and with the Bayesian Optimization approach (AdaResU-Net BO).

Model	Dropout probability	Kernel 1	Kernel 2	Kernel 3	Activation Function	Number of Filters	Learning Rate
AdaResU-Net MEA	0.05	1	3	3	elu	32	1×10^{-5}
AdaResU-Net BO	0.45	1	5	3	ReLU	32	1×10^{-5}

Table 7 Evaluation metrics of the AdaResU-Net with the proposed MEA learning framework and three other architectures on the Heart MRI test set.

Model	Mean Dice Coefficient	Mean Sensitivity	95% HD (mm)	MSD (mm)	Number of Trainable Parameters
AdaResU-Net MEA	0.832 ± 0.139	0.903 ± 0.132	6.078 ± 5.858	2.361 ± 1.891	8.8×10^6
AdaResU-Net BO	0.820 ± 0.163	0.900 ± 0.142	6.950 ± 8.928	2.839 ± 3.845	15.7×10^6
ResU-Net	0.791 ± 0.144	0.824 ± 0.159	7.727 ± 7.550	3.076 ± 2.176	10.9×10^6
U-Net [22]	0.770 ± 0.152	0.869 ± 0.151	8.616 ± 7.289	3.542 ± 2.292	31.0×10^6

Table 8 P-values of paired t-test between two models on the Heart MRI test set, * denotes a statistically significant difference with alpha value of 0.05 ($p < 0.05$) and ** a statistically significant difference with alpha value of 0.01 ($p < 0.01$).

Compared Models		Mean Dice Coefficient	Mean Sensitivity	95% HD	MSD
Model 1	Model 2				
AdaResU-Net MEA	AdaResU-Net BO	$2.12 \times 10^{-11} **$	2.37×10^{-1}	$2.89 \times 10^{-4} **$	$3.05 \times 10^{-8} **$
AdaResU-Net MEA	ResU-Net	$5.87 \times 10^{-49} **$	$3.79 \times 10^{-113} **$	$6.76 \times 10^{-14} **$	$1.22 \times 10^{-41} **$
AdaResU-Net MEA	U-Net	$9.34 \times 10^{-133} **$	$1.78 \times 10^{-29} **$	$6.24 \times 10^{-35} **$	$4.01 \times 10^{-117} **$
AdaResU-Net BO	ResU-Net	$7.62 \times 10^{-20} **$	$3.04 \times 10^{-85} **$	$3.25 \times 10^{-3} **$	$9.10 \times 10^{-3} **$
AdaResU-Net BO	U-Net	$2.20 \times 10^{-62} **$	$8.61 \times 10^{-21} **$	$1.20 \times 10^{-10} **$	$1.16 \times 10^{-14} **$
ResU-Net	U-Net	$2.36 \times 10^{-13} **$	$2.62 \times 10^{-35} **$	$7.49 \times 10^{-4} **$	$2.05 \times 10^{-17} **$

In Figure 11 we show the visual results of the segmentation on the heart MRI dataset. The AdaResU-Net MEA captures the form and dimensions of the endocardium. Also, as in the segmentation of the prostate, the AdaResU-Net MEA provides smooth contours that correctly delineate the shape of the endocardium. The U-Net over segments some of the images and provide irregular boundaries in some regions. The ResU-Net has a better segmentation than the U-Net but still suffers from over segmentation on certain areas.

3.5.3 Effect of the Number of Training Images on the MEA Learning Framework

An important factor in the convergence of the MEA learning framework is the number of images available to train the candidate architectures. To test how this variable affected the performance of the algorithm, we tested the learning framework on the heart dataset with 500, 1000, 1500, 2000 and 2500 training images. These images were selected randomly from the 2997 images in the training set to increase the probability of obtaining a representative

sample. The Pareto frontier and criterion space obtained with the varying dataset size, including the results with the 2997 training images from the experiments presented in Section 3.5.2, are shown in Figure 12.

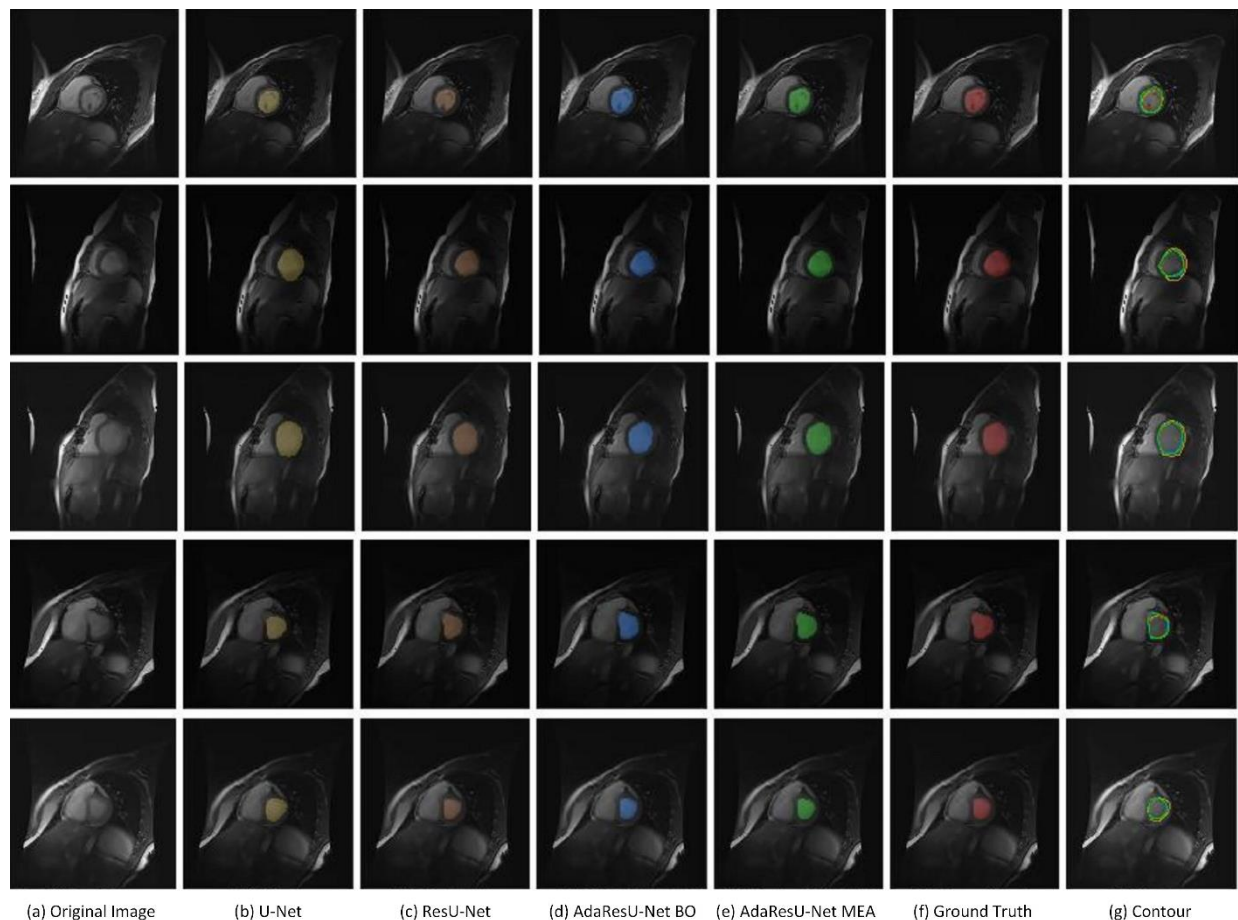


Figure 11 Segmentation results for different cases of the Heart MRI test set. a) Original MR image b) Image segmented with the U-Net architecture c) Image segmented with the ResU-Net architecture d) Image segmented with the AdaResU-Net architecture with Bayesian optimization e) Image segmented with the AdaResU-Net architecture with our MEA framework f) Ground truth g) Segmentation contours over the original image: ground truth (red), U-Net (olive), ResU-Net (brown), AdaResU-Net BO (blue) and AdaResU-Net with the proposed learning framework (green).

Figure 12 shows that the validation Dice loss of the optimal architectures decreases as the number of training images in the dataset increases. This behavior is expected because the candidate architectures are able to learn features from a larger variety of images and are trained for more iterations per epoch. Furthermore, it can be observed that as the dataset becomes larger, the size of the optimal architecture also increases. This is also explained by the additional training images that allow the model to learn significant correlations without overfitting the dataset. This factor also improves the performance, in terms of the validation Dice loss, of the optimal architectures from the bigger datasets.

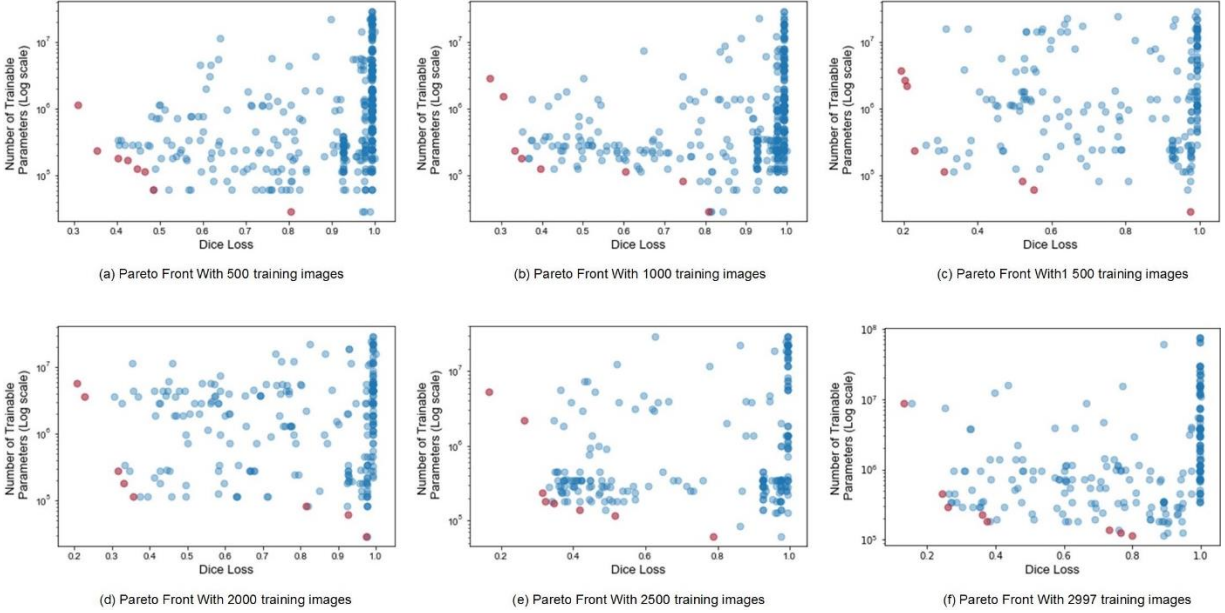


Figure 12 Pareto front for the heart dataset with varying number of training images. As more training images become available the MEA learning framework finds bigger architectures that have a smaller Dice loss. However, the set of hyperparameters still construct

Table 9 Optimal hyperparameter values of the AdaResU-Net architecture with varying dataset sizes.

Dataset Size	Dropout probability	Kernel 1	Kernel 2	Kernel 3	Activation Function	Number of Filters	Learning Rate	Number of Trainable Parameters
2997	0.05	1	3	3	elu	32	1×10^{-5}	8.8×10^6
2500	0	1	3	1	elu	32	5×10^{-5}	5.3×10^6
2000	0.15	1	5	5	elu	16	1×10^{-5}	5.6×10^6
1500	0.2	5	3	3	elu	16	1×10^{-5}	3.7×10^6
1000	0	5	1	3	elu	16	5×10^{-6}	2.8×10^6
500	0	5	1	5	elu	8	1×10^{-4}	1.1×10^6

Table 10 Performance metrics on the Heart test set of the optimal AdaResU-Net architectures with varying dataset size. The architectures with statistically significant higher mean are shown in bold.

Dataset Size	Mean Dice Coefficient	Mean Sensitivity	95% HD (mm)	MSD (mm)
2997	0.832 ± 0.139	0.903 ± 0.132	6.107 ± 5.890	2.372 ± 1.899
2500	0.825 ± 0.144	0.846 ± 0.154	5.947 ± 6.955	2.387 ± 2.068
2000	0.829 ± 0.128	0.903 ± 0.125	5.964 ± 4.385	2.425 ± 1.691
1500	0.821 ± 0.133	0.878 ± 0.133	6.116 ± 3.544	2.520 ± 1.694
1000	0.809 ± 0.151	0.894 ± 0.137	6.360 ± 3.098	2.655 ± 1.563
500	0.804 ± 0.132	0.880 ± 0.118	6.995 ± 4.738	2.872 ± 1.825

The optimal configurations of the AdaResU-Net architectures for the different dataset sizes are presented in Table 9. The set of selected hyperparameters share some similarities. First, the dropout probability remains small, which we believe is because the learning framework prefers to prevent overfitting by reducing the size of the neural

network than by increasing this probability. Then, the activation function remains unchanged and in most of the cases the learning rate has a similar magnitude. When focusing on the hyperparameters related to the structure (kernel sizes and number of filters), the models seem to change in clusters. Models obtained with 500 to 2000 images have kernels of size 5 and most have 16 filters on the first residual block. The models with 2500 and 2997 training images only include kernels of size 1 and 3 while increasing the number of filters of the first residual block to 32. These structural hyperparameters define the size of the network and, as discussed previously, seem to change to increase the capacity of the model as more training images become available.

The optimal AdaResU-Net architectures found with the different sample sizes and presented in Table 9 were trained with the whole training set, using the same optimization parameters described in Section 3.4.2.1, and evaluated on the test set. The performance metrics are displayed in Table 10. Applying a paired t-tests with a 0.05 significance show that the AdaResU-Net architecture obtained with the 2997 training images has a statistically higher mean Dice coefficient than the other models, followed closely by the resulting architectures with 2500 and 2000 training images. In relation to the 95 percentile Hausdorff distance and mean surface distance, the three models obtained with 2997, 2500 and 2000 training images are statistically smaller and therefore have a better spatial configuration that resembles the ground truth.

Based on the test metrics, it is concluded that the MEA learning framework works better if more training images are provided. This is also an expected outcome since more information about the complexity of the dataset enables the learning framework to determine the most appropriate size and hyperparameters for the architecture. However, as observed in the datasets with 2500 and 2000 training images, the MEA learning framework does not require all of the training images to reach to a good solution. Thus, the framework can provide a configuration that does well in the test set by using a representative subset of a big dataset. It is also important to mention that even though the architectures found with 500 and 1000 training images have a slight decrease in the mean Dice coefficient and increase in the distance based measures in comparison with the other bigger architectures, they still provide better segmentation results than the U-Net and ResU-Net with a considerably reduced number of trainable parameters.

3.5.4 Computation Time Analysis

In the proposed model, training the candidate AdaResU-Nets during the evolution process makes up most of the computational time. The computation time, in seconds, and number of candidate AdaResU-Net architectures tested by the MEA learning framework to approximate the Pareto Front on the two datasets is shown in Table 11.

Furthermore, the time and number of architectures tested with the Bayesian optimization approach are also presented. On the prostate and heart dataset, the BO method converges to an optimal solution in a lesser number of iterations and time. Considering that presented values can differ from one problem to another because both methods rely on stochastic values that affect the speed and time of convergence, the main reason of this difference is the problem each method tries to tackle. The proposed MEA learning framework aims to optimize two objective functions simultaneously. Thus, it requires additional iterations to find and allow the set of optimal Pareto solutions to stabilize for 10 generations. In the Bayesian optimization approach, only one objective function is defined, minimize the validation dice loss, and thus the convergence of only one optimal solution is needed. Therefore, adding an objective function to the hyperparameter optimization problem adds computational time during training. However, as it can be seen in the experimental results, the MEA algorithm provides equally or more accurate architectures that are smaller, faster to train and segment, and require less memory.

Table 11 Time, in seconds, it takes the AdaResU-Net to adapt to the prostate MRI dataset and heart MRI dataset through the proposed MEA learning framework and using the Gaussian Process Bayesian optimization.

Adaptation Method	Metric	Prostate MRI dataset	Heart MRI dataset
MEA Learning Framework	Time	221,588.88	594,397.40
	Architectures tested	400	240
GP Bayesian Optimization	Time	129,765.90	173,659.35
	Architectures tested	202	59

Table 12 Computation time, in seconds, the AdaResU-Net MEA, AdaResU-Net BO, ResU-Net and U-Net take to train and segment an image.

Dataset	Task	U-Net	ResU-Net	AdaResU-Net BO	AdaResU-Net MEA
Prostate	Training	20,354.01	11,705.91	11,003.02	8,778.44
	Segmentation	1.21×10^{-2}	0.79×10^{-2}	0.71×10^{-2}	0.57×10^{-2}
Cardiac	Training	108,084.65	63,326.68	83,081.687	57,515.25
	Segmentation	1.09×10^{-2}	0.68×10^{-2}	0.97×10^{-2}	0.66×10^{-2}

In comparison with more traditional hyperparameter optimization approaches, the time our learning framework takes to converge accounts for the time a designer takes to manually or semi-automatically fit an architecture to a dataset. There are no formal statistics about the time a designer spends in fine-tuning a neural network, but it is known to require extensive time, knowledge and intuition. Our MEA learning framework has three major advantages: 1. The search process is fully automated (only the ranges for the hyperparameters must be set in the initialization), 2. The algorithm generates hyperparameter combinations that might not be usually tested by designers but after implementing have better performance than manually-tuned architectures, 3. The algorithm considers the

architecture size as another objective function, which decreases training and prediction time. Therefore, the time invested in adapting the AdaResU-Net reduces the time spent in subsequent steps.

In Table 12 we present the computation time, in seconds, the AdaResU-Net MEA and the other three competing architectures spent in training (500 epochs) and the average prediction time per one image. The AdaResU-Net MEA achieves a reduction in training time between 10% and 56.9% and a reduction in prediction time ranging from 3% to 52.9% in comparison with the competing architectures on the two datasets.

3.5.5 Extension of the AdaResU-Net Model to 3D Segmentation

As previously mentioned, the AdaResU-Net model has been developed to perform 2D segmentations. However, to test its applicability to 3D image segmentations, the optimal AdaResU-Net architectures defined in subsection 3.5.1 and 3.5.2 were applied to the 3D test images from the prostate and heart dataset. The AdaResU-Net model performed the segmentation in a slice-wise manner and the resulting 2D segmentations were stacked in the z-axis to obtain the 3D volume. The mean Dice coefficient is computed to evaluate the 3D segmentation, which is defined as in subsection 3.4.2.3 with the exception that it measures the similarity between the predicted volume and ground truth volume. The quantitative results of our proposed model and state-of-the-art models published using the same prostate and heart datasets are presented in Table 13 and Table 14 respectively.

Table 13 Mean Dice coefficient of the proposed model in the 3D test images from the prostate dataset and competing state-of-the-art models.

Model	Type	Mean Dice Coefficient
AdaResU-Net	Automatic: 2D CNN	0.877±0.031
Brosch et al. [93]	Automatic: Model-based	0.905
Lequan et al. [41]	Automatic: 3D CNN	0.894
Tang et al. [94]	Semi-Automatic: Graph Cuts	0.893 ± 0.031
Sciolla et al. [95]	Automatic: 3D/2D CNN	0.893
Jia et al. [96]	Automatic: 3D CNN	0.889

Table 14 Mean Dice coefficient of the proposed model in the 3D test images from the cardiac dataset and competing state-of-the-art models.

Model	Type	Mean Dice Coefficient
AdaResU-Net	Automatic: 2D CNN	0.862±0.076
Barba-J et al. [97]	Semi-Automatic: Active Contour	0.912±0.026
Ehrhardt et al. [98]	Semi-Automatic: Atlas-based	0.86
Santiago et al. [99]	Automatic: Active Shape	0.794±0.081

In the prostate dataset, the AdaResU-Net has the lowest performance in comparison with the latest published works. However, this result is expected given that all the other models either employ a 3D CNN or, in the case of [94], use a semi-automatic method that requires a manual intervention to specify the prostate boundary points. Therefore,

we believe the AdaResU-Net has promising results considering that it does not fully exploit volumetric information or relies on any manual tuning. For the cardiac dataset, the AdaResU-Net has the second best result. The leading model, proposed by Barba-J et al [97], utilizes hand-crafted features especially catered for the endocardium segmentation, applies an elliptical shape constraint that reduces the model’s generalization capability and requires the manual identification of limits for base and apex of the left ventricle. Thus, given the automatic adaptability of the AdaResU-Net, our model is more flexible and amenable with the user while having a competitive performance.

3.6 Discussion

The experiments show that the AdaResU-Net architecture with the proposed MEA learning framework (AdaResU-Net MEA) is able to successfully adapt to the prostate and heart datasets. Moreover, on both sets of images the AdaResU-Net MEA achieves a high segmentation accuracy with an adequate spatial definition that outperforms the U-Net and ResU-Net in terms of the mean Dice coefficient, 95 percentile Hausdorff distance, and mean surface distance. Meanwhile, the U-Net and ResU-Net perform differently on the two datasets, having a substantially lower mean Dice coefficient on the heart dataset. The variability in the results demonstrates that tuning only the learning rate and dropout may not be sufficient to adapt an architecture to distinct datasets. Moreover, the set of hyperparameters we have selected for our learning framework to tune allows our model to successfully fit new datasets. The results of AdaResU-Net with MEA learning framework and the AdaResU-Net with BO approach suggest that our algorithm results in similar or better performance compared to the well-known Gaussian Process Bayesian hyperparameter optimization method, while providing considerably smaller architectures.

In both datasets, the ResU-Net has a statistically smaller 95 percentile Hausdorff distance and mean surface distance than the U-Net. These measures help assess how boundaries in the segmentation output resemble the boundaries of the ground truth. Therefore, we can conclude that the addition of residual connections helps the AdaResU-Net provide a better holistic shape and spatial distribution of the segmented region. However, the U-Net has a statistically better mean sensitivity on both datasets and mean Dice coefficient on the prostate dataset than the ResU-Net. This means that merely adding residual connections are not enough for improving the overlap measures (Dice coefficient and sensitivity) in the testing images. Implementing the proposed MEA learning framework to tune the hyperparameters in the AdaResU-Net architectures is an important factor in significantly increasing the mean Dice coefficient and sensitivity, as well as decreasing the Hausdorff distance and mean surface distance on both datasets.

The hyperparameters selected for the AdaResU-Net on the two datasets differ in most values, which means that the flexibility provided by our model is fully exploited by the proposed MEA algorithm. The architecture for the prostate dataset has lesser number of filters per layer than the architecture for the heart dataset, but the kernel size is bigger. Since the number of trainable images on the prostate dataset is small, having too many filters per layer will make the architecture prone to overfitting. This can be observed in Figure 8 a), where architectures with more than 5.5×10^6 trainable parameters have a Dice loss higher than 0.85. However, we believe that to compensate the reduction in depth of the layers, the algorithm selects a bigger kernel size that increases the capacity of a convolutional layer to learn local relations. On the heart dataset, the architecture is deeper and has more trainable parameters. As discussed in section 3.5.3, since the number of images on the training dataset is bigger, the architecture is able to learn a higher level of abstractions without overfitting the dataset. Overall, both architectures have a selected kernel of size 1. Using 1×1 convolutional layers have shown to reduce the dimension of an architecture while increasing the depth and improving feature representation [100]. Therefore, introducing the network size as an objective function encourages the learning framework to find efficient configurations that achieve a high accuracy.

The limitation of the proposed MEA learning framework relative to the tested Bayesian optimization method is the number of iterations needed to approximate the Pareto frontier. Viable solutions to address this problem include parallelizing the training of the candidate AdaResU-Net architectures in each generation and using a surrogate-assisted evolutionary process to reduce the number of architectures trained as presented in [101]. Also, in the present work the termination criteria was based on the stabilization of the whole non-dominated frontier, which increased the number of generations produced in most experiments. Considering only the convergence of the Pareto solution that minimizes the Dice loss can potentially reduce the number of iterations.

A limitation of the AdaResU-Net model is having a basic fixed architecture. However, designing a CNN involves an extensive number of hyperparameters that define an enormous search space. Therefore, to keep the problem computationally tractable, some hyperparameters need to be defined beforehand. In this work, the AdaResU-Net fixed architecture has been defined after a thorough review of successful CNN architectures for medical image segmentation and it is this tactic that allows the method to optimize for the segmentation error and model's size. Also, given the proven capability of the model to adapt to different datasets, we believe the unset group of hyperparameters provide a good flexibility.

The AdaResU-Net architecture and MEA learning framework have been developed for the segmentation of medical images. However, they are generic enough to be applied in other settings. The learning framework provides a technique that solves the challenging hyperparameter optimization problem in neural network design and can be applied for the fine-tuning of any architecture to a specific dataset. Compared with Bayesian optimization, the proposed algorithm provides architectures that perform comparably well with lesser number of trainable parameters. Furthermore, the objective functions of the MEA algorithm can be changed or new functions added to include additional considerations such as training speed, memory usage or distance-based measures (i.e., Hausdorff distance).

The AdaResU-Net has been designed to perform 2D segmentations. However, considering the limitations of a 2D CNN, encouraging quantitative results are obtained in 3D segmentation tasks. Future work will focus on modifying the AdaResU-Net architecture for 3D image segmentation by including 3D convolutions. The proposed MEA algorithm can also be applied for the segmentation of multiple structures from medical images by replacing the Dice loss with a weighted cross entropy loss in the optimization. Finally, the presented AdaResU-Net has shown to adapt and have a better performance than the U-Net in the segmentation of the prostate and endocardium. Given that the U-Net has been successfully applied to the segmentation of natural images [102, 103], we believe the AdaResU-Net can also be used in other non-medical segmentation tasks while providing smaller architectures.

Chapter 4: AdaEn-Net-An Ensemble of Adaptive 2D-3D Fully Convolutional Networks for Medical Image Segmentation

4.1 Note to Reader

This chapter has been previously published in *Neural Networks, 2020* (see reference [104]), and has been reproduced with permission from Elsevier Publishing. The copyright permission for reuse can be found in Appendix A.

4.2 Introduction

In this chapter, an adaptive 2D-3D ensemble of fully convolutional networks for medical image segmentation that incorporates volumetric information while optimizing both the performance and model's size is presented. The proposed **adaptive ensemble**, called AdaEn-Net, has two main components: a 2D FCN that extracts intra-slice and long-range 2D contexts, and a 3D FCN architecture that exploits inter-slice and volumetric information. Build upon the 2D AdaResU-Net model, the 2D and 3D architectures have an encoder-decoder structure that is automatically fitted for a specific medical image dataset. The adaptation process is performed using a multiobjective evolutionary based algorithm that maximizes the expected segmentation accuracy and minimizes the model's size. The basic building module of the proposed FCNs are residual blocks, which have proven to improve information transmission and gradient flow [105]. During the search process, the evolutionary based algorithm determines the optimal number of residual blocks, kernel sizes, and number of filters. Thus, simultaneously optimizes the width and depth of the network. The final segmentation takes advantage of the 2D FCN and 3D FCN by combining the results through an ensemble network. The method is evaluated on the task of prostate segmentation from the PROMISE12 Grand Challenge [81], and cardiac segmentation from the MICCAI ACDC challenge [106]. In both benchmarks, the AdaEn-Net achieved a rank within the top performing algorithms in the leaderboard. It achieves comparable performance to manually-designed architectures and surpasses the performance of automatically-designed architectures, while being considerably smaller in size.

4.3 Methods

The AdaEn-Net is an adaptive 2D-3D ensemble network that is constructed in two phases as shown in Figure 13. In Phase I, the 2D FCN and 3D FCN architectures are automatically fitted to a specific segmentation task. This is achieved by first dividing the medical image dataset into k -folds and selecting a fold at random to define the 2D FCN and 3D FCN architectures. Each FCN has a symmetrical encoder-decoder structure where the hyperparameters of the final architecture are learned using our previously developed Multiobjective Evolutionary based Algorithm (MEA) algorithm [71]. In the present work, the MEA algorithm defines the architecture by minimizing two objective functions: the number of trainable parameters and the expected segmentation error. The latter is quantified through a novel objective function that considers the training loss, validation loss, and expected training improvement of the networks. This makes the MEA algorithm search for architectures that have enough capacity to learn the relevant features from the training data and improve their performance in unseen images after being fully trained. The MEA algorithm is applied independently on the 2D FCN and 3D FCN architecture adaptation, allowing for the fitting process to be performed in parallel on different GPUs.

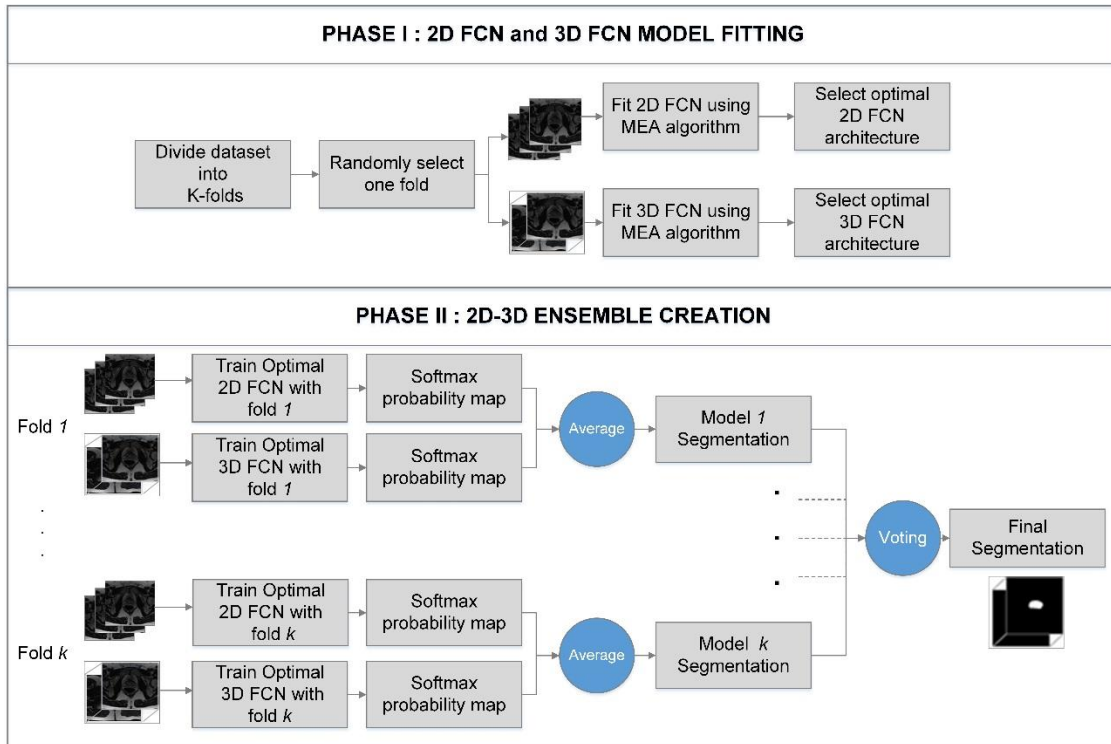


Figure 13 Overview of the proposed framework for the 2D-3D FCN ensemble construction. In Phase I, the dataset is divided into k -folds and one fold is selected to find the optimal 2D and 3D FCN architectures. In Phase II, the 2D-3D ensemble network is constructed

After finding the optimal 2D FCN and 3D FCN architectures, the 2D-3D ensemble network is constructed in Phase II. This is accomplished by training the 2D and 3D optimal architectures with each of the k folds from the training dataset and averaging their corresponding softmax probability maps in each fold. Thus, k 2D-3D FCN ensemble models are formed resulting in k predicted segmentations. The final image segmentation is achieved by forming a second level ensemble, in which a technique of majority voting is applied between the k predicted segmentations.

In the following sections, we first describe Phase I of the framework that consists of defining the 2D FCN and 3D FCN structure, the hyperparameter search space, and the application of the MEA algorithm to the particular problem. Secondly, we describe how the 2D-3D ensemble network is constructed and applied for 3D image segmentation.

4.3.1 Phase I: 2D-3D FCN Model Fitting

4.3.1.1 FCN Structure

A schematic view of the FCN architecture is shown in Figure 14. The 2D FCN and 3D FCN have a similar structure. The difference relies in that the 2D FCN receives as input image slices and performs 2D convolutions, while the 3D FCN processes 3D images and applies volumetric convolutions. The FCN has an encoder-decoder structure with an equal number of down-sampling and up-sampling modules. As shown in Figure 14 a), each module consists of three convolutional stages, each composed of a zero-padded convolutional layer, batch normalization layer, and an activation function layer. The first and last convolutional stages are linked through a residual connection [72], forming a residual block. In the down-sampling module, the residual block is followed by a max-pooling layer with a stride of 2 that reduces the feature map size by half. The up-sampling module is implemented through a transposed convolutional layer that doubles the size of the feature map. Opposite down-sampling and up-sampling residual blocks are connected through a merge operation. This long-range connection has shown to enable low-level feature preservation, promote information sharing, and improve gradient propagation [22]. Furthermore, to prevent overfitting, a spatial dropout layer [107] is included before the residual blocks (except for the first residual block). Regular dropout assumes independence between feature map activations [108]. However, adjacent pixels in an image are usually highly correlated, causing feature map activations to also have a strong correlation. Hence, independently dropping the activation of a neuron in a convolution feature map does not prevent co-adaptation. Spatial dropout

overcomes this limitation by dropping the entire feature map. The last convolutional layer of the architecture has a fixed kernel of size 1 and a softmax activation function.

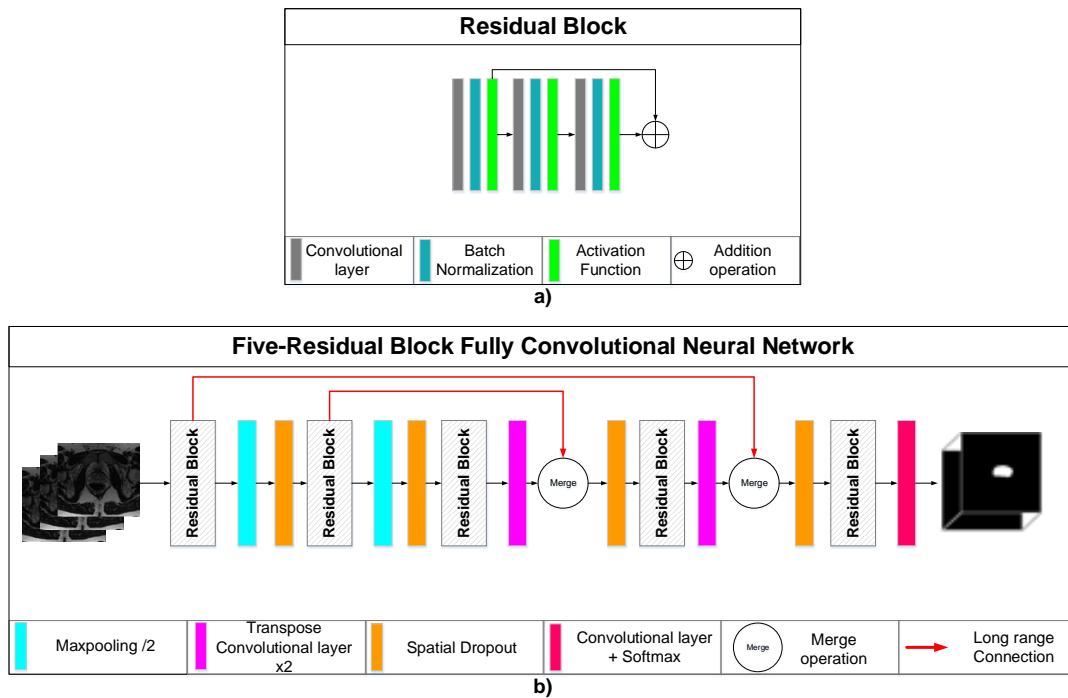


Figure 14 Schematic view of the FCN architecture. a) The residual block is the basic building module of the FCN architecture. b) An example of a five-Residual Block FCN.

Having the overall structure of the FCN defined, the architecture is constructed by determining the following hyperparameters: total number of residual blocks, kernel size for each convolutional layer, number of filters on each convolutional layer, activation function, merge operation to connect feature maps in opposite modules, spatial dropout probability, and learning rate. These hyperparameters represent the decision variables the MEA algorithm optimizes to adapt the 2D and 3D FCN architectures to a specific image dataset. Table 15 shows the set of hyperparameters along with their corresponding search space. The number of residual blocks, kernel sizes and number of filters completely define the depth and width of the neural network. Therefore, the MEA algorithm aims to find the appropriate architecture while simultaneously minimizing its size. Once the MEA algorithm determines the hyperparameters to construct the FCN, the weights of the FCN (which we refer as parameters) are trained with the traditional stochastic gradient descent algorithm.

Time is a major concern when automatically or manually designing neural networks because it requires training feasible configurations. As the hyperparameter search space expands, the number of potential architectures trained must also increase to effectively analyze the correlation between the tested hyperparameters and the loss

function. Hence, to reduce the complexity of the optimization problem and convergence time, all the unset hyperparameters are strategically summarized into nine decision variables as shown in Table 15. In the following paragraphs we describe the decision variables.

Table 15 Set of hyperparameters and corresponding search space the MEA algorithm optimizes for the construction of the 2D FCN and 3D FCN. Residual blocks refer to the total number of blocks in the FCN (depth).

Hyperparameter	Formula	2D FCN Search Space	3D FCN Search Space
Residual Blocks	$2n_b + 1$	$n_b \in [1,2,3,4]$	$n_b \in [1,2,3,4]$
Number of filters for NF_1	2^{n_f}	$n_f \in [2,3,4,5]$	$n_f \in [2,3,4,5]$
Kernel size for Conv. layer 1	-	[1x1, 3x3, 5x5, 7x7]	[1x1x1, 3x3x3, 5x5x5]
Kernel size for Conv. layer 2	-	[1x1, 3x3, 5x5, 7x7]	[1x1x1, 3x3x3, 5x5x5]
Kernel size for Conv. layer 3	-	[1x1, 3x3, 5x5, 7x7]	[1x1x1, 3x3x3, 5x5x5]
Activation Function	-	[ReLu, elu]	[ReLu, elu]
Merge Operation	-	[Summation, Concatenation]	[Summation, Concatenation]
Dropout Probability	$0.05 * n_p$	$n_p \in [0,1, \dots, 14] \subset \mathbb{Z}$	$n_p \in [0,1, \dots, 14] \subset \mathbb{Z}$
Learning Rate	-	$[1 \times 10^{-8}, 9 \times 10^{-3}]$	$[1 \times 10^{-8}, 9 \times 10^{-3}]$

Total number of residual blocks: First, the number of residual blocks (N_{block}) specifies the depth of the neural network by defining the number of down-sampling and up-sampling modules to be implemented. This value is obtained using Equation (11):

$$N_{block} = 2n_b + 1 \text{ where } n_b \in \{1,2,3,4\} \quad (11)$$

When constructing the FCN architecture, n_b residual blocks constitute the down-sampling path, n_b residual blocks comprise the up-sampling path, and one residual block connects the down-sampling and up-sampling paths. An example of a FCN with 5 residuals blocks is displayed in Figure 14 b). In this example, the first 2 residual blocks are part of the down-sampling path, the last 2 residuals blocks make up the up-sampling path and the middle residual block connects the down-sampling and up-sampling path.

Kernel size and number of filters on each convolutional layer: The width of the FCN is defined by the kernel size and number of filters in the convolutional layers. Three variables define the kernel size of the three convolutional layers in the residual blocks. Specifically, each variable defines the kernel size of a convolutional layer and we implement the same size for the kernel's width, height and depth (in the case of 3D convolutions).

On the other hand, the number of filters for the three convolutional layers in a residual block are set to the same value. This value is determined based on the position of the residual block in the down-sampling or up-sampling path, and the number of filters assigned to the first residual block of the network (NF_1). In order to obtain a symmetric architecture, NF_1 is determined utilizing Equation (12). Here, n_f is set to an integer value to force the number of filters in NF_1 to be a power of 2.

$$NF_1 = 2^{n_f} \text{ where } n_f \in [2,3,4,5] \quad (12)$$

To determine the number of filters for the remainder residual blocks, the residual blocks in the FCN are first enumerated from 1 to N_{block} . Then, the rule used to compute this value is to double the number of filters after a max-pooling operation and halve the number of filters after a transposed convolution. In this way, whenever the feature map is halved or doubled with the max-pooling operation or transposed convolution, the hidden state dimensions will remain approximately constant in all residual blocks. Therefore, the information extracted at each layer is preserved as well as the time complexity. The number of filters for residual block i (NF_i) is computed using Equation (13):

$$NF_i = \begin{cases} NF_1 * 2^{i-1} & \text{for } i \in [1, 2, \dots, \frac{N_{block}}{2} + 0.5] \\ NF_1 * 2^{N_{block}-i} & \text{Otherwise} \end{cases} \quad (13)$$

The number of filters in the entire architecture is encoded into one variable, which is the number of filters in the first residual block of the FCN. The number of filters for the rest of the residual blocks is computed with Equation (13).

Activation function, merge operation, spatial dropout probability, and learning rate: The activation function and merge operation used to connect residual blocks from down-sampling and up-sampling layers are encoded into two categorical variables. The same activation function is applied to all residual blocks and the same merge operation is used for all long range connections. Finally, to perform a more effective search, the learning rate is sampled from a logarithmic scale and the spatial dropout probability is discretized into multiples of 0.05.

As observed in Table 15, kernels of size 7x7 are considered in the search space of the 2D FCN. The purpose is to allow the MEA algorithm to select kernels that capture long-range spatial correlations in two dimensions. Although it would be ideal to have kernel size 7x7x7 as an option for the 3D FCN, the memory consumption and training time would be too high. Thus, we decided to allow the 3D FCN to specialize in short range 3D contextual information and complement it with the wider field of view of the 2D FCN through the proposed ensemble. In Figure 15, we present an example of hyperparameter values and their corresponding decoded 2D and 3D FCN architectures.

4.3.1.2 MEA Algorithm

The adaptation of the 2D FCN and 3D FCN to a medical image dataset is performed by applying the MEA algorithm shown in Algorithm 1. The adaptation process is modeled as a multiobjective hyperparameter optimization problem. In this problem, the aim is to find a model's hyperparameter configuration x that minimizes a set of M objective functions $F(x) = \{f_1(x), \dots, f_M(x)\}$. In the M objective functions, at least one function must measure the model's prediction error. When the objective functions are conflicting, often one single optimal solution will not

minimize all functions. Thus, the objective is to find the set of Pareto optimal solutions, also known as non-dominated solutions. A solution is Pareto optimal if none of the objectives functions can yield an improvement without degrading at least one other objective function. The image of a Pareto optimal solution in the criterion space is called a non-dominated point and the set of all non-dominated points is referred as the Pareto Front. In the MEA algorithm, the approximate set of Pareto optimal solutions is found by using the Multiobjective Evolutionary Algorithm based on Decomposition (MOEA/D) [77] with a Penalty based Intersection approach (PBI). Furthermore, the non-dominated solution that minimizes the segmentation error is selected as optimal and used to construct the neural network.

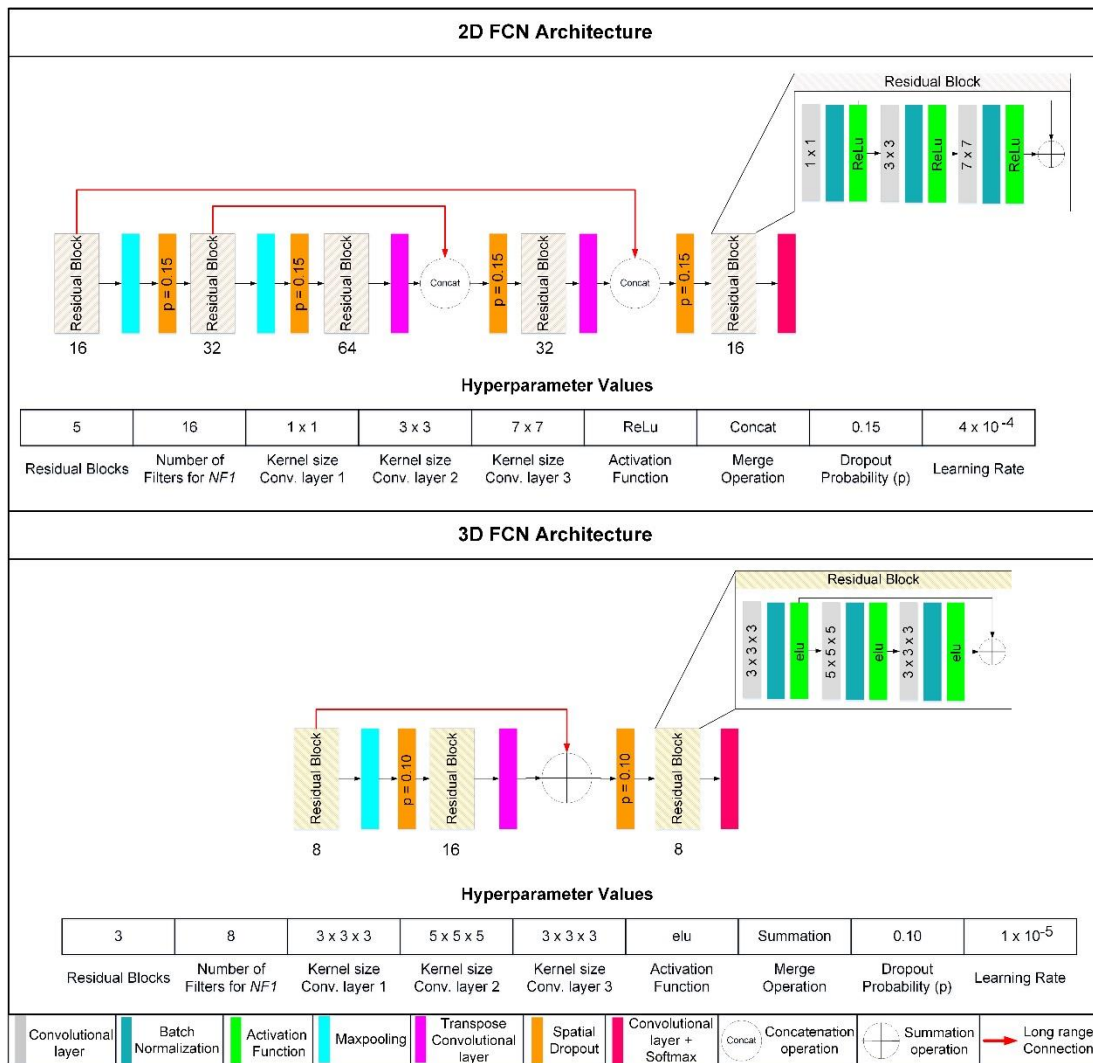


Figure 15 Example of two possible hyperparameter values and their corresponding decoded FCN architectures. On the top, a 2D FCN architecture. On the bottom, a decoded 3D FCN with 3 residuals blocks is presented. The number of filters for the three convolutional layers in a residual block are shown below the residual block figure, and the kernel size for a convolutional layer is shown inside the convolutional layer figure.

In this work, the nine hyperparameters shown in Table 15 constitute the hyperparameter configuration vector $x = \{x^{(1)}, x^{(2)}, \dots, x^{(9)}\}$. As previously discussed, this hyperparameter vector encodes the information to construct the 2D FCN or 3D FCN architecture. Each hyperparameter has a search domain $\Omega^{(1)}, \dots, \Omega^{(9)}$, also defined in Table 15, and the cross-product of these domains constitute the hyperparameter search space $\Omega = \Omega^{(1)} \times \dots \times \Omega^{(9)}$. The ultimate goal is to find a FCN architecture (in 2D or 3D) that can accurately segment the regions of interest in 3D medical images while maintaining the smallest possible size. Hence, two conflicting objective functions are defined: minimizing the expected segmentation error and minimizing the number of parameters in the model. For the first objective function, a loss function based on the multi-class Dice similarity coefficient is implemented.

The Dice similarity coefficient measures the spatial overlap between two segmentations and is defined in Equation (14):

$$Dice = \frac{2 \sum_i \hat{y}_i y_i}{\sum_i \hat{y}_i + \sum_i y_i} \quad (14)$$

where \hat{y}_i are the voxels from the predicted segmentation and y_i the voxels from the ground truth segmentation. This coefficient ranges between 0 and 1, where values near 1 indicate significant overlap and values close to 0 no spatial overlap. The multi-class Dice coefficient is applied when multiple structures need to be segmented from an image. Therefore, it measures the overlap between the predicted and ground truth segmentation over all classes and is defined as shown in Equation (15):

$$MCDice = \sum_c \frac{2 \sum_i \hat{y}_{ic} y_{ic}}{\sum_i \hat{y}_{ic} + \sum_i y_{ic}} \quad (15)$$

where \hat{y}_{ic} are the voxels that correspond to the predicted segmentation in class c , and y_{ic} are the voxels from the ground truth that are part of class c . This coefficient now ranges between 0 and C , where C is the number of classes. Values near C indicate a significant overlap between the predicted and the ground truth segmentations of the substructures. In this work, the function $C - MCDice$ is denominated as the multi-class Dice loss.

In this work, we introduce a new loss function for the MEA algorithm to quantify the expected segmentation error, which is composed of three terms: the multi-class Dice loss in the training set, the multi-class Dice loss in the validation set, and the distance to the training epoch with the maximum validation multi-class Dice coefficient. The first two terms are used to measure the architecture's segmentation accuracy. Meanwhile, the last term aims to account for the expected segmentation improvement if the architecture is trained for additional epochs. The latter term is added because we implement the strategy of partial training during evolution. The loss function is defined in Equation (16).

$$f_1(x) = \alpha(C - MCDice_{Train}) + (C - MCDice_{Val}) + \beta \left(\frac{E - e_{max}}{E} \right) \quad (16)$$

where $MCDice_{Train}$ and $MCDice_{Val}$ are the multi-class Dice coefficients in the training and validation set, respectively. These values are computed for each set by averaging the multi-class Dice coefficient of the last 5 training epochs. This number of epochs has shown to reduce random variation while providing a good estimation of the loss. E is the maximum number of epochs the candidate architectures are trained for, e_{max} is the number of epoch the maximum validation multi-class Dice coefficient is attained, and α and β are weight parameters whose ranges are between $[0, 1]$.

In the loss function, both the multi-class Dice coefficient in the training and validation set are considered. This is because in some problems, the training set can be as hard to segment as the validation set. Therefore, it is important to construct an architecture that can adequately segment both sets of images. Furthermore, since our second objective function is to minimize the size of the network, it is also important to ensure that the resultant architecture has enough capacity to learn important features from the training data. To ensure generalization, an α weight is added to balance the effect the training multi-class Dice coefficient has over the hyperparameter optimization process.

To reduce the time the MEA algorithm takes to find a solution, we apply the strategies of partial training and early termination. All candidate architectures are partially trained for a maximum of E epochs. The selection of E provides a trade-off between the computational time and the quality of the optimal solution found. In this work, we test distinct values of E and select the number of epochs that provide a good balance between the two indicated factors. It should be noted that ensembling networks, which each represent a local optimal solution, increases the robustness of the final prediction and provides a better approximation to the true global optimal solution. Therefore, the proposed network does not require the full training of the candidate architectures to obtain a good performing ensemble model. Also, we prematurely terminate an architecture's training if the validation multi-class Dice coefficient stops improving for a consecutive number of epochs or if exploding/vanishing gradients are encountered during backpropagation.

Using partial training and early termination helps improve the efficiency in the evaluation process. However, these strategies can lead to an incorrect rank of an architecture's performance. Particularly, two architectures can be ranked equally if their average multi-class Dice coefficient is similar during the partial training. However, it could be the case that one architecture has already reached its full learning capacity during the E training epochs, while the other architecture could improve its performance if it is trained for additional epochs. An example is shown in Figure 16, candidate architectures FCN¹ and FCN² have a similar average validation $MCDice$ during the E training epochs.

Nevertheless, FCN^1 has already achieved the maximum performance and the validation $MCDice$ remains the same after increasing the number of training epochs. Meanwhile, the validation performance of FCN^2 improves (achieves a higher validation $MCDice$) with additional training and surpasses the performance of FCN^1 . To differentiate these network configurations, the distance between the epoch with the maximum validation multi-class Dice coefficient (e_{max}) and the maximum number of training epochs (E) is calculated. We assume that architectures that have their e_{max} closer to E have not reached their full learning capacity and can increase their segmentation accuracy with additional training. Thus, the third term in the loss function penalizes the normalized distance between e_{max} and E .

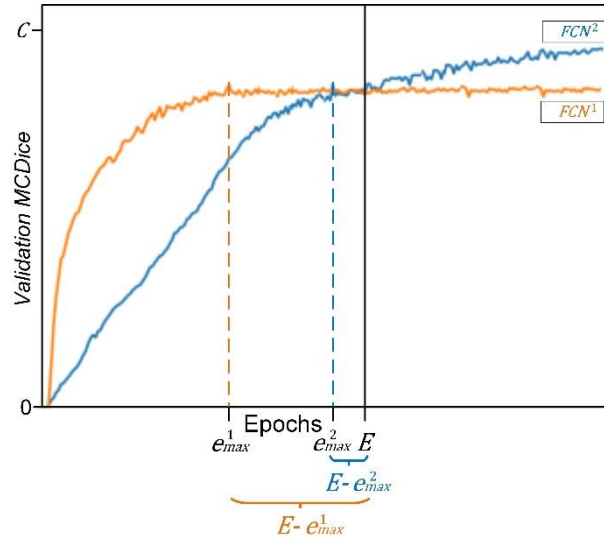


Figure 16 Validation multi-class Dice coefficient ($MCDice$) for the candidate architectures FCN^1 and FCN^2 during the training epochs. Candidate architectures are partially trained for E epochs and e_{max}^i represents the epoch with the maximum validation $MCDice$ for architecture i .

The α and β weight parameters in the loss function (16) balances the importance of the Dice loss in the training set, an architecture's expected performance improvement, and the loss in the validation set. Setting these weights depends on the specific segmentation problem. However, through experiments we have found $\alpha = 0.25$ to provide architectures with good capacity and a high generalization capability. On the other hand, the value of β is influenced by the number of maximum training epochs E . If E is small, most architectures would not have reached their full learning capacity by the end of the partial training and could improve if trained for additional epochs. On the other hand, if E is high most architectures would have already reached their full learning capacity by the end of the training. Thus, no improvement should be expected with further training. In both extreme cases, the third term in the loss function (16) that aims to quantify the improvement with further training would no longer be useful at discriminating the quality of the found solution. Hence, β should be set close to 0. Otherwise, $\beta = 0.25$ has shown

good results. The second loss function the MEA algorithm minimizes is the number of parameters in the network. The multiobjective optimization problem is described by Equations (17), (18), and (19):

$$\text{Minimize } f_1(x) = \alpha(C - MCDice_{Train}(\theta)) + (C - MCDice_{Val}(\theta)) + \beta \left(\frac{E - e_{max}}{E} \right) \quad (17)$$

$$\text{Minimize } f_2(x) = |\theta| \quad (18)$$

$$\text{subject to } x \in \Omega \quad (19)$$

where x is the 9-dimensional hyperparameter vector, Ω represents the hyperparameter search space, θ are the parameters in the FCN learned through backpropagation, and $|\cdot|$ is the cardinality operator that measures the number of parameters in the neural network.

A flowchart describing the implementation of the MEA algorithm is shown in Figure 17. The MEA algorithm starts by forming the initial population. This is achieved by randomly sampling N hyperparameter vectors from the hyperparameter search space and decoding them into N FCN architectures. These architectures are trained using the ADAM optimizer and applying the multi-class Dice loss as the loss function. The objective functions from Equations (17) and (18) are quantified for each FCN architecture and the values and corresponding solutions saved as the Pareto Front and set of non-dominated solutions, respectively. In each generation afterwards, for each subproblem, two solutions from the neighborhood are randomly selected. By applying cross-over and mutation operators to the two solutions, a new hyperparameter vector is created. This new hyperparameter vector is decoded into a candidate FCN architecture. The architecture is trained with the ADAM optimizer and the objective functions (17) and (18) computed. The set of neighboring solutions are updated using a Penalty-based Boundary Intersection (PBI) approach [77]. If the new solution dominates any point from the current non-dominated set, the set of non-dominated solutions and Pareto Front are updated. After the maximum number of generations is reached, the algorithm terminates and the set of non-dominated solutions and Pareto Front are returned. As initially discussed, the hyperparameter vector that minimizes the expected segmentation error in the set of non-dominated solutions is considered optimal. This hyperparameter configuration is used to construct the optimal FCN that will be part of the proposed 2D-3D FCN ensemble.

In the present work, the mutation operator is applied independently to each component of the hyperparameter vector with a probability that decreases monotonically. We propose a new mutation probability function by combining the characteristics from the mutation functions in [109] and [110]. A high mutation probability is set at the beginning of the search to diversify the candidate architectures constructed in the initial generations. This allows the algorithm to explore different areas of the search space and avoid premature convergence to a low quality solution. As the

number of generations increases, an exploitation strategy is preferred. Therefore, we opt to reduce the mutation probability up to a minimum value. This behavior avoids testing solutions that are considerably far from the current solution on late generations to prevent turning the optimization algorithm into a random search and allow faster convergence. The mutation probability $p(g)$ at the beginning of each generation g is computed using Equation (20):

$$p(g) = \max\left(\left[1 - \frac{\ln(g)}{\ln(G)}\right], \frac{1}{D}\right) \quad (20)$$

where G is the maximum number of generations, g is the current generation, and D the number of components in the hyperparameter configuration vector. In this work $D = 9$.

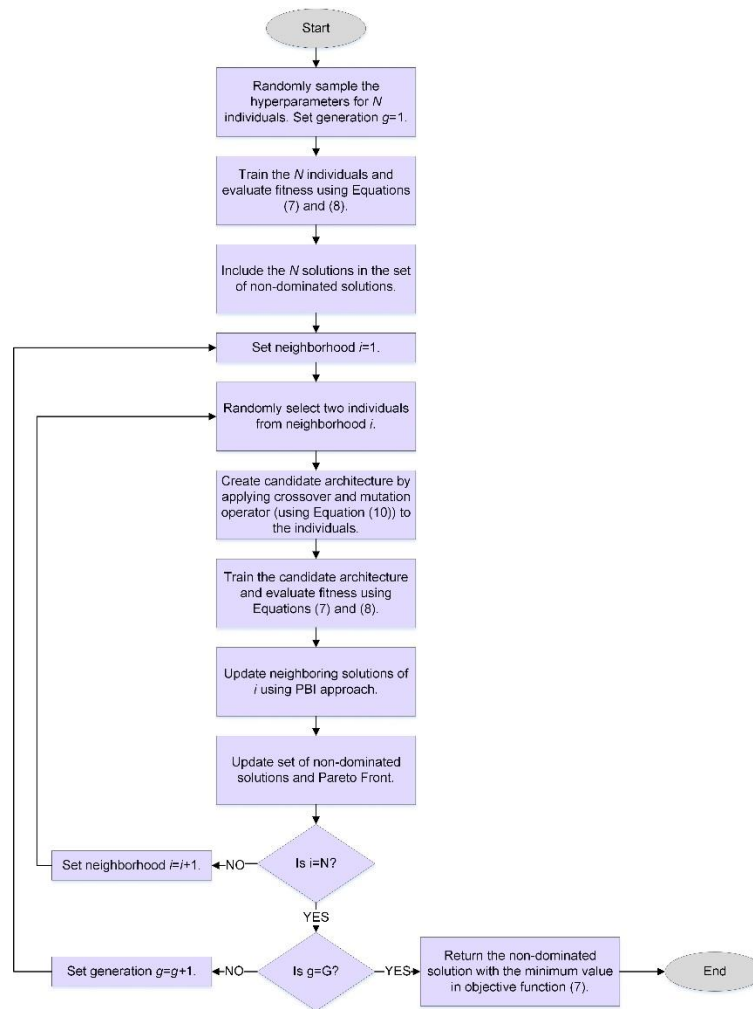


Figure 17 Flowchart of the MEA algorithm.

The values of the parameters used to implement the MEA algorithm are presented in Table 16. The population size, neighborhood size, rectification parameter and crossover probability are similar to those used in [71]. The maximum number of generations G has been set to 40 after finding that the algorithm converges to good performing

solutions. Furthermore, a high number of generations is not required because by forming an ensemble the representability of the AdaEn-Net is increased even if we converge to a local optima solution.

Table 16 Parameters for the MEA algorithm to obtain the optimal architecture.

Parameter	Value
Generations	40
Population size	8
Neighborhood size	4
Rectification parameter	0.80
Penalty factor	0.10
Crossover probability	0.50
Mutation probability	$\max\left(\left\lfloor 1 - \frac{\ln(g)}{\ln(40)} \right\rfloor, \frac{1}{9}\right)$

4.3.2 Phase II: 2D-3D Ensemble Network

After Phase I, the 2D and 3D FCNs optimal architectures are obtained. With each of the K folds from the training dataset, the optimal 2D FCN and 3D FCN are trained from scratch until convergence using the backpropagation algorithm and the ADAM optimizer. The parameters for the optimizer are set to beta 1: 0.9, beta 2: 0.999 and epsilon: 1×10^{-8} as suggested in [76]. The networks are trained using the multi-class Dice loss function, which was previously defined as $C - MCDice$ in Section 4.3.1.2. The learning rate and spatial dropout probability found with the MEA algorithm for each FCN are applied for training. Also, real-time data augmentation is implemented during training to improve generalization. Specifically, we apply random rotations, random vertical and horizontal translations, horizontal flip and random scaling on the training images and ground truth labels. Applying three dimensional data augmentation can be suboptimal in anisotropic data. Hence, we apply the transformations in a slice-wise manner for each sample. The weights in the 2D and 3D FCNs are initialized from a Gaussian distribution centered on 0 with a standard deviation of $\sqrt{2/N}$, where N are the number of input nodes of a neuron [111]

The predictions from the trained 2D FCN and 3D FCN in a fold are combined by averaging their softmax probability maps. Afterwards, the class for each pixel is assigned by selecting the label that has the highest probability. To increase the robustness and accuracy of the predicted segmentation, a second ensemble is formed. The latter is obtained by aggregating the K 2D-3D FCN ensemble models through a majority voting as shown in Figure 13. Therefore, a two-level ensemble network is proposed. In the first level, the 2D FCN and 3D FCN predictions are averaged forming the 2D-3D FCN ensembles. These ensembles provide a better segmentation accuracy by combining the intra-slice and inter-slice volumetric information. In the second level, the predictions from the K 2D-3D FCNs are

combined through a majority voting to produce the final segmentation prediction. This process is formulated as described by Equations (21) and (22):

$$\hat{y}_{ENS}^i = \operatorname{argmax}_c \sum_k H_k^{i,c} \quad (21)$$

where

$$H_k^{i,c} = \begin{cases} 1 & \text{if } \operatorname{argmax}_{c \in \operatorname{dom}(\hat{y}^i)} \frac{1}{M} \sum_m \hat{y}_{m,k}^i \text{ assigns class } c \text{ to pixel } i \\ 0 & \text{Otherwise} \end{cases} \quad (22)$$

\hat{y}_{ENS}^i is the predicted class label for pixel i using the ensemble network, $H_k^{i,c}$ is an indicator function, $\hat{y}_{m,k}^i$ is the predicted class softmax probabilities using model m trained in fold k , and M is the total number of models in the first-level ensemble.

Previous work has shown that the performance of an ensemble of classifiers can be at least as good as the best individual member if the classifiers are accurate and diverse [112]. Ensemble diversity is attained when the classifiers make different errors on new data points while an accurate classifier has a smaller error rate than random guessing. Therefore, it is critical for the success of an ensemble to be composed of complementary models that fail differently on the test set. Designing a diverse ensemble model is a non-trivial task. In the proposed framework, three strategies are implemented to promote error diversity, namely varying the neural network architectures, varying the training data, and varying the training parameters. As discussed in Section 4.3.1.1, the 2D FCN and 3D FCN architectures apply distinct convolutional operations and the architecture search process is performed independently using a different input type (slices vs. volumes). This results in an optimal 2D FCN and 3D FCN that differ significantly in structure. For the first level ensemble, the different types of architectures for 2D and 3D provide diversity. Moreover, the two types of architectures extract different spatial relationships allowing the ensemble to combine the information and provide a better pixel-wise classification.

In the second level ensemble, diversity is encouraged by training each 2D-3D ensemble in different folds of the original training set. Likewise, randomness is incorporated during training when initializing the weights and performing data augmentation to the input images. Error diversity stems from the fact that the loss function used to train neural networks has multiple local minima. Local search algorithms, such as stochastic gradient descent, will usually converge to this local optima. The aim of introducing randomness during training is to obtain neural networks that relate to different local optima and consequently do not make the same errors in unseen images. Similarly, neural networks trained in distinct datasets will tend to form different generalizations about the training set and make less correlated errors.

4.4 Experiments

The proposed AdaEn-Net is evaluated on two medical image segmentation challenges. The first corresponds to the segmentation of the prostate in anisotropic MR images from the MICCAI PROMISE12 challenge [81]. In the second challenge, the task is the segmentation of the left ventricle, right ventricle and myocardium in MR images from the MICCAI ACDC database [106]. The AdaEn-Net is implemented using Python 3.6 and Keras library [87]. The experiments were carried on an 8-GB NVIDIA GeForce GTX 1080 GPU, 3.60-GHz CPU and 16-GB RAM. In the following subsections, the datasets, implementation details, quantitative evaluation and comparison with state-of-the-art models are reported.

4.4.1 Prostate MR Image Segmentation Challenge (PROMISE12)

4.4.1.1 Dataset and Pre-processing

The publically available dataset is composed of 50 training cases with transverse T2-weighted MR images of the prostate and their corresponding reference segmentations. The test set consists of 30 cases for which the reference segmentations are hidden. The evaluation of the test cases is carried out via an online submission. The dataset includes images from different centers obtained with distinct acquisition protocols, scanners and field of strength. Therefore, the images exhibit a high variation in in-plane and through-plane resolution, position, anatomic appearance, and field of view.

The dataset is pre-processed by resampling the images to a spatial resolution of $1 \times 1 \times 1.5$ mm and setting them to a fixed size of $128 \times 128 \times 64$ voxels. Also, all pixel intensities outside 3 standard deviations from the mean are eliminated and replaced with the maximum allowed value. Finally, the pixel values are rescaled to a 0-1 range in a slice-wise manner. This is achieved by subtracting the minimum pixel value and dividing the result with the maximum pixel value in each slice. For the 2D FCN, the input are the slices of size 128×128 whereas the 3D FCN is trained with randomly extracted patches of size $96 \times 96 \times 16$.

4.4.1.2 Implementation of the AdaEn-Net

The prostate dataset is divided into 5-folds. One randomly selected fold is used to find the optimal 2D FCN and 3D FCN. The search process for both architectures is similar. 40 cases are used for training the candidate architectures and 10 cases for computing the validation loss. The hyperparameters to be fitted and their search space are presented in Table 15 whereas the parameters for the MEA algorithm are shown in Table 16. Additionally, an $\alpha = 0.25$ and $\beta = 0.25$ are applied for computing the expected segmentation error in the objective function (17). The

candidate architectures are trained for 120 epochs each. Running the 40 generations took approximately 66.31 hours for the 2D FCN search and 118.08 hours for the 3D FCN search. The optimal hyperparameter values for both architectures as well as the number of parameters in each model are presented in Table 17.

As Table 17 shows, the optimal architectures for the 2D and 3D FCNs differ significantly. In comparison to the 2D FCN, the 3D FCN is a shallower but wider network. Our hypothesis is that as volumetric images contain more information than slices, the 3D FCN is able to extract more significant features per layer. However, since the number of training images is small and the shape of the prostate is not substantially irregular, extending also the depth of the 3D FCN will end up in overfitting. Additionally, wider networks have shown to be good at capturing fine-grained details [113]. On the other hand, the 2D FCN has a deeper architecture and a larger receptive field by incorporating the kernel of size 7x7. This allows the 2D FCN to extract long-range 2D contextual information and provide a higher level of abstraction. In both cases, the spatial dropout probability is small, demonstrating that reducing the size of the network is an appropriate technique to prevent overfitting.

Table 17 Optimal hyperparameter values found with the MEA algorithm to construct the 2D FCN and 3D FCN for the prostate and cardiac datasets.

Hyperparameter	Prostate Dataset		Cardiac Dataset	
	2D FCN	3D FCN	2D FCN	3D FCN
Residual Blocks	7	5	7	9
Number of filters for NF_1	16	32	16	16
Kernel size for Conv. layer 1	1x1	3x3x3	3x3	1x1x1
Kernel size for Conv. layer 2	3x3	1x1x1	1x1	1x1x1
Kernel size for Conv. layer 3	7x7	5x5x5	7x7	3x3x3
Activation Function	ReLU	elu	elu	elu
Merge Operation	Concatenation	Concatenation	Summation	Concatenation
Dropout Probability	0.15	0	0	0
Learning Rate	4×10^{-4}	5×10^{-5}	4×10^{-4}	9×10^{-5}
Number of Parameters	1.6×10^6	3.9×10^6	1.5×10^6	3.3×10^6

The ensemble is formed by training the optimal 2D FCN for 3000 epochs in each fold and the 3D FCN for 6000 epochs. In the 3D FCN, an epoch is counted after one patch of all training images have been processed. The training details are presented in section 4.3.2. A batch size of 40 images was used for training the 2D FCN while a batch size of 4 image patches for the 3D FCN. The batch size for each model was set considering the maximum available computational memory. The weights with the minimum validation loss are used for predicting the final segmentation. The number of training epochs is set after verifying the validation loss achieved a minimum value and has stopped improving thereafter. Finally, a connected component analysis is applied as post-processing, where only the largest connected component is kept for the final predicted segmentation.

In Table 18, we present the evaluation metrics for the optimal 2D FCN, 3D FCN, and 2D-3D FCN ensemble using a 5-fold cross validation. A connected component analysis post-processing operation has been applied to the final segmentation of the three evaluated architectures. The metrics computed are the Dice similarity coefficient (DSC), sensitivity, 95 percentile Hausdorff distance (95 HD), and average boundary distance (ABD) between the predicted segmentation and ground truth segmentation. Additionally, a one-tailed paired t-test with a 95% confidence level is applied to compare the mean performance on the four evaluation metrics of all pairs of architectures. The model with the best performance is shown in bold in Table 18. The p-values for each paired t-test is presented in Table 19 and statistically significant results are indicated in bold.

Table 18 Evaluation metrics for the predicted prostate segmentation using the optimal 2D-3D FCN ensemble, 2D FCN, and 3D FCN. The values displayed are the result from a 5-fold cross validation. Best performances are indicated in bold.

Model	DSC [%]	Sensitivity [%]	95HD [mm]	ABD [mm]
2D-3D FCN ensemble	90.42±2.87	90.13±5.93	3.07±1.08	0.92±0.28
2D FCN	89.56±3.09	89.19±6.13	3.36±1.09	1.02±0.32
3D FCN	89.26±3.47	89.18±6.69	3.47±1.33	1.07±0.39

Table 19 P-values for the one-tailed paired t-test between the 2D-3D FCN ensemble, 2D FCN, and 3D FCN for the prostate segmentation. Statistically significant p-values are indicated in bold.

Compared Models		DSC [%]	Sensitivity [%]	95HD [mm]	ABD [mm]
2D-3D FCN ensemble	2D FCN	1.8x10⁻⁶	2.6x10⁻⁴	2.1x10⁻³	3.1x10⁻⁴
2D-3D FCN ensemble	3D FCN	2.2x10⁻⁵	1.4x10⁻²	3.1x10⁻⁴	1.1x10⁻⁵
2D FCN	3D FCN	2.2x10 ⁻¹	4.9x10 ⁻¹	2.3x10 ⁻¹	1.4x10 ⁻¹

The experimental results show that the differences in mean performance between the 2D FCN and 3D FCN are not statistically significant. This is interesting given that each architecture is structurally different and extracts a distinct level of information. On the other hand, the 2D-3D FCN ensemble has a statistically better mean performance, with a 95% confidence, in all evaluation metrics than both the 2D FCN and 3D FCN. This provides two important conclusions. First, that the 2D-3D FCN ensemble is able to successfully integrate the information extracted by the 2D FCN and 3D FCN. Secondly, that the proposed method can generate a diverse and accurate set of models for ensemble learning. Hence, the performance of the ensemble is superior to the individual members.

A qualitative comparison of the segmentation results using the 2D FCN, 3D FCN, and 2D-3D FCN ensemble are presented in Figure 17. To provide a better visual comparison, we show the slices where the segmentations between the three models differ. The examples show that the 3D FCN tends to over segment the prostate but provides a better shape definition. The 2D FCN, on the other hand, appears to under segment certain regions and produce a sharper delineation of the prostate contours. Meanwhile, the 2D-3D FCN ensemble takes advantage of both models and

produces segmentations with better spatial distribution and delineation. The ensemble provides an appropriate balance between the false positives of the 3D FCN and the false negatives of the 2D FCN, generating a better quality segmentation.

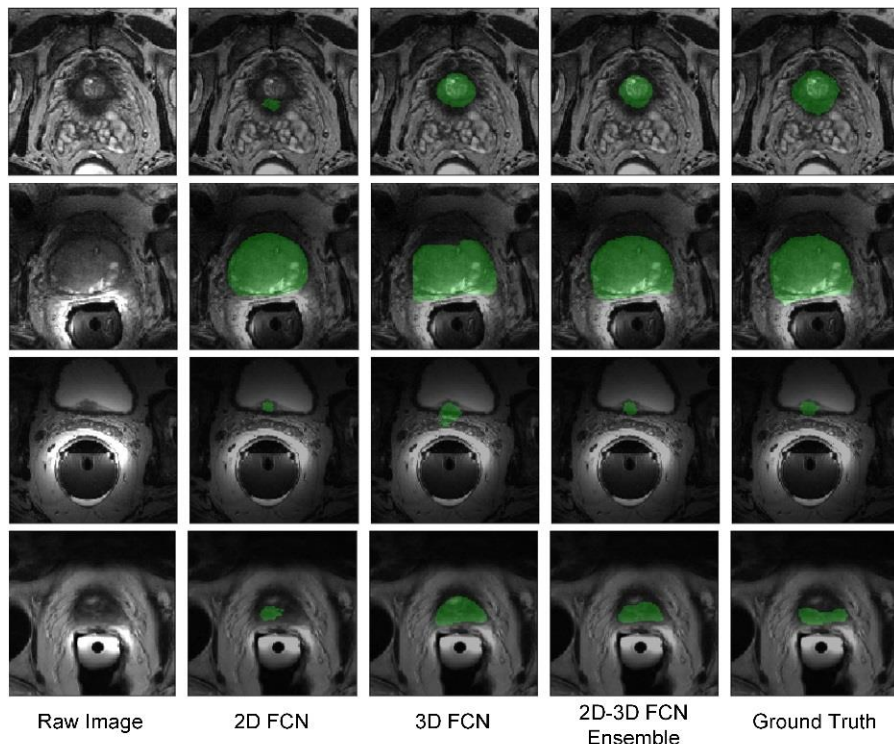


Figure 18 Examples of prostate segmentation using the optimal 2D FCN, optimal 3D FCN, and 2D-3D FCN ensemble on the validation dataset. The prostate region is denoted in color green.

4.4.1.3 Benchmark Results

The PROMISE12 benchmark evaluates the performance of the algorithms by computing the Dice similarity coefficient (DSC), absolute relative volume difference (aRVD), average boundary distance (ABD), and 95 percentile Hausdorff distance (95HD) between the submitted 3D segmentations and the segmentations obtained by experts. The metrics are calculated for the segmentation of the whole prostate, apex, and base parts of the prostate. The apex and base are considered to be the most difficult parts to segment due to the large inter-patient variability and differences in slice thickness [81]. The algorithms are ranked by combining all the mentioned metrics for the 30 test cases into a single score. The AdaEn-Net obtains an overall score of 89.29, which ranks it 9 out of 297 submissions as of September 2019. To compare the performance of our network, methods that are within the top 14 rank and have accompanied the submission with a methodology description (or have uploaded the description on a preprint server) were selected. This resulted in five competing models as presented in Table 20, Table 21 and Table 22. Other methods were not considered

because without a description, it is impossible to know how the displayed evaluation metrics were obtained and make a fair comparison of the results. Table 20 shows how the segmentation architectures were designed, overall score in the test set and the rank score. Table 21 and Table 22 presents the evaluation metrics averaged over the 30 test cases. These results were obtained from the challenge organizers webpage.

Table 20 Design type, overall score and rank score on the PROMISE12 challenge dataset. Only methods that have submitted a methodology description are displayed.

User	Design Type	Overall Score	Rank
AdaEn-Net	Automatic	89.293	9
HD_Net	Manual	90.344	1
Bowda-Net	Manual	89.585	2
Sunrise2014	Manual	89.461	6
nnU-Net (I)	Automatic	89.276	10
nnU-Net (II)	Automatic	89.076	14

Table 21 Mean Dice and 95% Hausdorff distance of the proposed method and five competing models on the PROMISE12 challenge dataset.

Model	DSC[%]			95 HD [mm]		
	Whole	Apex	Base	Whole	Apex	Base
AdaEn-Net	91.45	88.10	89.54	4.08	3.58	4.43
HD_Net	91.35	91.35	89.82	3.93	3.60	4.24
Bowda-Net	91.41	91.41	89.56	4.27	3.44	4.48
Sunrise2014	90.58	90.58	89.65	4.95	3.90	4.10
nnU-Net (I)	91.61	91.61	90.29	4.00	3.79	4.05
nnU-Net (II)	91.56	91.56	89.59	4.17	3.77	4.42

Table 22 Mean average boundary distance and absolute relative volume difference of the proposed method and five competing models on the PROMISE12 challenge dataset.

Model	ABD [mm]			aRVD[%]		
	Whole	Apex	Base	Whole	Apex	Base
AdaEn-Net	1.34	1.40	1.53	5.45	14.35	10.04
HD_Net	1.36	1.34	1.54	5.10	5.10	7.00
Bowda-Net	1.35	1.29	1.54	6.04	6.04	9.12
Sunrise2014	1.59	1.47	1.48	5.81	5.81	5.94
nnU-Net (I)	1.31	1.46	1.45	7.13	7.13	8.29
nnU-Net (II)	1.30	1.39	1.49	6.93	6.93	10.17

As shown in Table 20, architectures were automatically and manually fitted to the prostate segmentation task as denoted in the *Design Type* column. The top 3 models consist of manually designed encoder-decoder CNNs that implement specialized boundary attention mechanisms to overcome the lack of clear edge between the prostate and nearby tissues. Specifically, the two leading groups HD_Net and Bowda-Net designed an additional network to guide the prostate boundary segmentation. These boundary attention mechanisms seem to improve some of the apex evaluation metrics but require an additional architecture to assist the main segmentation network. Furthermore, the

resulting architectures are very specific for the task in hand making it difficult to generalize to other medical image datasets.

The AdaEn-Net performs the best when compared to other automatically adapted architectures. The nnU-Net [114] proposes a rule-based framework that automatically designs and executes the training pipeline of a group of U-Net architectures. The best performing model or ensemble in the training dataset is selected as optimal. Similar to our approach, a five-ensemble model is applied for the final prediction after using 5-fold cross validation. For the prostate task, the optimal nnU-Net is an ensemble of a 2D and 3D U-Net architectures with 29.4×10^6 and 43.7×10^6 number of parameters, respectively. Additionally, the nnU-Net group has two submissions: the first model, nnU-Net(I), was trained with images from the PROMISE12 challenge and images from an external dataset while the second model, nnU-Net(II), was trained with images solely from the challenge. The AdaEn-Net performs better than nnU-Net(I) and nnU-Net (II) while being $13 \times$ smaller (1.6×10^6 parameters for the 2D FCN and 3.9×10^6 for the 3D FCN). This demonstrates that our proposed AdaEn-Net is better at exploiting discriminative features from the available information and providing efficient architectures.

Overall, the AdaEn-Net performs better than the competing automatically-designed framework and is comparable to manually-designed architectures. In comparison with the latter, our model is generalizable to other segmentation tasks and avoids the time-consuming process of manually selecting the best hyperparameters for an architecture. In addition, our method also considers minimizing the model's size when constructing the optimal FCN architectures.

4.4.2 Automated Cardiac Diagnosis Challenge (ACDC)

4.4.2.1 Dataset and Pre-processing

The ACDC dataset consists of cardiac cine MR images acquired from 150 patients. The images were obtained with two MR scanners with distinct magnetic strength. The in-plane resolution ranges from 1.37 to 1.68 mm², with a slice thickness that varies from 5 to 8 mm and sometimes a slice gap of 5mm is present. A series of short axis slices cover the left ventricle from the base to the apex. Furthermore, depending on the patient, 28 to 40 3D images partially or completely cover the cardiac cycle. The 150 patients are divided into 5 evenly distributed subgroups that correspond to normal subjects, patients with previous myocardial infarction, patients with dilated cardiomyopathy, patients with hypertrophic cardiomyopathy and patients with abnormal right ventricle. The training dataset is composed of 100 patients with their corresponding manual reference. Meanwhile, the testing dataset has images from 50 patients and

no manual reference is available. The test evaluation metrics are obtained by submitting the predicted segmentations to the online portal. The goal of the contest is to segment the left ventricle cavity (LVC), left ventricle myocardium (LVM), and right ventricle cavity (RVC) on end diastolic (ED) and end systolic (ES) phases.

The MR images are resampled to $1.56 \times 1.56 \times 10$ mm per voxel and set to a fixed size of $192 \times 192 \times 10$ voxels. Identically to the pre-processing operations performed on the prostate dataset, the pixel values are clipped within 3 standard deviations from the mean and rescaled to a 0-1 range in a slice-wise manner. The 3D FCN is trained with random crops of size $144 \times 144 \times 10$ and the 2D FCN with slices of size 192×192 .

4.4.2.2 Implementation of the AdaEn-Net

The cardiac dataset is divided into 5-folds. Images from 80 patients are selected for the training set and images from 20 patients for the validation set. The search space explored and parameters for the MEA algorithm are presented in Table 15 and Table 16, respectively. Given the total number of slices in the dataset and the image resolution, training the candidate architectures for various epochs can be very time-consuming. Therefore, we set the number of training epochs E to 60 after testing its adequacy in distinguishing the quality of the solutions. However, with this number of epochs most tested architectures do not reach a stabilized performance by the end of the training. Thus, as explained in the Section 4.3.1.2, the ability of the factor $\left(\frac{E - e_{max}}{E}\right)$ in the objective function (17) to discriminate quality solutions is reduced. As a result, we set the parameter β to 0.1 and fix α to 0.25 as in the prostate dataset. Evolving the 40 generations took 116.54 hours for the 2D FCN search and 207.32 hours for the 3D FCN search. The optimal hyperparameter values found with the MEA algorithm for the 2D FCN and 3D FCN are shown in Table 17.

The optimal 2D FCN and 3D FCN are deep architectures with the same number of filters per layer. Interestingly, the optimal 2D FCN is very similar to the 2D FCN found for the prostate dataset. Thus, as previously discussed, we believe this structure allows the 2D FCN to analyze broad 2D spatial relationships. Differently from the 3D FCN found for the prostate dataset, the optimal 3D FCN is deeper. Specifically, the cardiac 3D FCN has the largest number of residual blocks and the smallest kernel sizes. Since in this dataset three differently shaped cardiac substructures need to be segmented, a deeper network can help to learn a higher level of abstraction and capture richer features. Furthermore, this hyperparameter selection confirms what various works have demonstrated, which is that deeper architectures have a bigger representation power and are better suited for complex tasks [115].

The optimal 2D FCN is trained for 3000 epochs and the optimal 3D FCN for 6000 epochs in each fold. For the 3D FCN, an epoch represents the forward and backward pass of a patch from all training images. The training is performed as described in section 4.3.2. Gradient updates are computed using a batch size of 20 images for the 2D FCN and batch size of 4 image patches for the 3D FCN. The batch size is determined considering the maximum computational memory available. The weights with the minimum validation loss during training are used to segment the test set images. For post-processing, a largest connected component analysis is performed for each predicted substructure.

To evaluate the performance of the 2D FCN, 3D FCN, and 2D-3D FCN ensemble, the Dice similarity coefficient (DSC) and 95 percentile Hausdorff distance (95HD) are computed between the predicted segmentation and ground truth segmentation of the right ventricle cavity (RVC), left ventricle cavity (LVC) and left ventricle myocardium (LVM). Table 23 presents the results using a 5-fold cross validation with the best performing models shown in bold. Also, a one-tailed paired t-test with 95% confidence level is applied to statistically measure the difference in the mean performance. The p-values for the t-test are presented in Table 24 with statistically significant results indicated in bold.

The mean performance of the 2D FCN and 3D FCN are statistically distinct in the validation set. The 2D FCN outperforms the 3D FCN in all evaluation metrics. These results might be unexpected because the 2D FCN does not integrate valuable spatial information along the z -direction. However, the cardiac dataset is highly anisotropic, having a large in-plane resolution [1.37 - 1.68 mm²] and a low resolution in the vertical axis [5-10mm]. Furthermore, some of the images exhibit a severe misalignment along the third dimension. Thus, applying an isotropic 3D kernel is ill-suited if the boundary along the z -direction is highly discontinuous and misaligned.

On the other hand, the 2D-3D FCN ensemble has an equal or better mean performance than the 2D FCN in all evaluation metrics except for the 95HD in the LVM segmentation. In the latter, the 2D FCN has a better performance with a small difference and a p-value close to 0.05. In contrast, forming the ensemble provides an improvement in the segmentation of the RVC, both in shape and spatial distribution. This achievement is particularly important because segmenting the right ventricle has been considered more challenging than the left ventricle due to the large morphological variability and ill-defined borders. Hence, we conclude that the ensemble provides an overall better performance than the individual members of the ensemble and is able to successfully integrate the information extracted by each model.

Table 23 Evaluation metrics for the predicted segmentation of the right ventricle cavity (RVC), left ventricle cavity (LVC), and left ventricle myocardium (LVM) using the optimal 2D-3D FCN ensemble, 2D FCN, and 3D FCN. The values displayed are the result from 5-fold cross validation. Best performances are indicated in bold.

Model	DSC [%]			95HD [mm]		
	RVC	LVC	LVM	RVC	LVC	LVM
2D-3D FCN ensemble	91.61±5.17	94.87±3.66	89.72±3.45	2.01±1.05	1.90±1.22	1.79±0.62
2D FCN	91.43±5.32	94.84±3.70	89.61±3.44	2.16±1.45	1.85±1.01	1.76±0.55
3D FCN	88.67±6.74	93.44±5.08	87.43±4.95	3.05±2.23	2.61±3.01	2.30±3.17

Table 24 P-values for the one-tailed paired t-test between the 2D-3D FCN ensemble, 2D FCN, and 3D FCN for the cardiac segmentation. Statistically significant p-values are indicated in bold.

Compared Models		DSC [%]			95HD [mm]		
		RVC	LVC	LVM	RVC	LVC	LVM
2D-3D FCN ensemble	2D FCN	3.6x10⁻⁴	3.4x10 ⁻¹	5.1x10 ⁻²	1.2x10⁻²	1.8x10 ⁻¹	3.7x10⁻²
2D-3D FCN ensemble	3D FCN	9.0x10⁻¹⁹	3.2x10⁻⁹	3.3x10⁻²¹	6.9x10⁻¹²	2.2x10⁻⁴	1.1x10⁻²
2D FCN	3D FCN	1.5x10⁻¹⁵	3.7x10⁻⁸	1.5x10⁻¹⁵	5.7x10⁻⁸	1.4x10⁻⁴	7.4x10⁻³

Examples of segmentations produced by the 2D FCN, 3D FCN, and 2D-3D FCN ensemble are presented in Figure 18. The 2D FCN and 3D FCN have problems primarily on the delineation of the RVC, both tend to over or under segmented the region of interest. On the other hand, the 2D-3D ensemble takes advantage of the best segmentation and provides a continuous and smooth segmentation for all substructures.

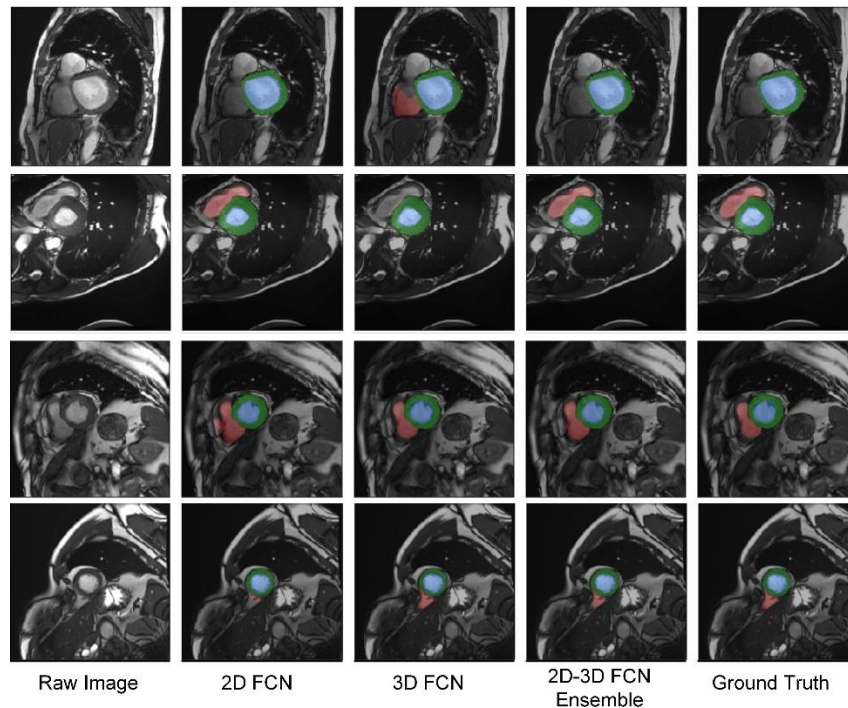


Figure 19 Examples of cardiac segmentation using the optimal 2D FCN, optimal 3D FCN, and 2D-3D FCN ensemble on the validation dataset. The red regions denote the right ventricle cavity (RVC), blue regions the left ventricle cavity (LVC), and green regions the left ventricle myocardium (LVM)

4.4.2.3 Benchmark Results

The ACDC challenge applies evaluation metrics that compare the performance of the methods from a geometrical and clinical standpoint. The geometrical metrics measure the accuracy of the segmentation and include the Dice similarity coefficient (DSC) and Hausdorff distance (HD) for the three substructures of interest on end diastolic (ED) and end systolic (ES) phases. For the clinical performance, the correlation (corr), bias, and standard deviation values are computed from the measurements of the ED volumes, ejection fractions (EF), and myocardium mass. The results of our method in the segmentation of the right ventricle cavity (RVC), left ventricle cavity (LVC), and left ventricle myocardium (LVM) on the test data are presented in Tables 25, 26 and 27, respectively. The top performing groups of the challenge, as of September 2019, are also included. Moreover, a column denoted as *Design Type* has been added to describe if the method used has been manually or automatically (auto) adjusted to the segmentation task.

The proposed AdaEn-Net is ranked within the top 8 of the challenge leaderboard in all evaluation metrics. It achieves third and second place in terms of the DSC and HD in the RVC segmentation. It ranks first and third in the HD metric in the segmentation of the LVC on ED and ES phases. Finally, it achieves third and fourth place in the HD metric in the LVM segmentation on ED and ES phases, respectively. The highest agreement with the ground truth is obtained in the segmentation of the LVC, while the lowest in the segmentation of the LVM. This behavior results from the high contrast between the left ventricle and surrounding structures, and the low contrast found in the myocardium.

Table 25 Quantitative results in the segmentation of the Right Ventricle Cavity of the proposed method (AdaEn-Net) and top competing models on the ACDC challenge dataset. Best results are indicated in bold.

Method	Design Type	DSC [%] ED	DSC [%] ES	HD [mm] ED	HD [mm] ES	EF corr	EF bias	Volume ED corr	Volume ED bias
AdaEn-Net	Auto	93.6	88.4	10.183	12.234	0.899	-2.120	0.989	2.550
Isensee [116]	Auto	94.6	90.4	8.835	11.376	0.925	-2.966	0.991	2.136
Zotti [117]	Manual	93.4	88.5	11.052	12.650	0.869	-0.872	0.986	2.372
Zotti [118]	Manual	94.1	88.2	10.318	14.053	0.872	-2.228	0.991	-3.722
Khened [45]	Manual	93.5	87.9	13.994	13.930	0.858	-2.246	0.982	-2.896
Baumgartner [119]	Manual	93.2	88.3	12.670	14.691	0.851	1.218	0.977	-2.290
Wolterink [120]	Manual	92.8	87.2	11.879	13.399	0.852	-4.610	0.980	3.596
Rohe [121]	Manual	91.6	84.5	14.049	15.926	0.781	-0.662	0.983	7.340
Patravali [122]	Manual	91.1	81.9	13.517	18.729	0.791	6.784	0.945	5.634
Grinias [123]	Manual	88.7	76.7	19.041	24.249	0.756	-0.192	0.916	11.910
Yang [124]	Manual	78.9	77.0	30.285	31.089	0.576	8.832	0.789	47.322

Table 26 Quantitative results in the segmentation of the Left Ventricle Cavity of the proposed method (AdaEn-Net) and top competing models on the ACDC challenge dataset. Best results are indicated in bold

Method	Design Type	DSC [%] ED	DSC [%] ES	HD [mm] ED	HD [mm] ES	EF corr	EF bias	Volume ED corr	Volume ED bias
AdaEn-Net	Auto	95.8	90.3	5.592	8.644	0.981	0.490	0.997	3.07
Isensee [116]	Auto	96.5	93.3	5.608	6.300	0.992	0.338	0.997	1.590
Zotti [117]	Manual	96.4	91.2	6.180	8.386	0.990	-0.476	0.997	3.746
Khened [45]	Manual	96.4	91.7	8.129	8.968	0.989	-0.548	0.997	0.576
Baumgartner [119]	Manual	96.3	91.1	6.526	9.170	0.988	0.568	0.995	1.436
Wolterink [120]	Manual	96.1	91.8	7.515	9.603	0.988	-0.494	0.993	3.046
Rohe [121]	Manual	95.7	90.0	7.483	10.747	0.989	-0.094	0.993	4.182
Zotti [118]	Manual	95.7	90.5	6.641	8.706	0.987	-1.186	0.997	9.640
Patravali [122]	Manual	95.5	88.5	8.212	10.929	0.971	1.734	0.997	9.864
Grinias [123]	Manual	94.8	84.8	8.898	12.934	0.970	-1.736	0.992	2.454
Yang [124]	Manual	86.4	77.5	47.873	53.050	0.926	1.496	0.894	12.232

Table 27 Quantitative results in the segmentation of the Left Ventricle Myocardium of the proposed method (AdaEn-Net) and top competing models on the ACDC challenge dataset. Best results are indicated in bold.

Method	Design Type	DSC [%] ED	DSC [%] ES	HD [mm] ED	HD [mm] ES	Volume ES corr	Volume ES bias	Mass ED corr	Mass ED bias
AdaEn-Net	Auto	87.3	89.5	8.197	8.318	0.988	-1.790	0.989	-2.100
Isensee [116]	Auto	89.6	91.9	7.609	7.148	0.989	-3.402	0.986	-4.053
Zotti [117]	Manual	88.6	90.2	9.586	9.291	0.980	1.160	0.986	-1.872
Khened [45]	Manual	88.9	89.8	9.841	12.582	0.979	-2.572	0.990	-2.873
Patravali [122]	Manual	88.2	89.7	9.757	11.256	0.986	-4.464	0.989	-11.586
Baumgartner [119]	Manual	89.2	90.1	8.703	10.637	0.983	-9.602	0.982	-6.861
Zotti [118]	Manual	88.4	89.6	8.708	9.264	0.960	-7.804	0.984	-12.405
Wolterink [120]	Manual	87.5	89.4	11.121	10.687	0.971	0.906	0.963	-0.960
Rohe [121]	Manual	86.7	86.9	11.536	13.034	0.955	5.130	0.967	-3.373
Grinias [123]	Manual	79.9	78.4	12.300	14.567	0.890	-1.682	0.950	-19.625

Furthermore, a good performance is achieved in the segmentation of the RVC due to the creation of the 2D-3D ensemble that provides a better shape and spatial definition as shown in the experiments from Section 4.4.2.2. We also believe that combining the volumetric and in-plane spatial information through the ensemble allowed our model to obtain among the lowest HD metrics of the competition.

In the leaderboard, nine out of the ten implemented methods are deep convolutional networks that resemble a U-Net or FCN architecture. Moreover, with the exception of the top ranked method, the networks have been manually designed for the cardiac segmentation task. Various approaches apply techniques tailored for the cardiac anatomy such as the incorporation of a cardiac shape prior [118, 117], extraction of the region of interest using a Fourier analysis technique based on the heartbeat frequency [45], and rigid alignment by using the barycenter of the

LV and RV [121]. In comparison to those approaches, our AdaEn-Net has a simple architecture and training process that can easily generalize to other segmentation tasks.

The top ranked submission by Isensee et al. [116] implemented the nnU-Net. This adaptive network is the same as previously discussed in the section on the prostate segmentation. The optimal nnU-Net for the cardiac segmentation is an ensemble of 2D and 3D U-Net architectures with 18.1×10^6 and 29.0×10^6 number of parameters, respectively. In this approach, the hyperparameters are set using manually specified rules. Hence, the results of the adaptation process can vary depending on the adequacy of the predefined rules on unseen datasets. In contrast, the AdaEn-Net dynamically adapts the architecture of the network based on the characteristics of the image and the structure being segmented while minimizing the size of the model. Furthermore, our resultant architectures are smaller and more efficient. The optimal 2D FCN has 1.5×10^6 parameters and the optimal 3D FCN has 3.3×10^6 parameters that result in a model with $10 \times$ fewer parameters than the nnU-Net.

Table 28 Quantitative results on the ACDC dataset between the proposed method (AdaEn-Net) and a current reinforcement learning based algorithm for neural architecture search of medical image segmentation architectures. The best results are indicated in bold.

Method	Optimization Method	DSC [%]			HD [mm]		
		RVC	LVC	LVM	RVC	LVC	LVM
AdaEn-Net	Evolutionary Algorithm	91.0	93.0	88.4	11.2	7.1	8.3
[66]	Policy gradient algorithm	86.8	92.8	84.9	14.3	8.9	10.7

The AdaEn-Net is also compared with a recently proposed reinforcement learning based method for neural architecture search (NAS) of medical image segmentation architectures [66]. The model searches the best set of hyperparameters of a 2D densely connected encoder-decoder baseline network using a policy gradient approach and it is also tested on the ACDC dataset. The comparison between these two methods, using the average DSC and HD as evaluation metrics, for the RVC, LVC, and LVM segmentation is shown in Table 28.

Differently from our method, in [66] the depth of the architecture is previously defined and the search process does not consider minimizing the size of the optimal architecture. Furthermore, the volumetric information is incorporated as a post-processing operation by applying a 3D fully connected conditional random field to the stacked 2D segmentations. As shown in Table 28, our network performs better than the proposed reinforcement learning based algorithm in all evaluation metrics. Moreover, their optimal 2D architecture has a total of 30×10^6 parameters, which is bigger than our 5-fold AdaEn-Net, which has $1.5 \times$ fewer parameters. In terms of computational time, the method

proposed by [66] took 10 days of continuous training with 15 TitanX GPUs. In comparison, our proposed AdaEn-Net took approximately 4.9 days for the 2D search process and 8.7 days for the 3D search process using 1 GTX 1080 GPU. Therefore, our method is also more efficient during the architecture search and construction process.

4.5 Discussion

In this work, we presented the AdaEn-Net, an ensemble of fully adaptive 2D-3D FCN for medical image segmentation. The AdaEn-Net was tested on two publically available segmentation challenges. In both datasets, the proposed model achieved a performance within the top places of the leaderboard demonstrating its success in adapting to the datasets. Our model performed better than a top-ranked ruled-based adaptive framework in the prostate dataset, while being considerably smaller, and achieved better results than a NAS reinforcement learning based algorithm on the cardiac dataset. Additionally, it achieved comparable results with state-of-the art CNNs manually tailored for the segmentation task in hand. Differently from the competing manually and automatically designed networks, the proposed AdaEn-Net automatically adapts to a particular 3D segmentation task by incorporating volumetric and in-plane information while simultaneously optimizing the network’s performance and size. Thus, our optimal models are accurate, efficient, and simple. They could easily be implemented, trained, and applied in settings with restricted computational resources.

Another important characteristic of the AdaEn-Net is its capability to simultaneously define the optimal width and depth of the architecture. Recent work has exposed that balancing a network’s width and depth to the input image resolution can significantly increase accuracy and efficiency [125]. The experiments showed that balancing the dimensions of the architecture is necessary when constructing the 3D FCN. For each dataset, the hyperparameters of the structure (number of filters, kernel sizes and number of residual blocks) for the 3D FCNs were considerably different. On the other hand, the optimal structural hyperparameters for the 2D FCNs were almost identical on both datasets. We believe these results are related to the complexity of the segmentation task and type of architecture. Since 3D FCNs need to learn more complicated relationships to directly segment volumetric images, the model’s performance is particularly sensitive to finding the appropriate depth and width. Meanwhile, the 2D FCN structure is better at generalizing to a different dataset. The depth and large kernel’s sizes in the optimal 2D FCNs shows that using long-range 2D contextual information is key for an accurate slice-wise segmentation. Nevertheless, we want to highlight that finding the optimal hyperparameters for a 2D FCN is a complex optimization problem. Although in

these problems the architecture turn out to be similar, using the same 2D FCN on another unseen dataset will not guarantee a good performance.

In terms of the individual performance of the optimal 2D FCN and 3D FCN, it is observed that the 2D FCN achieves a high segmentation accuracy in both datasets. In contrast, the 3D FCN has a lower performance in the segmentation of the cardiac structures. There are at least three factors that may influence this behavior. The first one is related to the quality of the volumetric data. As the resolution along the z -axis decreases in relation to the resolution in the x - y plane, the 3D FCN becomes less precise. This is because information of nearby slices are less relevant at predicting the current segmentation. Thus, many of the extracted features will be less meaningful and can be a source of overfitting. Second, 3D FCNs assume the scale in all directions is the same (i.e., apply isotropic kernels). Therefore, it becomes hard to learn useful features when the information density is distinct along each dimension. Lastly, the difference in performance has a relation to the complexity of the optimization problem. 3D FCNs have a bigger input, need to learn more complicated correlations to explain the output variable (segmentation), and have more decision variables (weights) to optimize. Consequently, the difficulty to reach a good local optima with a local search optimizer increases.

The AdaEn-Net takes advantage of the intra-slice and inter-slice information through the creation of the 2D-3D FCN ensemble. In both datasets, the ensemble achieved a better segmentation performance than the individual 2D FCN and 3D FCN. Specifically, in the segmentation of the prostate, the ensemble had a statistically better performance on all evaluation metrics. This result is expected because the 2D FCN as well as the 3D FCN have highly accurate predictions and their combination increases the robustness of the final prediction. In the cardiac dataset, the ensemble outperformed the individual models in the segmentation of the right ventricle cavity and had an equal performance to the 2D FCN in the segmentation of the left ventricle cavity and left ventricle myocardium. Although the 3D FCN is statistically less accurate than the 2D FCN, the final ensemble successfully incorporates the information from both architectures. Hence, taking advantage of 3D contextual information while reducing the effects of anisotropic dimensions and slice misalignments. These results also confirm an important contribution of our work, which is that the proposed ensemble framework can develop a diverse set of models with less correlated errors.

The proposed AdaEn-Net applied a multiobjective evolutionary approach to automatically design architectures for medical image segmentation. The results obtained in the segmentation of 3D structures demonstrates that this methodology is feasible and worth exploring further. Moreover, other objective functions can be considered

and incorporated into the optimization process. In both datasets, the majority of the top performing submissions employed architectures that were hand-designed for the specific segmentation task. However, neural architecture search (NAS) has recently achieved higher performance than manual architecture engineering in well-known computer vision challenges [11]. This indicates that NAS specialized in medical image segmentation is a promising area of research and the presented work contributes towards this development.

During the adaptation of the 2D and 3D FCNs, training the candidate architectures throughout evolution is the most time-consuming step. On the prostate dataset, running the MEA algorithm for 40 generations took approximately 66.31 hours for the 2D FCN and 118.08 hours for the 3D FCN. On the cardiac dataset, evolving 40 generations took approximately 116.54 hours for the 2D FCN search and 207.32 hours for the 3D FCN search. Although the time required to construct the AdaEn-Net is considerable, designing a neural network either automatically or manually is time-consuming because it inevitably requires testing candidate models to quantitatively assess the impact of the hyperparameter values. In comparison with a manual approach, our model reduces the need for expert knowledge, decreases the time a researcher is actively working on finding the optimal architecture and searches for configurations that are also efficient. Furthermore, when compared with a policy-gradient reinforcement learning NAS algorithm on the cardiac dataset, our model showed to converge faster to a more accurate and smaller network. Their search process took 10 days with 15 TitanX GPUs while our ensemble took approximately 13.6 days in total using 1 GPU. Nevertheless, one of the limitations of the AdaEn-Net is the time required for the MEA algorithm to converge to the Pareto optimal solutions. For our future work, we will improve the computational time by applying a surrogate loss function and sharing weights between candidate architectures.

Chapter 5: EMONAS-Net-Efficient Multiobjective Neural Architecture Search Framework Using Surrogate-Assisted Evolutionary Algorithm

5.1 Introduction

In this chapter, an **Efficient MultiObjective Neural Architecture Search** framework for 3D medical image segmentation named EMOAS-Net is presented. EMONAS-Net has two key components: a novel search space that considers the configuration of the micro- and macro-structure of the segmentation architecture, and a **Surrogate-assisted Multiobjective Evolutionary based Algorithm** (SaMEA algorithm) that improves the efficiency of the architectural hyperparameter search. In relation to the search space, the micro-structure or basic cell is represented by a directed acyclical graph (DAG) and the SaMEA algorithm determines the connections between the nodes and the most appropriate convolutional operation for each node. For the design of the macro-structure, the search space includes the hyperparameters that define the depth (number of decoder and encoder cells) and width (number of filters) of the architecture. As both the micro and macro-structure are being automatically designed, the proposed search space reduces the need for manual engineering or the implementation of a second-level search while allowing a better optimization of the network's size. The second component of our framework is the SaMEA algorithm that is build upon the MEA algorithm and searches for architectures that minimize the segmentation error and number of parameters of the network. In this work, we propose two strategies to improve the convergence and search time of the algorithm. First, information produced during the initial stages of the evolutionary search is used to increase the probability of selecting the best performing hyperparameter values and most promising subproblems. Secondly, a Random Forest model is applied as an inexpensive surrogate function to estimate a candidate's architecture performance and decrease the training time. The EMONAS-Net framework is tested on three medical image segmentations tasks from publically available datasets. These include the segmentation of the prostate in MR images from the MICCAI PROMISE12 challenge [81], the segmentation of the posterior and anterior parts of the hippocampus on MRI from the Medical Segmentation Decathlon challenge [126], and the segmentation of the left ventricle cavity, left ventricle myocardium and right ventricle cavity of cardiac MRI from the MICCAI ACDC challenge [106]. In all the datasets, the EMONAS-Net is able to find networks that perform better or similar to other

NAS methods, while being significantly smaller and requiring considerably less computational time for the architecture search.

5.2 Methods

The proposed EMONAS-Net is a surrogate-assisted multiobjective NAS framework that consists of two main components: the search space and the search algorithm. The search space defines the candidate architectures that can be generated given the group of unset hyperparameters and their search range. Our search space considers the configuration of the micro and macro-structure of the segmentation architecture. For the micro-structure search space, we search for the configuration of the basic cell that is repeated in the encoder and decoder path of the segmentation network. Meanwhile, for the macro-structure search space, we search for the appropriate number of encoder and decoder cells and the number of filters on each cell. The search algorithm refers to the optimization method utilized to generate the candidate architectures and the selection of the optimal solution. In this work, we propose an efficient **Surrogate-assisted Multiobjective Evolutionary based Algorithm**, SaMEA, which constructs the segmentation architectures by considering two objective functions: the expected segmentation error and the size of the network. The SaMEA algorithm is based on the previously developed MEA algorithm, but differently from the latter it incorporates information extracted from the initial generations of the evolutionary search to identify the most promising subproblems and hyperparameter values to select. Furthermore, a Random Forest surrogate model is introduced to estimate the performance of the generated candidate architectures. On the following sections, we describe the search space of our model, the Random Forest surrogate model, and the SaMEA algorithm.

5.2.1 Search Space

The general structure of the EMONAS-Net follows an encoder-decoder path with an equal number of encoder and decoder cells as shown in Figure 20 b). The encoder path receives as input the 3D medical image and extracts the most important features from the input data by applying a set of convolutional operations found in the encoder cells. Each cell in the encoder path is followed by a max-pooling operation with stride 2 that halves the size of the input feature map. This is followed by the decoder path, which receives the features provided by the encoder and doubles the size of the feature map by applying a transpose convolution. Cells on the opposite sides of the encoder-decoder path are connected through a summation operation to improve the flow of information and the gradient during backpropagation. The last layer of the decoder applies a 3D convolution with a kernel of size $1 \times 1 \times 1$ and a softmax activation function to produce the 3D segmented image. Having the general structure set, the proposed search space

includes the hyperparameters that define the configuration of the encoder-decoder cell, which we denominate the micro search space, and the hyperparameters that define the width and depth of the network that we refer to as the macro search space.

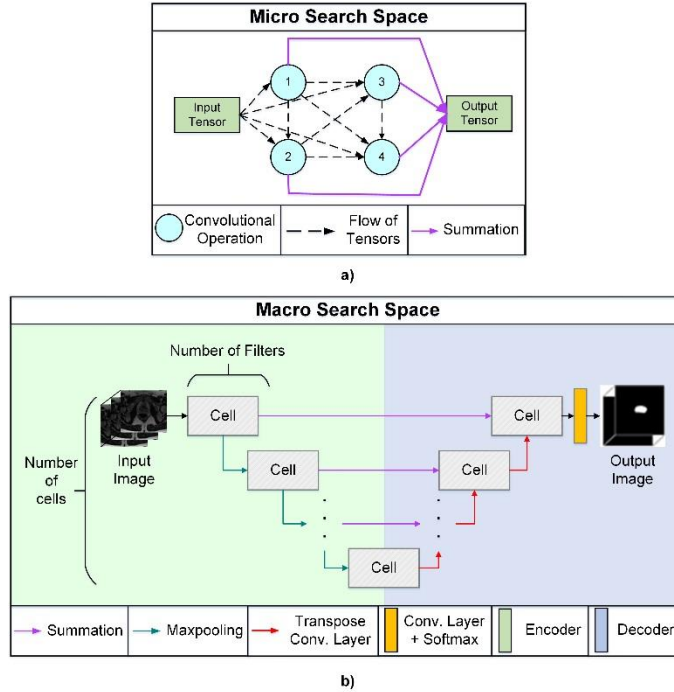


Figure 20 An overview of the micro and macro search space for the EMONAS-Net. a) The micro search space includes the decision variables that define the topology of the cell that is repeated in the encoder-decoder path. b) The general structure of the EMONAS-Net follows an encoder-decoder path.

5.2.1.1 Micro Search Space

Inspired by the idea of the micro search space proposed in [52], our micro search space includes the hyperparameters that define the topology of the cell that is to be repeated in the encoder-decoder path. The cell is represented as a directed acyl graph (DAG) that contains B nodes. Each node represents a convolutional operation applied to an input tensor, and each directed edge represents the flow of tensors between operations. In Figure 20 a) a cell with $B = 4$ nodes is shown. In the proposed work, each node $b \in B$ is characterized by two decisions variables: $i_b \in I_b$ that determines the input to node b , and $o_b \in O_b$ that specifies the type of convolutional operation applied to i_b in node b . All the possible convolutional operations, which are described in detail below, apply zero-padding and a stride of 1 on all of the three dimensions of the feature map.

- Input to node b (I_b): The set of possible inputs I_b to node b are the set of output tensors from all previous nodes in the cell plus the cell's input tensor. For example in Figure 20 a), the set of possible inputs to node 3 are the cell's input tensor, the output tensor of node 1 and the output tensor of node 2. In Figure 20 a), the

possible inputs for each node are shown as dotted arrows. In the case of node 1, which has no previous nodes in the cell, the only possible input is the cell’s input tensor. Hence, node 1 has only one decision variable that is the type of convolutional operation applied and the input is fixed to the cell’s input tensor.

- Type of convolutional operation in node b (O_b): The set of possible convolutional operations O_b that can be applied in node b is composed by 2D, 3D and Pseudo-3D convolutions and are shown in Table 28. Each convolutional operation is composed of a ReLU activation function, the respective convolutional layer (2D, 3D or P3D convolution) and an instance normalization layer [127].

Table 29 Set of possible convolutional operations in a node.

<ul style="list-style-type: none"> • $1 \times 1 \times 1$ 3D convolution • $3 \times 3 \times 3$ 3D convolution • $5 \times 5 \times 5$ 3D convolution • $3 \times 3 \times 1$ 2D convolution • $5 \times 5 \times 1$ 2D convolution • $7 \times 7 \times 1$ 2D convolution • $3 \times 3 \times 3$ Pseudo-3D convolution • $5 \times 5 \times 5$ Pseudo-3D convolution • $7 \times 7 \times 7$ Pseudo-3D convolution • identity
--

Pseudo-3D (P3D) convolutions were presented in [128] and replace a standard $k \times k \times k$ 3D convolutional layer with a sequence of one $k \times k \times 1$ convolutional layer and one $1 \times 1 \times k$ convolutional layer. This type of convolution has demonstrated to have a good learning capacity on anisotropic medical images [129] and can significantly reduce the number of trainable parameters in the network. Hence, our set of convolutional operations allows the proposed NAS framework to leverage between in-plane information captured by 2D convolutions, volumetric information obtained by costly 3D convolutions, and exploiting inter-slice and intra-slice information from anisotropic images with the P3D convolutions. Moreover, we have added the identity function as a possible operation, which returns as output the node’s input tensor without any change. This operation allows the cell to reduce the number of nodes if necessary and also the formation of skip connections inside the cell. Searching between 2D, 3D and P3D convolutional operations was previously presented in [68]. However, differently from their fixed cell structure and $3 \times 3 \times 3$ kernel size, our work adds flexibility to the construction of the cell by searching for the best connection between nodes, and permits the selection of the kernel sizes of each convolutional operation.

Finally, the output of the cell is the summation of the output tensor of all nodes. We have selected the summation of the feature maps as the combination operator because past work has shown that NAS algorithms prefer addition over concatenation [52, 13] and also reduces GPU memory usage. The number of nodes in a cell is a

hyperparameter that requires high computational resources to test. Therefore, following the most recent work of NAS in medical image segmentation architectures [67, 69], the cell is composed of $B=4$ nodes. The hyperparameters for the micro search space are encoded into seven decision variables. Four decision variables encode the type of convolutional operation applied to each of the 4 nodes (o_b), while the remaining three decision variables encode the input to nodes 2, 3 and 4 (i_b). As previously mentioned, the input to node 1 is fixed to be the cell's input tensor.

5.2.1.2 Macro Search Space

The macro search space includes the hyperparameters that define the depth and width of the global architecture as shown in Figure 20 b). The depth is determined by the number of cells on the encoder and decoder path, while the width is decided by the number of filters on each cell.

- Number of cells (N_{cells}): The general structure of the EMONAS-Net has an equal number of encoder and decoder cells. Hence, the number of cells N_{cells} on the architecture is computed by $N_{cells} = 2n_c + 1$, where $n_c \in [2,3,4]$. When constructing the encoder-decoder network, n_c cells are part of the encoder path, n_c cells form the decoder path and 1 cell connects the encoder-decoder path.
- Number of filters on cell (NF_i): To determine the number of filters on cell i (NF_i), where $i \in [1,2, \dots, N_{cells}]$, we utilize the common heuristic to double the number of filters whenever the size of the feature map has been halved, and to halve the number of filters whenever the size of the feature map has been doubled. Therefore, the number of filters for each cell on the architecture can be computed after determining the number of filters on the first encoder cell NF_1 and the location of the cell in the network. The number of filters for cell i (NF_i) is determined by Equation (23): Here $NF_1 = 2^{n_f}$, $n_f \in [3,4,5]$ to obtain a symmetric architecture

$$NF_i = \begin{cases} NF_1 * 2^{i-1} & \text{for } i \in [1,2, \dots, \frac{N_{cells}}{2} + 0.5] \\ NF_1 * 2^{N_{cells}-i} & \text{Otherwise} \end{cases} \quad (23)$$

5.2.1.3 Decision Variables in the Search Space

In summary, the micro- and macro-structure of the EMONAS-Net is encoded into a vector with 10 decision variables denominated the hyperparameter vector. The first seven decision variables define the composition of the encoder-decoder cell, the following two decision variables determine the depth and width of the macro architecture, and the last decision variable is the learning rate used to train the network. The learning rate is included because it is the most important hyperparameter to set when training and it needs to be calibrated each time the architecture

changes. The 10 decision variables that make up the search space are presented with their corresponding search range in Table 30. Meanwhile, an example of a hyperparameter vector as well as the decoded EMONAS-Net architecture are shown in Figure 21.

Table 30 The 10 decision variables that constitute the hyperparameter search space for generating the EMONAS-Net architecture and their corresponding search range.

Decision Variable	Formula	Search Range
Input to node 2 (I_2)	-	[Input tensor, node 1]
Input to node 3 (I_3)	-	[Input tensor, node 1, node 2]
Input to node 4 (I_4)	-	[Input tensor, node 1, node 2, node 3]
Type of Convolutional operation in node 1 (O_1)	-	Refer to Table 1.
Type of Convolutional operation in node 2 (O_2)	-	Refer to Table 1.
Type of Convolutional operation in node 3 (O_3)	-	Refer to Table 1.
Type of Convolutional operation in node 4 (O_4)	-	Refer to Table 1.
Number of cells (N_{cells})	$2n_c + 1$	$n_c \in [2,3,4]$
Number of filters for NF_1	2^{n_f}	$n_f \in [3,4,5]$
Learning rate	-	$[1 \times 10^{-6}, 9 \times 10^{-3}]$

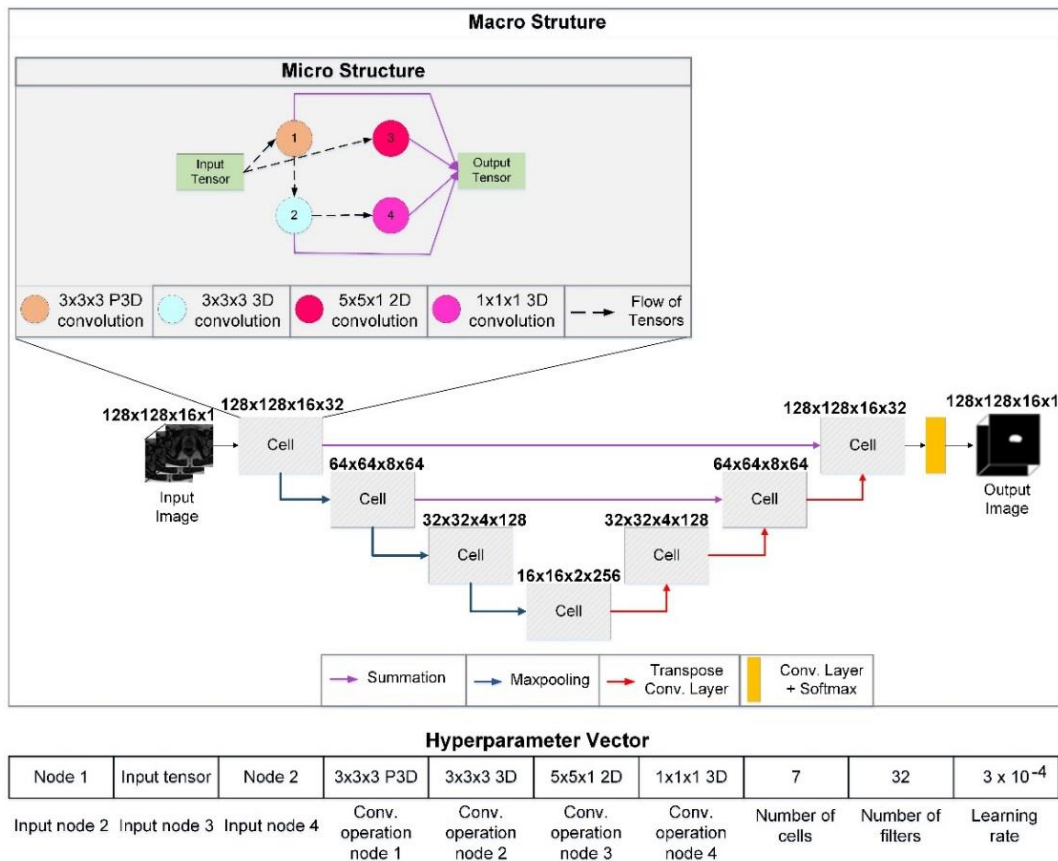


Figure 21 Example of a hyperparameter vector with the 10 decision variables that define the micro- and macro-structure of the EMONAS-Net and the decoded architecture.

5.2.2 Search Algorithm

In this section, we describe the **S**urrogate **a**ssisted **M**ultiobjective **E**volutionary based **A**lgorithm (SaMEA), which searches for the best architecture in the proposed search space. In the following sub-sections, we present the optimization problem the SaMEA algorithm solves, the hyperparameter and subproblem selection probabilities used to guide the search, the Random Forest surrogate model implemented for the performance prediction and the algorithm.

5.2.2.1 Problem Statement

The SaMEA algorithm solves a multiobjective hyperparameter optimization problem. In this problem, the aim is to find the optimal hyperparameters for a model that minimize a set of M objective functions $F(x) = [f_1(x), \dots, f_M(x)]$. Specifically in this work, the interest is to design an architecture that minimizes both the segmentation error and the number of parameters in the network. Meanwhile, as presented in Table 30, the hyperparameters are the 10 decision variables that define the micro- and macro-structure of the EMONAS-Net.

For the first objective function, the expected segmentation error function proposed in our previous work [104] is applied. The expected segmentation function is measured by using the multi-class Dice coefficient and defined in Equation (24).

$$MCDice(\theta) = \sum_c \frac{2 \sum_i \hat{y}_{ic}(\theta) y_{ic}}{\sum_i \hat{y}_{ic}(\theta) + \sum_i y_{ic}} \quad (24)$$

where C is the number of segmentation classes or substructures being segmented in the image. $\hat{y}_{ic}(\theta)$ are the voxels predicted to be part of the segmentation class $c \in C$, θ are the weights/parameters learned by the EMONAS-Net network during training, and y_{ic} the voxels from the ground truth segmentation that are part of class $c \in C$. The $MCDice$ coefficient has a value between 0 and C , where values close to C indicate a significant overlap between the predicted segmentation and the ground truth segmentation on all classes. Hence, the segmentation error can be quantified by $C - MCDice$. The function $C - MCDice$ is refer as the multi-class Dice loss.

The expected segmentation error (ESE) function has three terms that are the multi-class Dice loss in the training set, the multi-class Dice loss in the validation set, and the distance to the training epoch with the maximum validation multi-class Dice coefficient. The first term measures the network's capacity to learn significant features from the training data. The second term assesses the network's ability to generalize well to an unseen dataset. Since we apply the strategy of partial training during evolution to reduce the training time, the third term helps determine if

the network may have an improvement after being fully trained. The expected segmentation error function is shown in Equation (25).

$$ESE = \alpha(C - MCDice_{Train}(\theta)) + (C - MCDice_{Val}(\theta)) + \beta \left(\frac{E - e_{max}}{E} \right) \quad (25)$$

where $MCDice_{Train}$ is the multi-class Dice coefficients in the training set and $MCDice_{Val}$ the multi-class Dice coefficients in the validation set. E is the maximum number of epochs the candidate architectures are trained for during evolution and e_{max} is the number of epoch with the maximum validation multi-class Dice coefficient. α and β are weight parameters whose value ranges between [0, 1] and helps balance the importance of the validation multi-class Dice coefficient over the other two terms.

For the second objective function, the logarithm of the number of parameters in the network is minimized. A logarithmic transformation is applied to reduce the high variability between architecture sizes (the smallest architecture generated can have thousands of parameters while the biggest architecture tens of millions of parameters). The transformation also promotes the algorithm to search for different architectures and not to prematurely discard big models from the population. The multiobjective hyperparameter optimization problem solved by the SaMEA algorithm is shown in Equations (26), (27) and (28).

$$\text{Minimize } f_1(x) = \alpha(C - MCDice_{Train}(\theta)) + (C - MCDice_{Val}(\theta)) + \beta \left(\frac{E - e_{max}}{E} \right) \quad (26)$$

$$\text{Minimize } f_2(x) = \log(|\theta|) \quad (27)$$

$$\text{subject to } x \in \Omega \quad (28)$$

where x is the hyperparameter vector with the 10 decision variables that define the EMONAS-Net architecture. Ω represents the search space, θ are the parameters the EMONAS-Net learns during training, and $|\cdot|$ is the cardinality operator.

5.2.2.2 Guiding the Search with Selection Probabilities

The SaMEA algorithm builds upon the MEA algorithm to approximate the set of Pareto optimal solutions by applying the MOEA/D algorithm [77]. However, the SaMEA algorithm introduces what we denominate the selection probabilities to address two limitations of the MEA algorithm and improve the efficiency of the architecture search.

First, the MEA algorithm relies on crossover and random mutation to generate the new individuals during evolution. However, by analyzing the segmentation performance of previously tested candidate architectures, a correlation can be found between hyperparameter values and architecture performance. Therefore, using random

mutation can be inefficient at finding good performing segmentation architectures, and neglects important and costly information obtained in previous generations. In this work, we propose using a hyperparameter mutation probability that increases the probability of selecting hyperparameter values that have shown a good performance on the tested architectures and reduces the probability of selecting hyperparameter values that have obtained a poor performance. To compute the hyperparameter mutation probabilities, each hyperparameter value in the search space is scored using the average expected segmentation error function (ESE) presented in Equation (25). Let $S_{h_{ij},G}$ be the score for the hyperparameter decision variable i with value j at generation G , where i refers to a hyperparameter decision variable (i.e., number of cells, convolutional operation, input to node b) and j refers to a specific value assigned to that decision variable (i.e., if the hyperparameter decision variable is the number of cells, then the possible hyperparameters values are 5, 7 and 9). Then, $S_{h_{ij},G}$ is computed using Equation (29):

$$S_{h_{ij},G} = \sum_{g=1}^{G-1} \frac{I(\hat{f}_g) * [ESE_{max} - ESE(\hat{f}_g)]}{I(\hat{f}_g)} \quad (29)$$

where $ESE(\hat{f}_g)$ is the expected segmentation error of candidate architecture \hat{f}_g tested in generation g . ESE_{max} is the maximum expected segmentation error that any architecture can obtain and is described in Equation (30).

$$ESE_{max} = \alpha * C + C + \beta \quad (30)$$

where α and β are the weight parameters used in the ESE objective function and C is the number of segmentation classes. The term $[ESE_{max} - ESE(\hat{f}_g)]$ measures the distance between the expected segmentation error of the candidate architecture \hat{f}_g and the maximum attainable error, which increases as the segmentation performance of the candidate architecture \hat{f}_g improves. Meanwhile, $I(\hat{f}_g)$ is an indicator function that has a value of 1 if the hyperparameter i with value j (h_{ij}) was applied to construct the candidate architecture \hat{f}_g and 0 otherwise. $I(\hat{f}_g)$ is shown in Equation (31).

$$I(\hat{f}_g) = \begin{cases} 1 & \text{if } h_{ij} \text{ is used to construct candidate architecture } \hat{f}_g \\ 0 & \text{otherwise} \end{cases} \quad (31)$$

A higher $S_{h_{ij},G}$ means that the hyperparameter value j produces architectures with a smaller expected segmentation error. Finally, the probability of mutating to hyperparameter value j at generation G ($P_{h_{ij},G}$) is defined in Equation (32).

$$P_{h_{ij},G} = \frac{PS_{h_{ij},G}}{\sum_{j \in J} PS_{h_{ij},G}} \quad (32)$$

where

$$PS_{h_{ij},G} = \frac{S_{h_{ij},G}}{\sum_{j \in J} S_{h_{ij},G}} + \varepsilon \quad (33)$$

J is the set that contains all the values of the hyperparameter decision variable i . To make sure all hyperparameter values have a positive probability of being selected ($P_{h_{ij},G} > 0$) regardless of whether they have the maximum ESE ($[ESE_{max} - ESE(\hat{f}_g)] = 0$) or have not been previously selected to construct a candidate architecture ($1(\hat{f}_g)=0$), $PS_{h_{ij},G}$ is first computed in Equation (33) where an $\varepsilon = 0.002$ value is added. Then, Equation (32) assures that the sum of all the probabilities of the hyperparameter values in set J add to 1. As with the score, $P_{h_{ij},G}$ increases as the average expected segmentation error of the hyperparameter h_{ij} decreases.

The second limitation of the MEA algorithm is determining the subproblems to be selected for solving the MOP. When approximating the Pareto Front, the MOP is decomposed into a number of scalar optimization subproblems by applying a nonlinear weighted aggregation function to the M objective functions $F(x) = [f_1(x), \dots, f_M(x)]$. In this work, a Penalty based Boundary Intersection Approach (PBI) aggregation method is applied after confirming through experimentation that it provides a better set of solutions than Tchebycheff or weighted sum approach. During evolution, each subproblem is solved once in a generation and the number of subproblems solved is the same as the size of the population (N). However, previous studies have shown that some subproblems discover more non-dominated points during the search process than others [130, 131, 132]. Therefore, to exploit the most promising search regions, subproblems that have actively contributed to approximate the Pareto Front or provide a diverse set of solutions must have a higher probability of being selected to be solved and can be solved more than once in a generation. For this aim, we implement the subproblem selection probabilities proposed by [130], which uses the NSGA-II selection scheme [133]. This selection scheme is applied because it considers the quality of the solutions both in terms of convergence and diversity. Thus, the probability of choosing subproblem $n \in N$ at generation G ($P_{SP_{n,G}}$) is computed with Equation (34).

$$P_{SP_{n,G}} = \frac{D_{n,G}}{\sum_{n=1}^N D_{n,G}} \quad (34)$$

where $D_{n,G}$ is the proportion of good solutions generated by subproblem n during the previous $G - 1$ generations, and N is the number of subproblems. Furthermore, the proportion of good solutions generated by subproblem n in the previous $G - 1$ generations is computed using Equation (35).

$$D_{n,G} = \frac{\sum_{g=1}^{G-1} GS_{n,g}}{\sum_{n=1}^N \sum_{g=1}^{G-1} GS_{n,g}} + \varepsilon \quad (35)$$

where $GS_{n,g}$ is the number of good solutions generated by subproblem n at generation g when using the NSGA-II selection scheme. Similar to the hyperparameter mutation probabilities, an $\varepsilon = 0.002$ is applied when calculating the proportion of good solutions to ensure all subproblems have a positive probability of being selected in a generation ($P_{SP_{n,G}} > 0$) even if they have not produced any good solution in the past generations ($\sum_{g=1}^{G-1} GS_{n,g} = 0$).

For an evolutionary algorithm to be successful, it must establish a good ratio between exploration and exploitation of the search space. The presented selection probabilities are an exploitation strategy because they guide the search towards the most promising areas based on the information collected in the previous generations. Therefore, it is important to implement the selection probabilities after a diverse set of candidate solutions have been tested during evolution to avoid convergence to a local minima solution. However, if the implementation is performed too late during the search, the information produced is not effectively used and time to convergence remains unchanged. To allow the algorithm to initially explore the different areas from the hyperparameter search space and discover how each subproblem contributes to the approximation of the Pareto Front, during the initial LG generations all the hyperparameter values have an equal probability of being selected ($P_{h_{ij,G}}$ has a uniform distribution) and each subproblem is solved once in a generation ($P_{S_{n,G}}$ is not computed). After the LG generations, the selection probabilities are computed according to Equations (32) and (34), and updated after each remainder generation to foment the convergence to a good solution. In this work, we tested the proposed algorithm with $LG = 8, 10, 12, 14$ and 16 on the prostate MRI dataset in a total of 40 generations. $LG=10$ was selected because it improved the convergence speed without degrading the performance of the best solution found.

5.2.2.3 Random Forest Surrogate Model

Since the computational cost of training a segmentation network is very high, it is impractical to train all the candidate architectures generated during evolution. We therefore propose the use of a Random Forest surrogate model to approximate an architecture's expected segmentation error without the need of training it. A Random Forest model is adopted because of its applicability to discrete and categorical input variables, excellent approximation properties when a limited number of training samples is available and robustness to the hyperparameter setting [134]. Additionally, compared with other commonly used performance predictors such as Gaussian Processes and Neural Networks, the training and implementation of a Random Forest model is much cheaper in terms of runtime and computational cost.

The proposed Random Forest model fits Tr regression trees to n data points randomly drawn with replacement from the training set $\{(\delta_1, f_1(x_1)), \dots, (\delta_n, f_1(x_n))\}$, where $\delta_i = (\delta_{i,1}, \dots, \delta_{i,d})$ represents the hyperparameter configuration of network x_i and $f_1(x_i)$ the network's performance. In this work, the network's performance $f_1(x_i)$ is quantified by the expected segmentation error defined in Equation (25). Meanwhile, the hyperparameter configuration δ_i is made up of 11 components. The first 10 components are the 10 decision variables in the hyperparameter vector that define the EMONAS-Net architecture (shown in Table 29), and the last component is the logarithm of the number of parameters in the architecture. The latter was added as it reduced the model's Mean Square Error, Mean Absolute Error, and increased the R^2 score. For each tree, a minimum number of n_{min} data points are required to split a node. We set the number of regression trees to $Tr=100$ and the minimum number of data points for split to $n_{min}=5$ after using a random search method for the hyperparameter selection.

Given a new configuration δ_{n+1} , the Random Forest surrogate model aims to predict its true performance $f_1(x_{n+1})$. The Random Forest predicted performance $\hat{f}_1(x_{n+1})$ is computed as the mean of the individual trees' predictions, shown in Equation (36).

$$\hat{f}_1(x_{n+1}) = \frac{\sum_{t=1}^{Tr} \hat{f}_{1,t}(x_{n+1})}{Tr} \quad (36)$$

where $\hat{f}_{1,t}(x_{n+1})$ is the predicted expected segmentation error of regression tree t , and Tr are the total number of trees in the Random Forest model. Furthermore, we also calculate the prediction's dispersion $\sigma(\hat{f}_1(x_{n+1}))$ by computing the standard deviation of the individual trees' predictions as presented in Equation (37).

$$\sigma(\hat{f}_1(x_{n+1})) = \sqrt{\frac{\sum_{t=1}^{Tr} [\hat{f}_{1,t}(x_{n+1}) - \hat{f}_1(x_{n+1})]^2}{Tr}} \quad (37)$$

The prediction's dispersion quantifies the uncertainty of the prediction, which grows as the new point goes farther from the data available in the training set. We use this value in the SaMEA algorithm to select the most promising architectures to train, which is later described in Section 5.2.2.4.

Similar to the implementation of the selection probabilities, during the initial LG generations all candidate architectures generated are trained using the stochastic gradient descent algorithm. The information obtained in these initial learning generations is used to populate the Random Forest training set. After the LG generations, the surrogate is applied to predict the expected segmentation error of the candidate architectures and retrained as new points are added to the training set.

5.2.2.4 SaMEA Algorithm

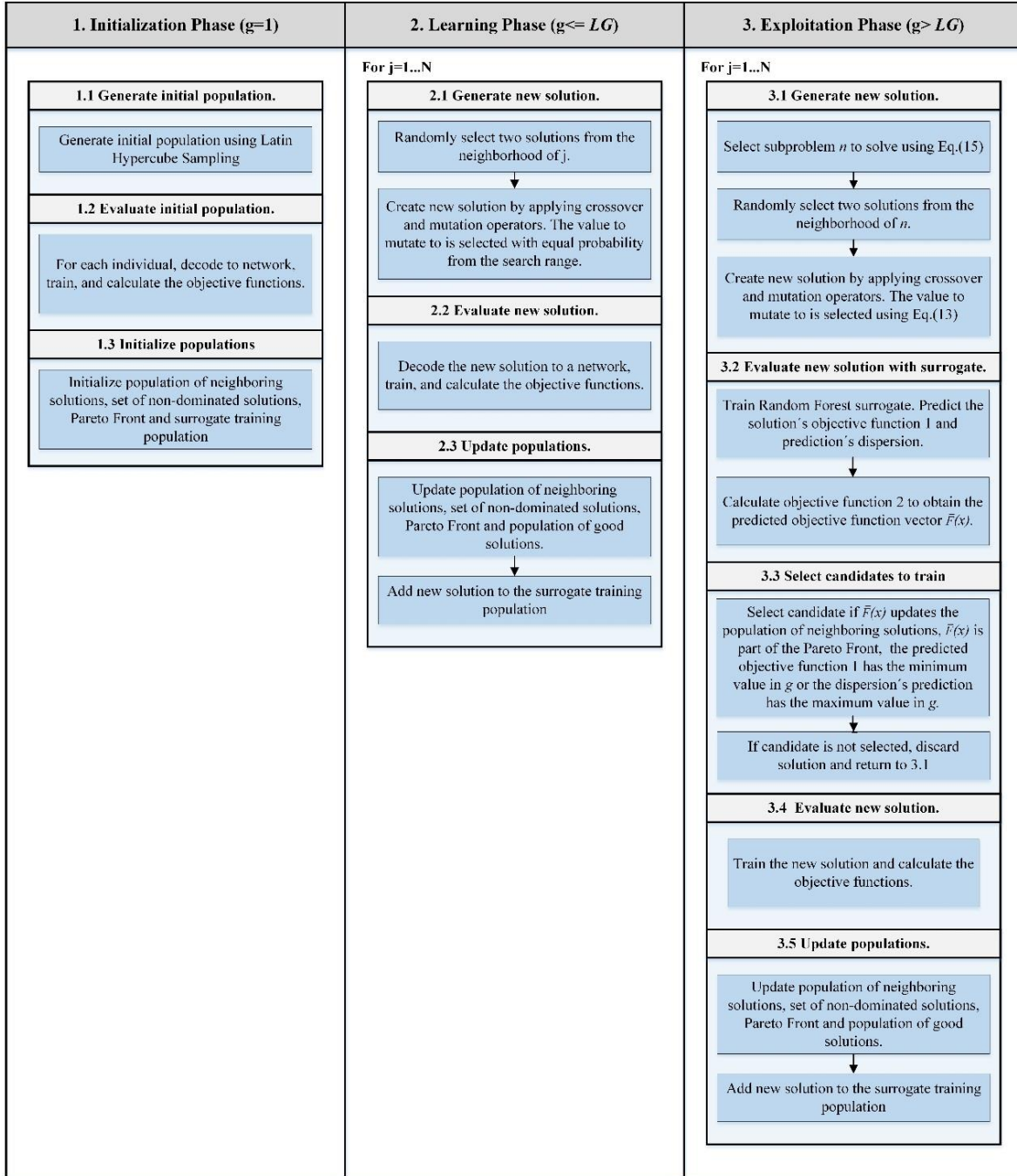


Figure 22 Flowchart of the search process applied by the proposed SaMEA algorithm

The pseudo code of the SaMEA algorithm is shown in Algorithm 2, while a flowchart of the search process is presented in Figure 22. The algorithm is divided into three phases: 1) Initialization phase 2) Learning phase ($g \leq LG$), and 3) Exploitation phase ($g > LG$). In the initialization phase, the initial population is generated as well as the sets containing the best solutions initialized. In the learning phase, the algorithm gathers information from the

optimization problem by testing all the scalar subproblems in a generation, randomly exploring the different areas from the hyperparameter search space, and training all the candidate architectures generated. Meanwhile in the exploitation phase, the information produced in the previous two phases is used to guide the selection of the subproblems and hyperparameter values, and the Random Forest surrogate applied to choose the candidate architectures to train.

1. *Initialization Phase:*

a. *Generate initial population:* The algorithm starts by generating an initial population of size N . This is achieved by using a Latin hypercube sampling (LHS) method to sample the hyperparameter vectors for each individual. LHS aims to distribute samples evenly over the feasible set, which allows our algorithm to test the different regions in the search space.

b. *Evaluate initial population:* The initial population is evaluated by decoding the hyperparameter vectors into the corresponding EMONAS-Net architectures and training them using the ADAM optimizer and the multi-class Dice loss as the loss function. Then, the objective functions $F(x) = [f_1(x), f_2(x)]$ defined in Equations (26) and (27) are calculated for each architecture.

c. *Initialize populations:* This initial set of solutions and objective function values are saved in the set of non-dominated solutions (NDS) and Pareto Front, respectively. Also, these solutions are used to initialize the surrogate training population (STP) for the Random Forest surrogate fitting.

2. *Learning Phase:*

a. *Generate new solutions:* After initialization, the search process evolves for G generations. During the initial LG generations ($LG < G$) each subproblem $j \in \{1, 2, \dots, N\}$ is solved once in a generation. This is accomplished by randomly selecting two solutions from the neighborhood of j ($B(j)$) and applying crossover and mutation operators to generate a new solution. When applying the mutation operator, the hyperparameter value to mutate to is chosen from all the possible values in the search range with an equal probability.

b. *Evaluate the new solution:* The new solution is decoded into the corresponding network, trained and evaluated using the objective functions $f_1(x)$ and $f_2(x)$ described in Equations (26) and (27), respectively.

c. *Update populations:* The population of neighboring solutions is updated using the PBI approach. The set of NDS and Pareto Front is also updated if the new solution dominates any point in the set of NDS . Finally, all the

solutions and objective function values obtained in the *LG* generations are included in the *STP* population used to train the surrogate model.

3. Exploration Phase:

a. Generate new solutions: After the *LG* generations, the selection probabilities are implemented to guide the search. First, in each generation, N subproblems are solved by selecting a subproblem according to the probability defined in Equation (34). For the selected subproblem n , two solutions are randomly chosen from $B(n)$ and a new solution obtained by applying crossover and mutation operators. During mutation, the hyperparameter value selected to mutate is based on the probability presented in Equation (32).

b. Evaluate new solution with the surrogate: The new solution is decoded into a candidate EMONAS-Net architecture. Using the *STP* population, the Random Forest surrogate is trained and used to predict the expected segmentation error objective function $\hat{f}_1(x)$ and the prediction dispersion $\sigma(\hat{f}_1(x))$. The second objective function $f_2(x)$, found in Equation (27), is also calculated to obtain the predicted objective function vector $\hat{F}(x) = [\hat{f}_1(x), f_2(x)]$.

c. Select candidates to train. The candidate EMONAS-Net architecture is selected to be trained if any of the four criteria explained below is met. On the other hand, if none of the four criteria is satisfied, the new solution generated is discarded and the algorithm selects another subproblem to solve.

The criteria to select a candidate EMONAS-Net model to train is the following: 1) The predicted objective function vector $\hat{F}(x) = [\hat{f}_1(x), f_2(x)]$ updates the population of neighboring solutions using the PBI approach. 2) None of the points in *NDS* dominates the predicted objective function vector $\hat{F}(x) = [\hat{f}_1(x), f_2(x)]$ and thus it is predicted to be a non-dominated solution and part of the Pareto Front. 3) The predicted expected segmentation error $\hat{f}_1(x)$ has the minimum value in the generation. Hence, it is the most competitive architecture produced during that generation in terms of the segmentation accuracy. 4) The prediction dispersion $\sigma(\hat{f}_1(x))$ has the maximum value in the generation. This indicates that the Random Forest Surrogate had the highest uncertainty when predicting the performance of that EMONAS-Net architecture.

Only solutions that have been trained are used to update any population in the algorithm or selection probabilities. This ensures that there is no error when ranking the architectures and that the Pareto Front is truly approximated. Therefore, the first criteria makes sure that the population of neighboring solutions is kept updated to

correctly guide the creation of new solutions. The second criteria tests any solution that might be part of the Pareto Front so that our algorithm closely approximates the real Pareto Front and no solution is missed. The third criteria allows the most competitive architectures produced in a generation to be evaluated and used to update the populations that guide the search. Finally, the fourth criteria aims to reduce the surrogate's prediction error by including in the training set points that lie in regions for which the model is still uncertain.

d. Evaluate new solution: If the new solution was selected according to the four criteria, it is decoded into the corresponding network and trained. The true objective function vector $F(x) = [f_1(x), f_2(x)]$ is calculated.

e. Update populations: The population of neighboring solutions of subproblem n is updated using the PBI approach, and the set of *NDS* and Pareto Front is updated. The new solution and objective function values are included in the *STP* population.

After the G generations, the SaMEA algorithm returns the set of non-dominated solutions and Pareto Front. Since the main objective is to obtain an accurate segmentation, the non-dominated solution that has the smallest expected segmentation error is selected as the best solution. Although we are not choosing a solution that minimizes the network size, the inclusion of the second objective function forces the algorithm to search and include small architectures in the population of neighboring solutions. Hence, when new solutions are generated from subproblems that prioritize the segmentation accuracy, they tend to be both efficient and accurate as they also inherit hyperparameter values from the smaller architectures.

A summary of all the parameter values used to implement the SaMEA algorithm are presented in Table 31. The maximum number of generations G is set to 40 as it showed to be enough to allow the algorithm to converge to a good solution. Furthermore, the mutation operator is applied independently to each decision variable in the hyperparameter vector with a probability that decreases monotonically in each generation g according to the formula presented in Table 31. The population size is set to a small value because the aim is to find one efficient and good performing segmentation architecture. However, to keep a diverse population of neighboring solutions, a big neighborhood size relative to the population size is chosen. Finally, after testing the penalty factor values Θ of 0.001, 0.01, 0.05, 0.1 and 1 on the prostate MRI dataset, a value $\Theta = 0.001$ was selected because it provided a diverse and good performing population of neighboring solutions.

Table 31 Summary of parameter values used to implement the SaMEA algorithm.

Parameter Name	Parameter Value
Generations (G)	40
Population size (N)	10
Neighborhood size(T)	3
Learning Generations (LG)	10
Penalty factor (Θ)	0.001
Crossover probability	0.50
Mutation probability	$\max\left(\left[1 - \frac{\ln(g)}{\ln(40)}\right], \frac{1}{10}\right)$
Number of trees for surrogate (Tr)	100
Minimum number of data points for split (n_{min})	5

Algorithm 2: Surrogate assisted Multiobjective Evolutionary based Algorithm (SaMEA)
<p>Input:</p> <ul style="list-style-type: none"> • A MOP with M objective functions and a feasible set Ω. • N: population size in each generation. • $\lambda^1, \lambda^2, \dots, \lambda^N$: a set of N weight vectors where $\lambda^i \in \mathbb{R}^M$. • T: Neighborhood size. • LG: The number of initial generations the algorithm runs without applying the selection probabilities or surrogate prediction. • A termination criteria: Maximum number of generations G. <p>Output:</p> <ul style="list-style-type: none"> • Best EMONAS-Net architecture: the solution with smallest expected segmentation error in the set of non-dominated solutions (NDS). <p>1. Initialization</p> <p>1.1 For each weight vector λ^j, $j \in \{1, 2, \dots, N\}$, obtain the T closest weight vectors $\lambda^{j_1}, \lambda^{j_2}, \dots, \lambda^{j_T}$ by computing the Euclidean distance between any two vectors. Set the neighborhood of λ^j, $B(j) = \{j_1 \dots j_T\}$ as the T closest weight vectors.</p> <p>1.2 Use LHS method to generate the initial population $P = \{x_1^1, \dots, x_1^N\}$ from Ω.</p> <p>1.3 Decode the population into N EMONAS-Net architectures and train. Calculate the M objective functions for each architecture $F(x_1^j) = [f_1(x_1^j), \dots, f_M(x_1^j)] \forall j \in \{1, 2, \dots, N\}$.</p> <p>1.4 Set the surrogate training population $STP = \{([x_1^1, f_2(x_1^1)], f_1(x_1^1)), \dots, ([x_1^N, f_2(x_1^N)], f_1(x_1^N))\}$ where $f_1(x_1^j)$ and $f_2(x_1^j)$ represents the expected segmentation error and the logarithm of the number of parameters of architecture x_1^j respectively. Set the population with N good solutions using the NSGA-II selection scheme $A = \{x_1^1, \dots, x_1^N\}$, $NDS = \{x_1^1, \dots, x_1^N\}$, $Pareto\ Front = \{F(x_1^1), \dots, F(x_1^N)\}$, and generation $g=1$</p> <p>1.5 Uniformly initialize the hyperparameter mutation probabilities.</p> <p>1.6 Set the ideal point $z^* = \{z_1^*, \dots, z_M^*\}$, where $z_m^* = \rho [\min\{f_m(x_1^1), \dots, f_m(x_1^N)\}] \forall m \in \{1, 2, \dots, M\}$ and $\alpha \in [0, 1)$.</p> <p>2. Evolution</p> <p>For all $g=1 \dots G$ do:</p> <p style="padding-left: 20px;">2.1 Generate new solution</p> <p style="padding-left: 40px;">If $g \leq LG$ then:</p> <p style="padding-left: 60px;">For all $j=1 \dots N$ do:</p> <p style="padding-left: 80px;">2.1.1 Reproduction: Randomly select two solutions x_g^k and x_g^l from $B(j)$. Generate a new solution x_g^j by applying crossover and mutation operators. Use the uniformly initialized hyperparameter mutation probabilities to select the hyperparameter value to mutate to. Set $Sp=j$.</p> <p style="padding-left: 80px;">2.1.2 Evaluate candidate model: Decode x_g^j into a EMONAS-Net architecture and train. Calculate the objective functions $(x_g^j) = [f_1(x_g^j), \dots, f_M(x_g^j)]$.</p> <p style="padding-left: 40px;">End If</p> <p>End For</p>

ElseIf $g > LG$ **then:**

For all $j = 1 \dots N$ **do:**

2.1.3 Select subproblem: Select subproblem $n \in \{1, 2, \dots, N\}$ to solve using the subproblem selection probability defined in Equation (34). Set $Sp = n$.

2.1.4 Reproduction: Randomly select two solutions x_g^k and x_g^l from $B(n)$. Generate a new solution x_g^j by applying crossover and mutation operators. Use the hyperparameter mutation probability defined in Equation (32) to select the hyperparameter value to mutate to.

2.1.5 Evaluate candidate model using the surrogate: Decode x_g^j into a EMONAS-Net architecture. Use STP to train the Random Forrest surrogate and predict the expected segmentation error $\hat{f}_1(x_g^j)$ and dispersion of the prediction $\sigma(\hat{f}_1(x_g^j))$. Calculate the other $M - 1$ objective functions to obtain $\hat{F}(x_g^j) = [\hat{f}_1(x_g^j), \dots, f_M(x_g^j)]$.

2.1.6 Select candidate models to train: Train x_g^j if $\hat{F}(x_g^j)$ is predicted to update the population of neighboring solutions, $\hat{F}(x_g^j)$ is predicted to be part of the *Pareto Front*, $\hat{f}_1(x_g^j)$ has the minimum predicted value in generation g or $\sigma(\hat{f}_1(x_g^j))$ has the maximum value in generation g . Otherwise discard the solution and return to 2.1.3.

2.1.2 Evaluate candidate model: If x_g^j has not been discarded, set $F(x_g^j) = [f_1(x_g^j), \dots, f_M(x_g^j)]$.

End ElseIf

2.2 Update populations

For all $j = 1 \dots N$ **do:**

If x_g^j **has not been discarded do:**

2.2.1 Update the surrogate training population: Set $STP = STP \cup ([x_g^j, f_2(x_g^j)], f_1(x_g^j))$.

2.2.2 Update the population of neighboring solutions: For each index $w \in B(Sp)$, where Sp is the subproblem that generated x_g^j , replace x_g^w with x_g^j and $F(x_g^w)$ with $F(x_g^j)$ if $g^{pbi}(x_g^j | \lambda^w, z^*) \leq g^{pbi}(x_g^w | \lambda^w, z^*)$.

2.2.3 Update set of non-dominated solutions NDS and Pareto Front: If none of the points in *NDS* dominates $F(x_g^j)$, add $F(x_g^j)$ to *Pareto Front* and x_g^j to *NDS*. Remove from *Pareto Front* and *NDS* all non-dominated points and solutions dominated by $F(x_g^j)$.

2.2.4 Update ideal point: If $\rho[f_m(x_g^j)] < z_m^*$ then set $z_m^* = \rho[f_m(x_g^j)] \quad \forall m \in \{1, 2, \dots, M\}$.

2.2.5 Update population of good solutions A: Select the best N solutions using the NSGA-II selection criteria and update A .

End For

Return the solution with the smallest expected segmentation error in *NDS*

5.3 Experiments

The proposed EMONAS-Net is tested on three 3D medical image segmentation tasks. The first task is the segmentation of the prostate in anisotropic MR images from the MICCAI PROMISE12 challenge [81]. The second task is the segmentation of the anterior and posterior parts of the hippocampus in MR images from the Medical Segmentation Decathlon challenge [126]. Finally, the third task is the segmentation of the left ventricle cavity, left

ventricle myocardium and right ventricle cavity on highly anisotropic cardiac MR images from the MICCAI ACDC challenge [106]. The EMONAS-Net framework is implemented with Keras library and Python 3.6, using one 8-GB GeForce GTX 1070 Ti GPU. In the following sections, we present the benchmark datasets, implementation details and results. Additionally, we also analyze the search efficiency of the proposed framework on the prostate MRI dataset.

5.3.1 Prostate MR Image Dataset (PROMISE12 Challenge)

5.3.1.1 Dataset Details

The PROMISE12 challenge provides a dataset composed of 50 transverse T2-weighted MR training images of the prostate with their corresponding ground truth segmentation, and 30 unlabeled testing images. The images were obtained from 4 medical centers, with differences in scanner manufacturer, field of strength and acquisition protocol. Hence, there is high variation in terms of image size, resolution, anatomic appearance, and position. The scans were acquired for prostate cancer detection or staging purposes. Since the images have a heterogeneous voxel spacing, they are resampled to have a voxel spacing of $1\text{mm}\times 1\text{mm}\times 1.5\text{mm}$ and set to a fixed size of $128\times 128\times 64$ voxels. Furthermore, all intensity values are clipped to be inside the 3 standard deviations from the mean and rescaled to a $[0,1]$ range in a slice-wise manner.

5.3.1.2 Implementation Details

For the implementation of the SaMEA algorithm, the training dataset set is split into a training and validation set. 80% of the training images, corresponding to 40 cases, are randomly selected to form the training set while the remainder 10 cases make up the validation set. The algorithm runs for a total of 40 generations, using the parameters shown in Table 31. When computing the expected segmentation error objective function, defined in Equation (26), the weight parameters applied are $\alpha = 0.25$ and $\beta = 0.25$. Furthermore, each candidate architecture is partially trained for 120 epochs. Patches of size $128\times 128\times 16$ voxels are used for training the candidate architectures. These parameters follow the recommendations of [104]. The SaMEA algorithm runs for approximately 3.02 days on 1 GPU and selects 194 candidate architectures to be trained during evolution. A total of 7 non-dominated solutions are found. The approximate Pareto Front and architectures tested during the search are presented in Figure 23. This graph shows the trade-off between the two objective functions. Usually, the smaller architectures have the worst performance in terms of the expected segmentation error. On the other hand, the architectures with sizes between 5×10^6 parameters and 1×10^7 parameters have the lowest segmentation error. It is interesting to observe that the network's segmentation

accuracy starts to deteriorate for models with more than 1×10^7 parameters. This is mostly caused by an overfitting of the training set. This observation highlights the importance of considering both efficiency and accuracy when designing an architecture, and that in some datasets simply fine-tuning an already constructed architecture may not provide the best results. The selected best solution has a total of 5.9×10^6 parameters, and a multiclass Dice loss of 0.27 in the training set and 0.32 in the validation set. The hyperparameters of the best EMONAS-Net architecture are shown in Table 32, while a representation of the final architecture is presented in Figure 21.

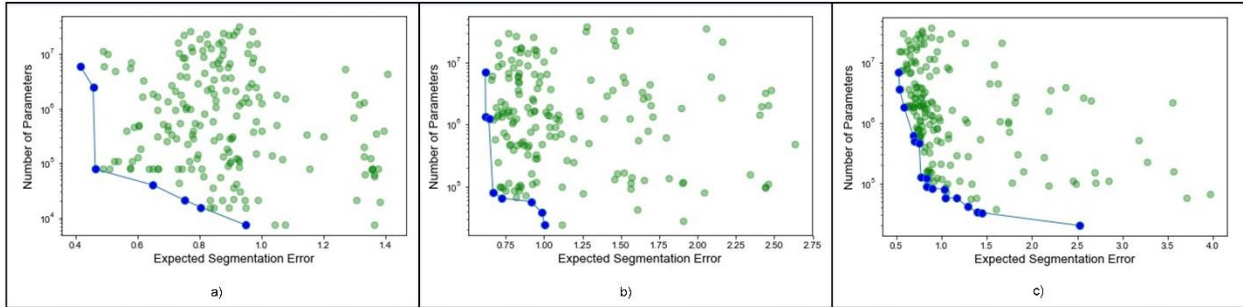


Figure 23 Trained architectures (green points) and approximate Pareto Front (blue points) found with the SaMEA algorithm. a) Criterion space obtained with the prostate MRI dataset. b) Criterion space obtained with the hippocampus MRI dataset. c) Criterion space obtained with the cardiac MRI dataset.

Table 32 Best hyperparameter values and size of the EMONAS-Net architecture on the Prostate, Hippocampus and Cardiac MRI datasets.

Decision Variable	Prostate Dataset	Hippocampus Dataset	Cardiac Dataset
Input to node 2 (I_2)	Node 1	Input Tensor	Input Tensor
Input to node 3 (I_3)	Input Tensor	Node 1	Node 1
Input to node 4 (I_4)	Node 2	Node 1	Node 3
Type of Convolutional operation in node 1 (O_1)	$3 \times 3 \times 3$ P3D	$3 \times 3 \times 3$ 3D Conv.	$3 \times 3 \times 1$ 2D Conv.
Type of Convolutional operation in node 2 (O_2)	$3 \times 3 \times 3$ 3D Conv.	$3 \times 3 \times 3$ P3D	$3 \times 3 \times 3$ P3D
Type of Convolutional operation in node 3 (O_3)	$5 \times 5 \times 1$ 2D Conv.	$3 \times 3 \times 3$ P3D	$1 \times 1 \times 1$ 3D Conv.
Type of Convolutional operation in node 4 (O_4)	$1 \times 1 \times 1$ 3D Conv.	$5 \times 5 \times 1$ 2D Conv.	$7 \times 7 \times 1$ 2D Conv.
Number of Cells (N_{cells})	7	7	7
Number of filters for NF_1	32	32	32
Learning Rate	3×10^{-4}	4×10^{-4}	4×10^{-4}
Number of Parameters	5.9×10^6	7.0×10^6	7.1×10^6

In the cell, the flow of information is not linear as most CNNs for medical image segmentation. Two branches are formed, the first branch is composed by nodes 1-2-4 and the second only by node 3. Interestingly, branch one includes 3D type convolutions while branch 2 a 2D convolution. We believe this selection of inputs allows each branch of the cell to specialize in the extraction of inter-slice or intra-slice information, and then by fusing the output of each through the addition function a result similar to a 3D-2D ensemble is obtained. Furthermore, a bigger kernel size is

selected for the 2D convolutional operation, which allows the second branch to have a wider in-plane field of view without an excessive computational cost.

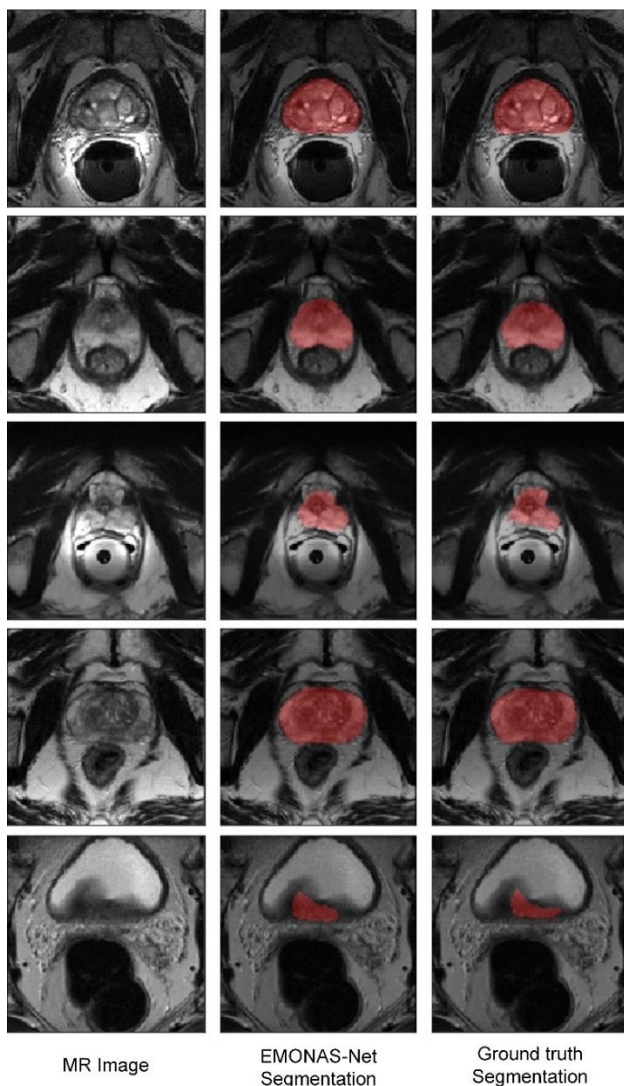


Figure 24 Visualization of the segmentation results on the prostate dataset using the best EMONAS-Net architecture. The prostate region is denoted in color red.

The best EMONAS-Net architecture for the prostate dataset is trained for 6000 epochs using patches of size $128 \times 128 \times 16$ voxels and the multi-class Dice loss function. One epoch represent the forward-backward pass of one patch from all training images. This number of epochs has been selected after confirming that no improvement can be attained with further training. The ADAM optimizer is applied with the parameter values suggested in [76]. The learning rate found with the SaMEA algorithm is used for training. Moreover, the learning rate decays by a factor of 0.5 if no improvement is obtained on the validation loss after 30 subsequent epochs of training. Data augmentation is applied by transforming the original image in a slice-wise manner with 90° range rotations, 0.5 range scaling, 0.4

range horizontal and vertical translation, and horizontal flips. The weights are initialized using the He Normal initialization [111] and the weights with the minimum validation loss used for segmenting the test images. A largest connected component analysis is applied as a post-processing operation to obtain the final segmentations. A visual illustration of the segmentation results in the validation set are presented in Figure 24. The EMONAS-Net is able to successfully segment regular and irregular prostate shapes and provide a continuous contour.

5.3.1.3 Benchmark Results

In order to evaluate the effectiveness and efficiency of the proposed EMONAS-Net framework, we compare the best EMONAS-Net architecture against the top performing auto machine learning (AutoML) approaches applied to the PROMISE12 challenge as of August 2020. To make an informed and fair comparison, we have included only results from submissions that have provided a methodology description on the challenge webpage or submitted it to a preprint server. The segmentation results of the best EMONAS-Net architecture are evaluated by submitting the segmentation of the 30 unlabeled test images to the PROMISE12 challenge website. For each submission, the average Dice similarity coefficient (DSC) and 95 percentile Hausdorff distance (95HD) are reported for the whole prostate segmentation as well as for the apex and base parts of the prostate. In Table 33, we present the models along with the optimization method applied, number of GPU days to find the best architecture, evaluation metrics, and size of the architecture. In Table 34 the evaluation metrics for each method is presented. A dashed line means that the information was not reported

Table 33 Description of top competing AutoML approaches on the PROMISE12 challenge dataset.

Model	Optimization Method	GPU days	Number of Parameters
EMONAS-Net	Evolutionary	3.02	5.9×10^6
AdaEn-Net [135]	Evolutionary	7.68	27.5×10^6
nnU-Net (I) [114]	Rule-based	-	365.5×10^6
nnU-Net (II) [114]	Rule-based	-	365.5×10^6
nnU-Net (III) [114]	Rule-based	-	365.5×10^6

Table 34 Evaluation metrics for the EMONAS-Net and top competing AutoML approaches on the PROMISE12 challenge dataset.

Model	DSC[%]			95 HD [mm]		
	Whole	Apex	Base	Whole	Apex	Base
EMONAS-Net	91.01	88.16	89.50	4.40	3.69	4.37
AdaEn-Net [135]	91.45	88.10	89.54	4.08	3.58	4.43
nnU-Net (I) [114]	91.61	91.61	90.29	4.00	3.79	4.05
nnU-Net (II) [114]	91.56	91.56	89.59	4.17	3.77	4.42
nnU-Net (III) [114]	91.93	88.82	91.93	3.95	3.69	4.21

There are two competing AutoML approaches. The AdaEn-Net proposes a NAS framework that applies the evolutionary-based MEA algorithm to adapt a 2D-3D FCN ensemble. The submission is made using a two-level ensemble of five 2D-3D FCN ensemble networks. Meanwhile, the nnU-Net is a rule-based framework that automatically tunes and trains a group of U-Net architectures, and selects the best performing network or ensemble of networks. Similar to the AdaEn-Net, the final segmentation is produced with a five-model ensemble of the selected best network/ensemble. The nnU-Net makes three different submissions to the challenge and according to the method description selects an ensemble of a 2D and 3D U-Net architecture in all of them. As it is shown, the nnU-Net (III) has the best performance in most evaluation metrics but also has a significantly larger number of parameters.

Focusing on the NAS methods, the EMONAS-Net has a competitive segmentation performance against the AdaEn-Net and reduces the search time by 60.7%. Furthermore, differently from the competing AutoML methods that use a total of 10 networks for the final submission, the EMONAS-Net only applies one network that exploits the 2D-3D information by the different convolutional operations inside the basic encoder-decoder cell. This difference is visible in the number of parameters of the network, where the EMONAS-Net has $62\times$ fewer parameters than the nnU-Net and $4.7\times$ fewer parameters than the AdaEn-Net. Hence, the EMONAS-Net is a better method in terms of both efficiency and performance.

5.3.1.4 Search Efficiency

One of the main contributions of the EMONAS-Net is proposing a Surrogate-assisted Multiobjective NAS framework for medical image segmentation that exploits prior information to improve convergence and search speed. To evaluate the search efficiency of the EMONAS-Net, we compared it against two multiobjective NAS frameworks developed for medical image segmentation: the AdaEn-Net [104] and the MEA algorithm [71]. Additionally, we also compare our method against a uniform random sampling search. The EMONAS-Net has three main components, these are the novel micro and macro search space, the inclusion of selection probabilities to guide the search, and the Random Forest surrogate for performance prediction. To analyze the impact of each of these components, we compare the performance in terms of segmentation accuracy and search efficiency between the following model versions: 1.EMONAS-Net framework, 2.EMONAS-Net search space with the MEA algorithm (EMONAS-SS+MEA), 3.EMONAS-Net search space with the MEA algorithm and the Random Forrest surrogate (EMONAS-SS+MEA+RFS), 4.EMONAS-Net search space with the MEA algorithm and the selection probabilities (EMONAS-

SS+MEA+SProbs), 5.EMONAS-Net search space with random search (EMONAS-SS+RandomSearch), and 6.AdaEn-Net on the prostate MRI dataset.

The methods that apply the EMONAS-Net search space and an evolutionary optimization algorithm (models: 1, 2, 3, 4) solve the MOP defined in Equations (26)-(28) and are implemented for 40 generations using the same parameters displayed in Table 31 when applicable. The solution that has the smallest expected segmentation error in the Pareto Front is selected for the comparison. For the method that applies the EMONAS-Net search space with the random search, we run 400 iterations that produce the same number of trained architectures as 40 generations and select the architecture with the smallest expected segmentation error. All the top architectures are trained for 6000 epochs using the same training parameters detailed in section 5.3.1.2. In the case of the AdaEn-Net, we present the 5-fold cross-validation values published in [104]. The segmentation accuracy of the best architectures are evaluated by computing the Dice similarity coefficient (DSC), sensitivity, 95 percentile Hausdorff distance (95 HD), and average boundary distance (ABD) between the predicted segmentation and ground truth segmentation. In Table 35, these evaluation metrics are presented using a 5-fold cross-validation. Also, a one-tailed paired t-test with 95% confidence level is applied to compare the mean performance between the models. The best mean values are shown in bold. Meanwhile, to analyze the search efficiency in Table 36, we present the number of GPU days it took each method to run the 40 generations (or 400 iterations) on a 8 GB-GPU, the number of architectures trained during the search and the iteration in which the best solution was found.

Table 35 Segmentation accuracy metrics with the different search methods. Best results are shown in bold after applying a one-tailed paired t-test with 95% confidence level.

Method	DSC [%]	Sensitivity [%]	95HD [mm]	ABD [mm]	Number of Parameters
1. EMONAS-Net	90.49±2.92	90.84±5.50	2.97±0.93	0.93±0.29	5.9 x10 ⁶
2. EMONAS-SS+MEA	90.28±3.00	90.60±5.30	3.11±1.27	0.97±0.35	7.4 x10 ⁶
3. EMONAS-SS+MEA+RFS	90.00±3.13	90.22±5.53	3.11±0.96	0.99±0.30	4.9 x10 ⁶
4. EMONAS-SS+MEA+SProbs	90.33±2.91	91.09±4.72	3.02±0.99	0.95±0.29	4.2 x10⁶
5. EMONAS-SS+RandomSearch	89.21±3.29	90.45±6.14	3.37±0.96	1.07±0.32	10.3 x10 ⁶
6. AdaEn-Net [104]	90.42±2.87	90.13±5.93	3.07±1.08	0.92±0.28	5.5 x10 ⁶

In regards to the segmentation accuracy, the EMONAS-Net framework, the EMONAS-Net search space with the MEA algorithm, and the EMONAS-Net search space with the MEA algorithm and selection probabilities have the leading performance in all evaluation metrics. Following close is the AdaEn-Net, with a small decrease in terms of the sensitivity. As explained before, the AdaEn-Net applies the MEA algorithm to find the best architectures of an ensemble of 2D and 3D FCN. These results show that when only one 2D-3D FCN ensemble of the AdaEn-Net is compared against a single EMONAS-Net network, the EMONAS-Net provides a better or similar performance.

Moreover, the proposed EMONAS-Net search space allows the best network to successfully leverage inter-slice and intra-slice information as well as a 2D-3D ensemble. Hence, there is no need to construct two independent networks to exploit long-range 2D contextual and 3D volumetric information, but rather permit the search algorithm to select the most appropriate type of convolutional operation within the cell.

Table 36 Search efficiency metrics with the different search methods

Method	GPU days	Number of Architectures Trained	Iteration of Best Solution
1. EMONAS-Net	3.02	194	188
2. EMONAS-SS+MEA	5.29	400	217
3. EMONAS-SS+MEA+RFS	3.26	192	329
4. EMONAS-SS+MEA+SProbs	5.05	400	186
5. EMONAS-SS+RandomSearch	9.20	400	235
6. AdaEn-Net [104]	7.68	640	<i>3D FCN: 310 - 2D FCN: 189</i> ¹

Within the methods that utilize an evolutionary optimization algorithm, the EMONAS-Net search space with the MEA algorithm and the Random Forest Surrogate has the lowest performance. This result can be expected because a lower number of architectures are being trained during the search and used to update the populations and Pareto Front. Moreover, an optimal or near optimal solution might be missed due to a prediction error. However, when we implement the surrogate model with the selection probabilities in the EMONAS-Net framework, the guiding provided by the latter allows the search algorithm to reach a solution with a high segmentation accuracy on all evaluation metrics. This result also demonstrates the necessity of including both strategies in the framework to improve the search efficiency and accuracy. On the other hand, the architecture obtained with the EMONAS-Net search space and random search has the worst performance from all the methods and is also the biggest, showing that the SaMEA algorithm and MEA algorithm are able to outperform this baseline.

When analyzing the search efficiency, the methods that apply the Random Forest surrogate model (EMONAS-Net and EMONAS-SS+MEA+RFS) have the best performance in relation to the computational time and number of architectures trained during evolution. In comparison with the method that uses the EMONAS-Net search space and the MEA algorithm, the EMONAS-Net framework provides a reduction of 42.9% of the search time and 51.5% of the number of architectures trained. The reduction achieved against the AdaEn-Net is more significant. The EMONAS-Net framework decreases the search time by 60.7%, and the number of architectures trained by 69.7%. This efficiency gain is obtained while keeping a similar or better performance in the segmentation accuracy. Finally,

¹ We present the iterations in which the optimal architecture for the 3D FCN and 2D FCN search were found.

utilizing random search with the EMONAS-Net search space takes the longest. Since the random search method uniformly samples the hyperparameter values for the candidate architectures, a uniform number of big and small architectures are trained. Meanwhile, the multiobjective evolutionary algorithms tend to focus on smaller architectures, which reduces the training time.

Finally, in relation to the iteration in which the best solution is found, the EMONAS-Net framework and the EMONAS-Net search space with the MEA algorithm and selection probabilities have the best performance. Both methods converge to the best architecture in generation 18. Hence demonstrating that the information produced during the initial generations can help exploit the most promising search areas faster. In conclusion, the EMONAS-Net framework with all of its components provides the most efficient method to obtain the best performing segmentation architectures.

5.3.2 Hippocampus MR Dataset (Medical Segmentation Decathlon)

5.3.2.1 Dataset Details

The Medical Segmentation Decathlon provides 260 3D T1-weighted MR images of the hippocampus and their corresponding ground truth segmentation of the anterior and posterior parts of the hippocampus. The images were taken from the Vanderbilt University Center with a Philips Achieva scanner. The image size ranges between scans but all have a voxel size of 1mm^3 . We set the images to a fixed size of $56 \times 38 \times 44$ voxels, clip all pixel values within the 3 standard deviations of the mean and rescale them to a 0-1 range.

5.3.2.2 Implementation Details

The hippocampus dataset is randomly split into 80% and 20% for training and validation, where 208 images constitute the training set and 52 images the validation set. The parameters applied for the implementation of the SaMEA algorithm were presented in Table 31. As with the prostate dataset, the weight parameters for the expected segmentation error are $\alpha = 0.25$ and $\beta = 0.25$ and the candidate architectures are partially trained for 120 epochs. A patch of size $48 \times 32 \times 16$ voxels is applied for training to make sure the feature map size remains an integer value independent of the depth of the candidate architecture. During evolution, 174 candidate architectures are selected to be trained and it takes approximately 4.22 days to run the 40 generations. The resulting Pareto Front consist of 8 non-dominated solutions. The Pareto Front and candidate architectures tested during evolution are presented in Figure 23. In comparison with the criterion space of the prostate dataset, a bigger cluster of architectures have a smaller segmentation error. This could be caused by the increase on the number of training images, which allows the candidate

architectures to better learn the task, or that the hippocampus dataset is easier to segment and most of the architectures successfully learn to extract the best features. Moreover, when the architectures surpass the 1×10^5 number of parameters, the trade-off between number of parameters and expected segmentation error is not strongly defined. In this size range (size $> 1 \times 10^5$), both big and small architectures perform similarly well. This is one of the advantages of using a multiobjective NAS framework, it allows the researcher to see the whole Pareto and select the best solution according to the accuracy requirements and hardware restriction. For this work, the Pareto solution with the smallest expected segmentation error is selected, which has 7×10^6 parameters and a multiclass Dice loss of 0.62 in the training set and 0.46 in the validation set.

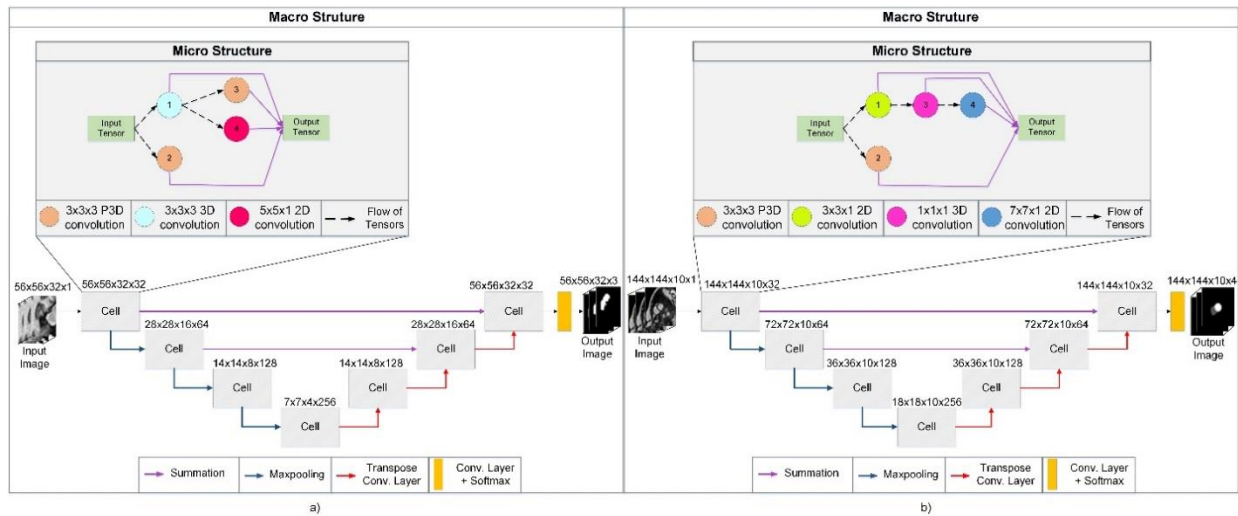


Figure 25 Configuration of the best EMONAS-Net architectures found with the proposed NAS framework. a) Best EMONAS-Net architecture on the hippocampus MRI dataset. b) Best EMONAS-Net architecture on the cardiac MRI dataset.

The hyperparameters of the best architecture for the hippocampus dataset are presented in Table 32 and the representation of the final architecture in Figure 25 a). The configuration of the cell is non-linear, having two input branches and three output branches. Differently from the prostate dataset, the branches mix both 2D and 3D convolutional operations. Since the hippocampus dataset is isotropic, we believe the 2D and 3D isotropic kernels used in the convolutional operations can successfully extract relevant features from any input tensor and there is no need for a branch to specialize only on inter-slice or intra-slice information. On the other hand, the 3D and P3D convolutional operations apply a kernel of size $3 \times 3 \times 3$ while the 2D convolutional implements a bigger kernel of size $5 \times 5 \times 1$. This shows that the 2D convolutions tend to contribute to the cell with a wider intra-slice field of view.

The best EMONAS-Net architecture is trained for 4000 epochs using a patch of size $56 \times 32 \times 32$ voxels and a batch size of 2. The size of the batch is selected considering the maximum available memory. The optimizer, learning

rate decay, data augmentation, weight initialization and largest connected component post-processing applied are the same as described for the prostate dataset in Section 5.3.1.2. The qualitative results of the best EMONAS-Net architecture on the validation set are shown in Figure 26. Our network is able to accurately segment the irregular shapes and discontinuous areas of the posterior and anterior parts of the hippocampus.

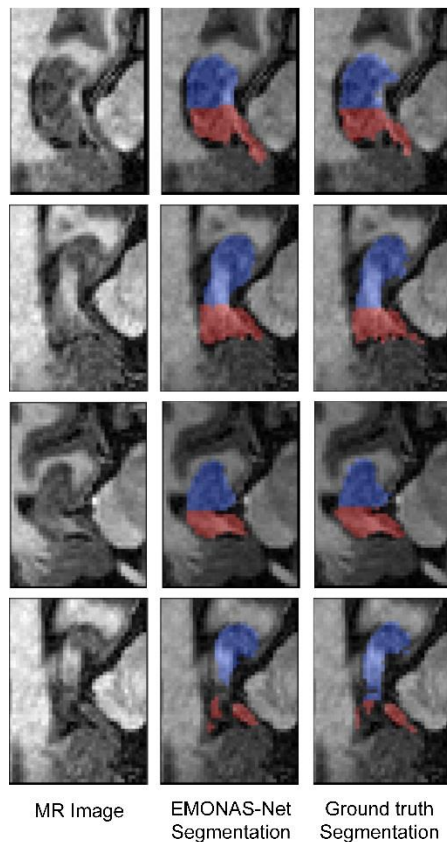


Figure 26 Visualization of the segmentation results on the hippocampus dataset using the best EMONAS-Net architecture. The red regions denote the anterior part of the hippocampus and the blue regions the posterior part.

5.3.2.3 Benchmark Results

The EMONAS-Net is evaluated by comparing its performance against architectures generated by two AutoML methods and architectures manually designed for the segmentation task. The metric used for the evaluation of the segmentation performance is the Dice similarity coefficient (DSC) between the predicted segmentations of the anterior (ANT) and posterior (POS) parts of the hippocampus and the ground truth segmentations. The results reported by the AutoML methods, including our results, use a 5-fold cross-validation. Meanwhile, the authors of the manually designed architectures divide the dataset into a training, validation and test set and report the results on their test set. Moreover, to further confirm the contributions of the proposed SaMEA algorithm over the MEA algorithm, we also apply the MEA algorithm to the EMONAS-Net search space (EMONAS-SS+MEA) and report the results of the best

architecture using a 5-fold cross-validation. Table 37 presents the design type for the architectures, optimization method and number of GPU days to find the best architecture. Table 38 shows the evaluation metric and size of the architectures. A dashed line means that there is no available or reported data.

Table 37 Description for the EMONAS-Net and competing architectures on the hippocampus dataset.

Model	Design Type	Optimization Method	GPU days
EMONAS-Net	Automatic	Evolutionary Alg.	4.22
EMONAS-SS+MEA	Automatic	Evolutionary Alg.	8.77
C2FNAS [136]	Automatic	Evolutionary Alg. + One-shot with uniform sampling	-
nnU-Net [114]	Automatic	Rule-based	-
NDN [137]	Manual	NA	NA
3D U-Net [137]	Manual	NA	NA
U-Seg-Ensemble-Net [137]	Manual	NA	NA

Table 38 Evaluation metrics for the EMONAS-Net and competing automatically and manually designed architectures on the hippocampus dataset.

Model	DSC [%]		Number of Parameters
	ANT	POS	
EMONAS-Net	89.08±0.39	87.54±0.34	7.0×10^6
EMONAS-SS+MEA	88.53±0.36	87.02±0.34	5.6×10^6
C2FNAS [136]	83.09	82.97	17.0×10^6
nnU-Net [114]	89.87	88.20	-
NDN [137]	87.93	88.69	-
3D U-Net [137]	87.48	84.92	-
U-Seg-Ensemble-Net [137]	85.75	88.29	-

In relation to the anterior part segmentation, the EMONAS-Net architecture is ranked second, surpassing the performance of all manually designed architectures and the architectures generated by the C2FNAS and MEA algorithm. For the posterior part segmentation, the EMONAS-Net performs better than the well-known 3D U-Net and the architectures obtained with the C2FNAS and MEA algorithm. Although the other two manually designed architectures have a higher accuracy in this specific segmentation task, it comes at the cost of being less generalizable and requiring expert knowledge to fine-tune their hyperparameters to new datasets.

When compared with AutoML methods, the EMONAS-Net is ranked second after the nnU-Net and has the leading performance against the NAS approaches both in accuracy and search efficiency. The C2FNAS applies a two-stage optimization process. In the first stage, the macro-level structure of the architecture is constructed using an evolutionary algorithm, while in the second stage the micro-level structure is defined by applying a single-path one-shot method with uniform sampling. The architecture is constructed using a pancreas dataset and it takes 5 days with 64 NVIDIA V100 16-GB GPUs to complete the first stage optimization and 1.4 days with 8 16-GB GPUs to run the

second optimization stage. Our method produces a better segmentation result with a smaller architecture and optimizes the micro- and macro-structure in a single search, which significantly reduces the optimization time while using only one 8GB-GPU. An increased search efficiency is also visible against the MEA algorithm, where the proposed SaMEA algorithm reduces the search time by 51.8% and the number of trained architectures by 56.5%.

5.3.3 Cardiac MR Dataset (ACDC Challenge)

5.3.3.1 Dataset Details

The ACDC dataset is comprised of 4D cardiac cine-MR images from 150 patients, from which 100 images with their corresponding manual reference are provided for training and 50 unlabeled images for testing. The aim is to segment the right ventricle cavity (RVC), left ventricle cavity (LVC), and left ventricle myocardium (LVM). The scans are from patients categorized into five medical groups: dilated cardiomyopathy, hypertrophic cardiomyopathy, myocardial infarction, abnormal right ventricle and normal patients. The cine-MR images were acquired with a conventional SSFP sequence, where a series of short axis slices cover the left ventricle from the base to the apex with a slice thickness of 5 to 10 mm and sometimes an inter-slice gap of 5mm. Meanwhile, the in-plane spatial resolution ranges from 1.37 to 1.68mm²/pixel. There is also a variation in terms of the number of volumes acquired per patient to cover completely or partially one cardiac cycle, ranging from 28 to 40 volumes. We resample the MR images to 1.56×1.56×10 mm³ per voxel and set to a fixed size of 192×192×10 voxels. The preprocessing operations implemented are the same as the ones applied for the two other datasets. We truncate the pixel values within 3 standard deviations of the mean and rescale them to a 0-1 range in a slice-wise manner.

5.3.3.2 Implementation Details

The cardiac dataset is divided into a training and validation set, where images from 80 patient are used for training and images from 20 patients for validation. The SaMEA algorithm is implemented for 40 generations using the parameters from Table 31. Following the recommendations of [104], an $\alpha = 0.25$ and $\beta = 0.10$ weight parameters are applied for the expected segmentation error objective function. Additionally, the candidate architectures are partially trained for 60 epochs using crops of size 144×144×10. The SaMEA algorithm takes 11.43 days to run and selects 186 candidate architectures to train. The approximate Pareto Front, composed of 17 non-dominated solutions, and trained candidate architectures are presented in Figure 23. The criterion space shows that the best performing architectures, in terms of the expected segmentation error, have between 2×10^6 and 3×10^7 parameters. Differently from the other datasets, where after a maximum number of parameters the performance of the

architectures start to deteriorate, in this criterion space almost all of the big architectures trained have a good segmentation accuracy. Hence, the SaMEA algorithm selects a large cluster of big architectures for training which causes the run time to increase in comparison with the other datasets. However, this selection also positively impacts the diversity of the approximate Pareto Front. The SaMEA algorithm finds a greater number of non-dominated points, especially having more solutions that favor the segmentation accuracy. The best architecture has 7.1×10^6 number of parameters with a 0.40 multiclass Dice validation loss and 0.49 multiclass Dice training loss.

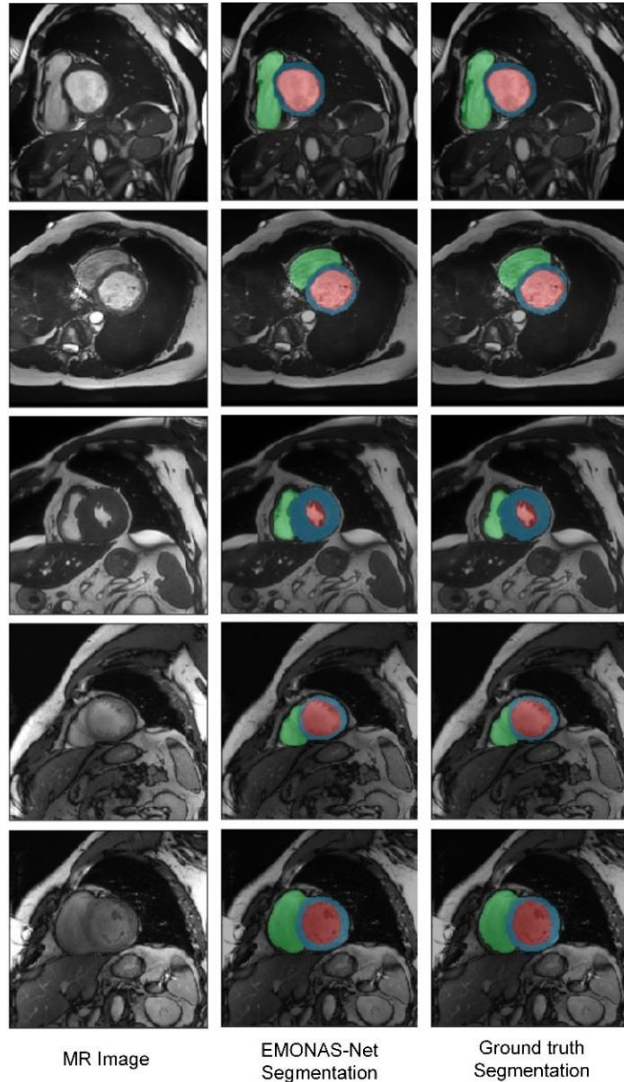


Figure 27 Visualization of the segmentation results on the cardiac dataset using the best EMONAS-Net architecture. The red region denotes the LVC, the blue region the LVM and the green region the RVC.

The hyperparameters for the best architecture for the cardiac dataset are presented in Table 32 and the configuration of the architecture in Figure 25 b). The best encoder-decoder cell has two parallel branches, one branch is composed of three nodes that apply 2D convolutions and the second branch has only one node with a P3D

convolution. This 2D-3D ensemble-like composition is similar to the composition of the cell found for the prostate dataset. However, since the cardiac dataset is highly anisotropic, having a low resolution in the vertical axis (5-10mm) and a larger resolution in x-y axis (1.37-1.68mm²), applying 3D isotropic convolutions can deteriorate the segmentation performance. Hence, the search algorithm selects for most of its nodes 2D convolutional operations that extract meaningful in-plane features and includes a P3D convolution to exploit inter-slice information.

The best EMONAS-Net architecture is trained for 2000 epochs with a patch of size 144×144×10 and a batch size of 2. The same training setting and post-processing operation applied in the prostate and hippocampus dataset are used in this dataset and described in section 5.3.1.2. The qualitative results of the EMONAS-Net on the validation set are shown in Figure 27. The network produces highly accurate segmentations that correctly delineate the contours of the RVC, LVC and LVM.

5.3.3.3 Benchmark Results

The EMONAS-Net is evaluated by submitting the segmentation results of the 50 unlabeled test images to the ACDC challenge website. Geometrical and clinical performance measures are calculated for the segmentation of RVC, LVC, and LVM. The geometrical measures utilized are the Dice similarity coefficient (DSC) and Hausdorff distance (HD) on end diastolic (ED) and end systolic (ES) phases. Meanwhile, the clinical performance measures include the correlation (corr), bias, and standard deviation of the ED volumes, ejection fractions (EF), and myocardium mass. The results for the RVC, LVC and LVM segmentation of the top performing groups in the challenge as of August 2020 are presented in Table 39, 40 and 41, respectively. In this challenge, both architectures manually designed for the cardiac segmentation task as well as architectures designed through an AutoML method are presented. This is denoted as design type in the tables heading. The competing AutoML methods use an ensemble of 10 networks to produce the final segmentation. Therefore, in addition to the single-model results (EMONAS-Net1) we also provide the results using a 5 network ensemble (EMONAS-Net5), in which the best architecture is trained using a 5-fold cross validation setting and the final segmentation is obtained through a majority voting of these 5 networks.

The EMONAS-Net has a competitive performance in comparison with the state-of-the-art methods, being ranked within the top 10 positions of the leaderboard in all evaluation metrics. The best performance is achieved in the segmentation of the RVC. In this substructure, the EMONAS-Net5 is ranked second in the mean DSC ES, mean HD ES and EF correlation. Furthermore, it is also positioned third in the mean HD ED. A good performance is also obtained in the segmentation of the LVM, where it is ranked first in the Mass ED correlation, and third in the mean

HD ES and volume ES correlation. In relation to the LVC segmentation, the EMONAS-Net5 is in the fourth position in terms of the mean HD ES and volume ED bias.

Although the ensemble of EMONAS-Net networks has an improvement over the single-network, the EMONAS-Net1 still has a very high performance that is not only comparable to the other automatically designed models but surpasses some manually designed architectures. Specifically in the segmentation of the RVC, the EMONAS-Net1 is ranked third in the mean DSC ED and DSC ES, and fourth in the mean HD ED, HD ES and EF correlation. In the LVM segmentation, it is positioned second for the volume ES correlation and fourth for the mean HD ED, HD ES and mass ED correlation. Finally in the LVC segmentation it is ranked second for the EF bias and fifth in the volume ED bias. In comparison with the AutoML approach of Isensee et al. [116] and the NAS framework of Baldeon et al. [104], which both utilize a 10 network ensemble totaling 235.5×10^6 and 24×10^6 number of parameters, respectively, the EMONAS-Net1 has only 7.1×10^6 number of parameters. Hence providing a reduction in the number of parameters of $33 \times$ against [116] and $3.4 \times$ against [104]. Moreover, while the other competing AutoML approaches must perform the hyperparameter fitting for each network in their ensemble, our method only requires optimizing the hyperparameters once which significantly reduces the computational cost.

Table 39 Evaluation metrics for the segmentation of the Right Ventricle Cavity of the EMONAS-Net and top competing models on the ACDC challenge dataset.

Method	Design Type	DSC [%] ED	DSC [%] ES	HD [mm] ED	HD [mm] ES	EF corr	EF bias	Volume ED corr	Volume ED bias
EMONAS-Net1	AutoML	93.7	88.8	10.251	12.421	0.889	-1.250	0.983	2.470
EMONAS-Net5	AutoML	93.4	89.0	10.189	12.175	0.900	-1.150	0.981	4.600
Baldeon [104]	AutoML	93.6	88.4	10.183	12.234	0.899	-2.120	0.989	2.550
Isensee [116]	AutoML	94.6	90.4	8.835	11.376	0.925	-2.966	0.991	2.136
Zotti [117]	Manual	93.4	88.5	11.052	12.650	0.869	-0.872	0.986	2.372
Zotti [118]	Manual	94.1	88.2	10.318	14.053	0.872	-2.228	0.991	-3.722
Painchaud [138]	Manual	93.3	88.4	13.718	13.323	0.865	-0.874	0.986	2.078
Khened [45]	Manual	93.5	87.9	13.994	13.930	0.858	-2.246	0.982	-2.896
Baumgartner [119]	Manual	93.2	88.3	12.670	14.691	0.851	1.218	0.977	-2.290
Wolterink [120]	Manual	92.8	87.2	11.879	13.399	0.852	-4.610	0.980	3.596
Rohe [121]	Manual	91.6	84.5	14.049	15.926	0.781	-0.662	0.983	7.340
Patravali [122]	Manual	91.1	81.9	13.517	18.729	0.791	6.784	0.945	5.634
Grinias [123]	Manual	88.7	76.7	19.041	24.249	0.756	-0.192	0.916	11.910
Yang [124]	Manual	78.9	77.0	30.285	31.089	0.576	8.832	0.789	47.322

Table 40 Evaluation metrics for the segmentation of the Left Ventricle Cavity of the EMONAS-Net and top competing models on the ACDC challenge dataset.

Method	Design Type	DSC [%] ED	DSC [%] ES	HD [mm] ED	HD [mm] ES	EF corr	EF bias	Volume ED corr	Volume ED bias
EMONAS-Net1	AutoML	95.4	89.8	6.593	9.024	0.987	-0.470	0.996	2.79
EMONAS-Net5	AutoML	95.5	90.7	6.291	8.406	0.986	-1.010	0.996	2.45
Baldeon [104]	AutoML	95.8	90.3	5.592	8.644	0.981	0.490	0.997	3.07
Isensee [116]	AutoML	96.5	93.3	5.608	6.300	0.992	0.338	0.997	1.590
Zotti [117]	Manual	96.4	91.2	6.180	8.386	0.990	-0.476	0.997	3.746
Painchaud [138]	Manual	96.1	91.1	6.152	8.278	0.990	-0.480	0.997	5.215
Khened [45]	Manual	96.4	91.7	8.129	8.968	0.989	-0.548	0.997	0.576
Baumgartner [119]	Manual	96.3	91.1	6.526	9.170	0.988	0.568	0.995	1.436
Wolterink [120]	Manual	96.1	91.8	7.515	9.603	0.988	-0.494	0.993	3.046
Rohe [121]	Manual	95.7	90.0	7.483	10.747	0.989	-0.094	0.993	4.182
Zotti [118]	Manual	95.7	90.5	6.641	8.706	0.987	-1.186	0.997	9.640
Patravali [122]	Manual	95.5	88.5	8.212	10.929	0.971	1.734	0.997	9.864
Grinias [123]	Manual	94.8	84.8	8.898	12.934	0.970	-1.736	0.992	2.454
Yang [124]	Manual	86.4	77.5	47.873	53.050	0.926	1.496	0.894	12.232

Table 41 Evaluation metrics for the segmentation of the Left Ventricle Myocardium of the EMONAS-Net and top competing models on the ACDC challenge dataset.

Method	Design Type	DSC [%] ED	DSC [%] ES	HD [mm] ED	HD [mm] ES	Volume ES corr	Volume ES bias	Mass ED corr	Mass ED bias
EMONAS-Net1	AutoML	86.5	88.7	8.655	8.788	0.988	-6.310	0.987	-6.570
EMONAS-Net5	AutoML	87.0	89.0	9.137	8.693	0.987	-4.240	0.990	-4.570
Baldeon [104]	AutoML	87.3	89.5	8.197	8.318	0.988	-1.790	0.989	-2.100
Isensee [116]	AutoML	89.6	91.9	7.609	7.148	0.989	-3.402	0.986	-4.053
Zotti [117]	Manual	88.6	90.2	9.586	9.291	0.980	1.160	0.986	-1.872
Painchaud [138]	Manual	88.1	89.7	8.651	9.598	0.979	0.296	0.987	-2.906
Khened [45]	Manual	88.9	89.8	9.841	12.582	0.979	-2.572	0.990	-2.873
Patravali [122]	Manual	88.2	89.7	9.757	11.256	0.986	-4.464	0.989	-11.586
Baumgartner [119]	Manual	89.2	90.1	8.703	10.637	0.983	-9.602	0.982	-6.861
Zotti [118]	Manual	88.4	89.6	8.708	9.264	0.960	-7.804	0.984	-12.405
Wolterink [120]	Manual	87.5	89.4	11.121	10.687	0.971	0.906	0.963	-0.960
Rohe [121]	Manual	86.7	86.9	11.536	13.034	0.955	5.130	0.967	-3.373
Grinias [123]	Manual	79.9	78.4	12.300	14.567	0.890	-1.682	0.950	-19.625

To further evaluate the efficiency of the EMONAS-Net framework against other NAS methods, we compare the EMONAS-Net against a reinforcement learning (RL) based NAS framework for medical image segmentation [66] tested on the ACDC dataset and the MEA algorithm with the EMONAS-Net search space (EMONAS SS+MEA). The reported evaluation metrics for the segmentation performance are the mean DSC and HD for the RVC, LVC and LVM. The results of the EMONAS-Net and EMONAS-Net SS+MEA are the ones provided by the challenge website on the test set while using a single network. For the efficiency evaluation, we present the number of GPU days used for the architecture search and the GPU specifications. If various GPUs were used in parallel for the search, the number

of GPU days was calculated by multiplying the number of GPUs with the number of days taken for the search. The results are shown in Table 42.

Table 42 Evaluation metrics on the ACDC challenge dataset between the EMONAS-Net framework and competing NAS approaches for medical image segmentation.

NAS Method	DSC [%]			HD [mm]			Num. of Params.	GPU days	GPU Specification
	RVC	LVC	LVM	RVC	LVC	LVM			
EMONAS-Net	91.2	92.6	87.6	11.3	7.8	8.7	7.1×10^6	11.43	GTX 1070 Ti
EMONAS SS+MEA	91.0	92.7	87.5	11.6	7.7	9.4	7.0×10^6	23.79	GTX 1070 Ti
RL based NAS[16]	86.8	92.8	84.9	14.3	8.9	10.7	30×10^6	10	15 GTX Titan X

The EMONAS-Net leads the performance in most evaluation metrics in both segmentation accuracy and efficiency of the search. Against the RL based NAS, the EMONAS-Net finds an architecture with $4.2 \times$ fewer parameters that produces better segmentations on the three cardiac substructures with exception of the LVC DSC. In relation to the EMONAS SS+MEA, the proposed framework finds an architecture that has a better or similar segmentation performance while decreasing the search time in 52%. This demonstrates the search efficiency of our EMONAS-Net method.

5.4 Discussion

In this work, a surrogate-assisted multiobjective NAS framework that efficiently searches for medical image segmentation architectures is presented. The proposed framework has three main components, these are the EMONAS-Net search space, the selection probabilities and the Random Forest surrogate model. The experiments demonstrate that each component actively contributes to improve the efficiency of the search. The EMONAS-Net search space allows the best network to successfully exploit inter-slice and intra-slice information on highly anisotropic as well as isotropic medical datasets without the need of an ensemble of networks. Furthermore, it simultaneously optimizes both the micro- and macro-structure of the architecture with only one run of the search algorithm. Meanwhile, the competing AutoML frameworks either apply an ensemble of networks to leverage the inter-slice and intra-slice information and must perform the hyperparameter search for each architecture in their ensemble [114, 104], or need to solve a two-stage optimization problem to define the macro and micro configuration of the architecture [136].

On the other hand, through the combined application of the selection probabilities and Random Forest surrogate, our method is able to find architectures that achieve better or similar performance than previous work but in a fraction of the time. When compared against the MEA algorithm with the EMONAS-Net search space, the

EMONAS-Net framework reduces the search time by 42.9% on the prostate dataset, 51.8% on the hippocampus dataset, and 52.0% on the cardiac dataset. Against the AdaEn-Net, the EMONAS-Net framework reduces the search time by 60.7% and the number of architectures trained by 69.7% on the prostate dataset.

An important characteristic of the EMONAS-Net framework is its capability to optimize multiple objective functions. In this work, two objective functions have been minimized, the expected segmentation error and the number of parameters in the network. It is commonly believed that an accuracy gain can be achieved by increasing the depth [72] or width of the architecture [113]. However, our experimental results demonstrate that this is not always the case. The Pareto Fronts obtained for the prostate and hippocampus datasets showed that after a maximum number of parameters, the network's segmentation accuracy started to deteriorate. Meanwhile, on the cardiac dataset adding parameters over the 7.1×10^6 did not increase the segmentation accuracy. A similar conclusion is obtained when comparing the segmentation performance and size of the best EMONAS-Net architectures against the competing NAS frameworks. On the hippocampus dataset, the EMONAS-Net performs better than the C2FNAS while having $2.42 \times$ fewer parameters. On the cardiac dataset, the EMONAS-Net outperforms a reinforcement learning based NAS algorithm with an architecture that has $4.2 \times$ fewer parameters. In comparison with the AdaEn-Net, the EMONAS-Net has a similar or competitive performance on the cardiac and prostate dataset while reducing the number of parameters by $3.4 \times$ and $4.7 \times$, respectively. Moreover, our framework provides the flexibility to incorporate other objective functions needed for architecture search such as power consumption, memory usage, and number of floating-point operations.

Our experiments also provided insights on how the best hyperparameters are selected for each dataset. In relation to the micro-structure, none of the constructed cells have a linear flow of information. Instead, they form various parallel branches that transform the input tensor simultaneously and later aggregate the results together. Using shorter convolutional paths assembled in a parallel manner has shown to improve the information flow from earlier cells, reduce training difficulty and computations [139]. Furthermore, since each of the branches in the cell is able to extract features at a distinct scale the subsequent cells receive a more diverse set of input features to process. In our experiments, the configuration of the cell on the anisotropic datasets (prostate and cardiac) was very particular. Each cell was divided into two branches. One branch applied 3D type convolutions and the second branch only implemented 2D convolutions, which resembles the flow of information of a 2D-3D ensemble. Furthermore, on the highly anisotropic cardiac dataset, most of the nodes in the cell applied only 2D convolutions. A similar result on

anisotropic datasets has been obtained in previous works [116, 114, 104]. In the latter, an ensemble of 2D-3D FCN produced a better segmentation performance than a single 3D architecture and the 2D FCN outperformed a 3D FCN in a highly anisotropic dataset. Meanwhile in the isotropic hippocampus dataset, 2D and 3D convolutions were mixed in the same branch and 3D type convolutions were favored over the 2D convolutions. These results lead us to believe that our EMONAS-Net framework is able to distinguish the distinct image resolutions and select the most appropriate set of operations for the specific dataset.

Regarding the optimization of the hyperparameters of the macro-structure, the best values for the three datasets are the same. This shows that the CNN performance is more robust to the setting of the depth and number of filters per cell than the configuration of the cell. Although it is impossible to know if using the same hyperparameters on other medical datasets will guarantee an optimal performance, the values found could be good options to initialize the population. Previous work has shown that balancing an architecture's width and depth significantly affects the accuracy [125]. Hence, keeping the flexibility of the macro-structure in our hyperparameter search space improves the network's adaptability to unseen datasets. Moreover, the values for these hyperparameters did vary between the non-dominated solutions in the Pareto Front. Therefore, the inclusion of the macro-structure hyperparameters is fundamental to assess the trade-off between accuracy and size or selecting an intermediate solution that satisfies the computational restrictions.

Finally, a limitation of our model is the need to apply partial training during evolution. Even though we try to provide a better approximation of the final validation loss by using the expected segmentation error function, we evidenced in some cases a bias towards fast learning architectures rather than best performing architectures. In the future, a surrogate predictor that estimates directly the final validation loss based on the initial training epochs will be investigated.

Chapter 6: Conclusions

Deep neural networks have become very successful in a variety of applications. One crucial factor for their success is developing the proper neural architecture for the problem in hand. In this dissertation, deep neural networks that automatically adapt to new datasets and simultaneously search for efficient architectures have been presented. Furthermore, two multiobjective evolutionary based algorithms have been proposed to solve the complex hyperparameter optimization problem in deep neural network design.

In chapter 3, a multiobjective adaptive convolutional neural network for the 2D segmentation of medical magnetic resonance images is presented. The proposed network, named AdaResU-Net, has a fixed U-net shape with a residual framework as the building block, and a group of unset hyperparameters that are automatically fitted to the new dataset. The hyperparameters are selected through a learning framework that uses a multiobjective evolutionary based algorithm (MEA) to minimize both the Dice loss and the number of trainable parameters. This framework searches for AdaResU-Net configurations that can efficiently obtain a high segmentation accuracy. The AdaResU-Net model was evaluated on two publically available prostate and heart MRI datasets. The experiments show that the AdaResU-Net architecture with the proposed MEA learning framework is able to successfully adapt to the segmentation of the left ventricle endocardium and the prostate. Moreover, on both sets of images the AdaResU-Net MEA achieves a high segmentation accuracy with an adequate spatial definition that outperforms the well-known U-Net. Furthermore, the proposed MEA algorithm finds a set of hyperparameters that achieve equally good or better performance than a Gaussian process based Bayesian optimization approach, while providing considerably smaller architectures. In terms of the components of the AdaResU-Net, the incorporation of residual connections helps provide a better holistic shape and spatial distribution of the segmented region.

In chapter 4, a self-adaptive 2D-3D ensemble of fully convolutional networks that incorporates volumetric information is proposed for 3D medical image segmentation. The proposed ensemble, named AdaEn-Net, is composed of a 2D FCN that extracts 2D long-range feature relationships and a 3D FCN that analyzes intra-slice information. The previously presented MEA algorithm is implemented to automatically construct the 2D FCN and 3D FCN architectures for a specific segmentation task by minimizing the expected segmentation error (ESE) function and the

number of parameters in the network. The novel ESE objective function considers the training loss, validation loss, and expected training improvement of the networks. This makes the MEA algorithm search for architectures that have enough capacity to learn the relevant features from the training data and improve their performance in unseen images after being fully trained. The AdaEn-Net was evaluated for prostate segmentation using data from the PROMISE12 Grand Challenge and for cardiac segmentation from the MICCAI ACDC challenge. In both datasets, the proposed model achieved a performance within the top places of the leaderboard. It obtained comparable results with state-of-the-art CNNs manually tailored for the segmentation task in hand. Furthermore, it had a competitive performance to a top-ranked ruled-based adaptive framework in the prostate dataset while having $13\times$ fewer parameters, and achieved better results than a NAS reinforcement learning based algorithm on the cardiac dataset while having $1.25\times$ fewer parameters.

In chapter 5, an efficient surrogate-assisted multiobjective neural architecture search framework for medical image segmentation is presented. The proposed framework, named EMONAS-Net, has two main components: a novel search space that simultaneously defines the hyperparameters of the micro and macro-structure of the architecture, and the SaMEA algorithm that efficiently selects the best hyperparameter values for the problem in hand. The SaMEA algorithm is based upon the MEA algorithm and implements two strategies to improve the convergence and search speed. First, it uses the information obtained in the initial generations of evolution to lead the search towards more promising subproblems in the Pareto Front and improve the selection of hyperparameter values to mutate. Second, it applies a Random Forrest surrogate model to estimate a network's segmentation error without the need of training it. The EMONAS-Net framework is used to optimize the expected segmentation error and number of trainable parameters in the network in the tasks of prostate, hippocampus, and cardiac segmentation from three publically available challenges. The architectures found by the proposed framework perform better or similarly to architectures designed by other state-of-the-art AutoML and NAS methods while significantly reducing the size of the architectures and the search time. When compared against the MEA algorithm with the EMONAS-Net search space, the EMONAS-Net framework reduces the search time by 42.9% on the prostate dataset, 51.8% on the hippocampus dataset, and 52.0% on the cardiac dataset. Against the AdaEn-Net, the EMONAS-Net framework reduces the search time by 60.7% and the number of architectures trained by 69.7% on the prostate dataset.

To conclude, it is important to mention that although the models proposed in this research have been applied for medical analysis, the algorithms and self-adaptive networks can be applicable to other fields. For example, the

proposed approaches can be easily adapted to aid in 3D understanding of outdoor scenarios, object recognition for self-driving cars, robot navigation and drone-based understanding. Also, the proposed algorithms can be utilized to optimize the architecture and hyperparameters of other types of deep neural networks such as recurrent neural networks or deep belief networks, while considering additional objective functions that are particular to the specific application. Finally, an important contribution of this work is increasing the accessibility of deep learning to a broader range of organizations and people by reducing the need of expert knowledge and high-end computational resources when designing deep neural networks.

References

- [1] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in neural information processing systems*, 2012.
- [2] J. Long, E. Shelhamer and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.
- [3] W. Chan, N. Jaitly, Q. Le and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.
- [4] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao and K. Macherey, "Google's neural machine translation system: Bridging the gap between human and machine translation," in *arXiv preprint arXiv:1609.08144*, 2016.
- [5] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le and J. Dean, "Efficient neural architecture search via parameter sharing," in *arXiv preprint arXiv:1802.03268*, 2018.
- [6] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *International Conference on Learning Representations*, 2017.
- [7] K. Kandasamy, W. Neiswanger, J. Schneider, B. Póczos and E. P. Xing, "Neural architecture search with bayesian optimisation and optimal transport," in *Advances in Neural Information Processing Systems*, 2018.
- [8] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le and A. Kurakin, "Large-scale evolution of image classifiers," in *Proceedings of the 34th International Conference on Machine Learning*, 2017.
- [9] R. Luo, F. Tian, T. Qin, E. Chen and T.-Y. Liu, "Neural architecture optimization," in *Advances in neural information processing systems*, 2018.
- [10] H. Liu, K. Simonyan and Y. Yang, "Darts: Differentiable architecture search," in *arXiv preprint arXiv:1806.09055*, 2018.
- [11] E. Real, A. Aggarwal, Y. Huang and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proceedings of the aaii conference on artificial intelligence*, 2019.
- [12] H. Cai, T. Chen, W. Zhang, Y. Yu and J. Wang, "Efficient architecture search by network transformation," in *Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, 2018.
- [13] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang and K. Murphy, "Progressive neural architecture search," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

- [14] Y. Sun, H. Wang, B. Xue, Y. Jin, G. G. Yen and M. Zhang, "Surrogate-Assisted Evolutionary Deep Learning Using an End-to-End Random Forest-based Performance Predictor," *IEEE Transactions on Evolutionary Computation*, 2019.
- [15] A. Canziani, A. Paszke and E. Culurciello, "An analysis of deep neural network models for practical applications," *arXiv preprint arXiv:1605.07678*, 2016.
- [16] M. Denil, B. Shakibi, L. Dinh and N. De Freitas, "Predicting parameters in deep learning," in *Advances in neural information processing systems*, 2013.
- [17] S. Han, H. Mao and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *arXiv preprint arXiv:1510.00149*, 2015.
- [18] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li and S. Han, "Amc: Automl for model compression and acceleration on mobile devices," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [19] K. Ahmed and L. Torresani, "Connectivity learning in multi-branch networks," in *arXiv preprint arXiv:1709.09582*, 2017.
- [20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in *arXiv preprint arXiv:1704.04861*, 2017.
- [21] K. K. Roy and A. Phadikar, "Automated Medical Image Segmentation: A Survey," in *Proc. of Int. Conf. on Computing, Communication & Manufacturing*, 2014.
- [22] O. Ronneberg, P. Fischer and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, 2015.
- [23] Y. Xue, T. Xu, H. Zhang, R. Long and X. Huang, "SegAN: Adversarial Network with Multi-scale L1 Loss for Medical Image Segmentation," *arXiv preprint arXiv:1706.01805*, 2017.
- [24] O. Cicek, A. Abdulkadir, S. S. Lienkamp, T. Brox and O. Ronneberger, "3D U-Net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*, 2016.
- [25] K. Kamnitsas, C. Ledig, V. F. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert and B. Glocker, "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation," *Medical Image Analysis*, vol. 36, pp. 61-78, 2017.
- [26] H. Chen, Q. Dou, L. Yu, J. Qin and P.-A. Heng, "VoxResNet: Deep voxelwise residual networks for brain segmentation from 3D MR images," *NeuroImage*, vol. 170, pp. 446-455, 2018.
- [27] X. Li, H. Chen, X. Qi, Q. Dou, C.-W. Fu and P. A. Heng, "H-DenseUNet: Hybrid densely connected UNet for liver and liver tumor segmentation from CT volumes," *arXiv preprint arXiv:1709.07330*, 2017.
- [28] P. Mlynarski, H. Delingette, A. Criminisi and N. Ayache, "3D convolutional neural networks for tumor segmentation using long-range 2D context," *Computerized Medical Imaging and Graphics*, pp. 60-72, 2019.
- [29] N. Ramesh, J. Yoo and I. Sethi, "Thresholding based on histogram," in *IEEE Proc Vision Image Signal Proc*, 1995.

- [30] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine learning*, pp. 679-698, 1986.
- [31] N. Sharma and A. Ray, "Computer aided segmentation of medical images," in *Proc. of Int. Conf. on Mathematical Biolog*, 2006.
- [32] N. Sharma and L. M. Aggarwal, "Automated medical image segmentation techniques," *Journal of medical physics*, vol. 35, p. 3, 2010.
- [33] T. Gevers and A. Smeulders, "Combining region splitting and edge detection through guided Delaunay image subdivision," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997.
- [34] F. Argenti, L. Alparone and G. Benelli, "Fast algorithm for texture analysis using co-occurrence matrices," in *IEE Proc Part F: Radar Signal Proc*, 1990.
- [35] R. O. Duda, P. E. Hart, D. G. Stork and others, *Pattern classification*, Wiley New York, 1973.
- [36] B. B. Chaudhuri and N. Sarkar, "Texture segmentation using fractal dimension," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 17, pp. 72-77, 1995.
- [37] N. Dhanachandra, K. Manglem and Y. J. Chanu, "Image segmentation using K-means clustering algorithm and subtractive clustering algorithm," *Procedia Computer Science*, pp. 764-771, 2015.
- [38] H. M. Mofteh, A. T. Azar, E. T. Al-Shammari, N. I. Ghali, A. E. Hassanien and M. Shoman, "Adaptive k-means clustering algorithm for MR breast image segmentation," *Neural Computing and Applications*, pp. 1917-1928, 2014.
- [39] S. Chen, C. Ding and M. Liu, "Dual-force convolutional neural networks for accurate brain tumor segmentation," *Pattern Recognition*, vol. 88, pp. 90-100, 2019.
- [40] G. Chlebus, A. Schenk, J. H. Moltz, B. van Ginneken, H. K. Hahn and H. Meine, "Automatic liver tumor segmentation in CT with fully convolutional neural networks and object-based postprocessing," *Scientific reports*, vol. 8, pp. 1-7, 2018.
- [41] L. Yu, X. Yang, H. Chen, J. Qin and P.-A. Heng, "Volumetric ConvNets with mixed residual connections for automated prostate segmentation from 3D MR images," in *Thirty-first AAAI conference on artificial intelligence*, 2017.
- [42] H. Chen, X. Qi, L. Yu and P.-A. Heng, "DCAN: Deep Contour-Aware Networks for Accurate Gland Segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [43] V. Badrinarayanan, A. Kendall and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, pp. 2481-2495, 2017.
- [44] S. Jegou, M. Drozdal, D. Vazquez, A. Romero and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2017.
- [45] M. Khened, V. Alex and G. Krishnamurthi, "Densely connected fully convolutional network for short-axis cardiac cine MR image segmentation and heart diagnosis using random forest," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.

- [46] R. Alkadi, A. El-Baz, F. Taher and N. Werghi, "A 2.5 D Deep Learning-based Approach For Prostate Cancer Detection on T2-weighted Magnetic Resonance Imaging," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [47] A. Prasoon, K. Petersen, C. Igel, F. Lauze, E. Dam and M. Nielsen, "Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network," in *International conference on medical image computing and computer-assisted intervention*, 2013.
- [48] J. Chen, L. Yang, Y. Zhang, M. Alber and D. Z. Chen, "Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation," in *Advances in neural information processing systems*, 2016.
- [49] F. Milletari, N. Navab and S.-A. Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," in *Fourth International Conference on 3D Vision (3DV)*, 2016.
- [50] T. Elsken, J. H. Metzen and F. Hutter, "Neural Architecture Search: A Survey," *Journal of Machine Learning Research*, pp. 1-21, 2019.
- [51] B. Baker, O. Gupta, N. Naik and R. Raskar, "Designing neural network architectures using reinforcement learning," in *5th International Conference on Learning Representations*, Toulon, 2017.
- [52] B. Zoph, V. Vasudevan, J. Shlens and L. Quoc V, "Learning transferable architectures for scalable image recognition.," in *IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, 2018
- [53] Z. Zhao, Y. Junjie, W. Wei, S. Jing and L. Cheng-Lin, "Practical block-wise neural network architecture generation," in *IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, 2018.
- [54] R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan and N. Duffy, "Evolving deep neural networks," *arXiv preprint arXiv:1703.00548*, 2017.
- [55] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary computation*, pp. 99-127, 2002.
- [56] S. Xie, H. Zheng, C. Liu and L. Lin, "SNAS: stochastic neural architecture," in *Proceedings of the International Conference on Learning Representations*, New Orleans, 2019.
- [57] Z. Arber, A. K. F. Stefan and H. Frank, "Towards Automated Deep Learning: Efficient Joint Neural Architecture and Hyperparameter Search," in *International Conference in Machine Learning*, 2018.
- [58] K. Aaron, F. Stefan, B. Simon, H. Philipp and H. Frank, "Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets.," in *International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, 2017.
- [59] C. Patryk, L. Ilya and H. Frank, "Back to basics: Benchmarking canonical evolution strategies for playing atari.," in *International Joint Conferences on Artificial Intelligence Organization*, 2018.
- [60] E. Thomas, H. M. Jan and H. Frank, "Simple And Efficient Architecture Search for Convolutional Neural Networks," in *ICLR*, 2017.
- [61] A. Brock, T. Lim, J. M. Ritchie and N. Weston, "SMASH: one-shot model architecture search through hypernetworks," in *arXiv preprint arXiv:1708.05344*, 2017.

- [62] Y.-H. Kim, B. Reddy, S. Yun and C. Seo, "NEMO: Neuro-Evolution with Multiobjective Optimization of Deep Neural Network for Speed and Accuracy," in *JMLR: Workshop and Conference Proceedings*, 2017.
- [63] J.-D. Dong, A.-C. Cheng, D.-C. Juan, W. Wei and M. Sun, "Dpp-net: Device-aware progressive search for pareto-optimal neural architectures," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [64] T. Elsken, J. H. Metzen and F. Hutter, "Efficient multi-objective neural architecture search via lamarckian evolution," in *International Conference on Learning Representations*, New Orleans, 2019.
- [65] Z. Lu, I. Whalen, V. Boddeti, Y. Dhebar, K. Deb, E. Goodman and W. Banzhaf, "NSGA-Net: neural architecture search using multi-objective genetic algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2019.
- [66] A. Mortazi and U. Bagci, "Automatically designing CNN architectures for medical image segmentation," in *International Workshop on Machine Learning in Medical Imaging*, 2018.
- [67] Y. Weng, T. Zhou, Y. Li and X. Qiu, "NAS-Unet: Neural Architecture Search for Medical Image Segmentation," *IEEE Access*, vol. 7, pp. 44247-44257, 2019.
- [68] Z. Zhu, C. Liu, D. Yang, A. Yuille and D. Xu, "V-NAS: Neural Architecture Search for Volumetric Medical Image Segmentation," in *International Conference on 3D Vision (3DV)*, 2019.
- [69] S. Kim, I. Kim, S. Lim, W. Baek, C. Kim, H. Cho, B. Yoon and T. Kim, "Scalable Neural Architecture Search for 3D Medical Image Segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2019.
- [70] W. Bae, S. Lee, Y. Lee, B. Park, M. Chung and K.-H. Jung, "Resource Optimized Neural Architecture Search for 3D Medical Image Segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2019.
- [71] M. Baldeon and S. Lai-Yuen, "AdaResU-Net: Multiobjective adaptive convolutional neural network for medical image segmentation," *Neurocomputing*, DOI: <https://doi.org/10.1016/j.neucom.2019.01.110>, 2019.
- [72] K. He, Z. Xiangyu, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.
- [73] M. Baldeon-Calisto and S. Lai-Yuen, "ResU-Net: Residual Convolutional Neural Network for Prostate MRI segmentation.," in *IISE Annual Conference*, Orlando, 2018.
- [74] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.
- [75] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural networks: Tricks of the trade*, Berlin, Springer, 2012, pp. 437-478.
- [76] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [77] Q. Zhang and H. Li, "MOEA/D: A multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Transactions on Evolutionary Computations*, vol. 11, pp. 712-731, 2007.

- [78] J. Guo, S. Yang and S. Jiang, "An Adaptive Penalty-based Boundary Intersection Approach for Multiobjective Evolutionary Algorithm Based on Decomposition," in *IEEE Congress on Evolutionary Computation (CEC)*, 2016.
- [79] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, pp. 577-601, 2014.
- [80] K. Li, K. Deb, Q. Zhang and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 19, pp. 694-716, 2015.
- [81] G. Litjens, R. Toth, W. van de Ven, C. Hoeks, S. Kerkstra, B. van Ginneken, G. Vincent, G. Guillard, N. Birbeck and J. Zhang, "Evaluation of prostate segmentation algorithms for MRI: the PROMISE12 challenge," *Medical Image Analysis*, vol. 18, pp. 359-373, 2014.
- [82] A. Andreopoulos and J. K. Tsotsos, "Efficient and generalizable statistical models of shape and appearance for analysis of cardiac MRI," *Medical Image Analysis*, pp. 335-357, 2008.
- [83] S. Onal, S. K. Lai-Yuen, P. Bao, A. Weitzenfeld and S. Hart, "MRI-based segmentation of pubic bone for evaluation of pelvic organ prolapse," *IEEE Journal of Biomedical and Health Informatics*, pp. 1370-1378, 2014.
- [84] S. Onal, S. Lai-Yuen, P. Bao, A. Weitzenfeld, K. Greene, R. Kedar and S. Hart, "Assessment of a semiautomated pelvic floor measurement model for evaluating pelvic organ prolapse on MRI," *International Urogynecology Journal*, pp. 767-773, 2014.
- [85] S. Onal, X. Chen, S. K. Lai-Yuen and S. Hart, "Automatic vertebra segmentation on dynamic magnetic resonance imaging," *Journal of Medical Imaging*, 2017
- [86] W. Cong, J. Song, K. Luan, H. Liang, L. Wang, X. Ma and J. Li, "A Modified Brain MR Image Segmentation and Bias Field Estimation Model Based on Local and Global Information," *Computational and Mathematical Methods in Medicine*, 2016.
- [87] F. Chollet, "Keras," 2015. [Online]. Available: <https://github.com/keras-team/keras>.
- [88] J. Snoek, H. Larochelle and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems*, 2012.
- [89] Q. Zhu, B. Du, B. Turkbey, P. L. Choyke and P. Yan, "Deeply-Supervised CNN for prostate segmentation," in *International Joint Conference on Neural Networks (IJCNN)*, 2017.
- [90] P. F. Christ, F. Ettliger, F. Grun, M. E. A. Elshaera, J. Lipkova, S. Schlecht, F. Ahmaddy, S. Tatavarty, M. Bickel, P. Bilic, M. Rempfler, F. Hofmann, M. Anastasi and S.-A. Ahmadi, "Automatic liver and tumor segmentation of ct and mri volumes using cascaded fully convolutional neural networks," *arXiv preprint arXiv:1702.05970*, 2017.
- [91] R. Li, W. Liu, L. Yang, S. Sun, W. Hu, F. Zhang and W. Li, "DeepUNet: A Deep Fully Convolutional Network for Pixel-level Sea-Land Segmentation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 11, pp. 3954-3962, 2018.

- [92] J. Gonzalez and Z. Dai, *GPyOpt: A Bayesian Optimization framework in Python*, <http://github.com/SheffieldML/GPyOpt>, 2016.
- [93] T. Brosch, J. Peters, A. Groth, T. Stehle and J. Weese, "Deep learning-based boundary detection for model-based segmentation with application to MR prostate segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2018.
- [94] Z. Tang, M. Wang and Z. Song, "Rotationally resliced 3D prostate segmentation of MR images using Bhattacharyya similarity and active band theory," *Physica Medica*, vol. 54, pp. 56-65, 2018.
- [95] B. Sciolla, M. Martin and P. Delachartre, "Multi-pass 3D convolutional neural network segmentation of prostate MRI images}," 2017.
- [96] H. Jia, Y. Song, D. Zhang, H. Huang, D. Feng, M. Fulham, Y. Xia and W. Cai, "3D Global Convolutional Adversarial Network for Prostate MR Volume Segmentation," *arXiv preprint arXiv:1807.06742*, 2018.
- [97] L. Barba-J, B. Escalante-Ramirez, E. V. Venegas and F. A. Cosio, "A 3D Hermite-based multiscale local active contour method with elliptical shape constraints for segmentation of cardiac MR and CT volumes," *Medical & biological engineering & computing*, vol. 56, pp. 833-851, 2018.
- [98] J. Ehrhardt, T. Kepp, A. Schmidt-Richberg and H. Handels, "Joint multi-object registration and segmentation of left and right cardiac ventricles in 4D cine MRI," in *Medical Imaging 2014: Image Processing*, 2014.
- [99] C. Santiago, J. C. Nascimento and J. S. Marques, "A new ASM framework for left ventricle segmentation exploring slice variability in cardiac MRI volumes," *Neural Computing and Applications*, pp. 2489-2500, 2017.
- [100] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions," *arXiv preprint arXiv:1409.4842*, vol. 7, 2015.
- [101] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation*, pp. 61-70, 2011.
- [102] M. Siam, S. Elkerdawy, M. Jagersand and S. Yogamani, "Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges," in *IEEE 20th International Conference on Intelligent Transportation Systems*, 2017.
- [103] Z. Zhang, Q. Liu and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, 2018.
- [104] M. Baldeon Calisto and S. Lai-Yue, "AdaEn-Net: An Ensemble of Adaptive 2D-3D Fully Convolutional Networks for Medical Image Segmentation," *Neural Networks*, DOI: <https://doi.org/10.1016/j.neunet.2020.03.007>, 2020.
- [105] K. He, X. Zhang, S. Ren and J. Sun, "Identity mappings in deep residual networks.," in *European Conference on Computer Vision*, 2016.
- [106] O. Bernard, A. Lalande, C. Zotti, F. Cervenansky, X. Yang, P.-A. Heng, I. Cetin, K. Lekadir, O. Camara, M. A. G. Ballester and others, "Deep learning techniques for automatic MRI cardiac multi-structures segmentation and diagnosis: Is the problem solved?," *IEEE transactions on medical imaging*, vol. 11, pp. 2514-2525, 2018.

- [107] J. Tompson, R. Goroshin, A. Jain, Y. LeCun and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.
- [108] N. Srivastava, G. K. A. Hinton, I. Sutskever and R. Salakhutdinov, "Dropout: A simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, 2014.
- [109] R. G. Regis and C. A. Shoemaker, "Combining radial basis function surrogates and dynamic coordinate search in high-dimensional expensive black-box optimization," *Engineering Optimization*, pp. 529-555, 2013.
- [110] J. Muller, C. A. Shoemaker and R. Piche, "SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems," *Computers & Operations Research*, pp. 1383-1400, 2013.
- [111] K. He, X. Zhang, S. Ren and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015.
- [112] C. Zhang, P. Lim, A. Qin and K. C. Tan, "Multiobjective Deep Belief Network Ensemble for Remaining Useful Life Estimation in Prognostics," *IEEE Transactions of Neural Networks and Learning Systems*, vol. 28, pp. 2306-2318, 2017.
- [113] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *arXiv preprint arXiv:1605.07146*, 2016.
- [114] F. Isensee, J. Petersen, A. Klein, D. Zimmerer, P. F. Jaeger, S. Kohl, J. Wasserthal, G. Koehler, T. Norajitra, S. Wirkert and others, "nnU-Net: Breaking the Spell on Successful Medical Image Segmentation," *arXiv preprint arXiv:1904.08128*, 2019.
- [115] S. Liang and R. Srikant, "Why deep neural networks for function approximation?," in *arXiv preprint arXiv:1610.04161*, 2016.
- [116] F. Isensee, P. F. Jaeger, P. M. Full, I. Wolf, S. Engelhardt and K. H. Maier-Hein, "Automatic cardiac disease assessment on cine-MRI via time-series segmentation and domain specific features," in *International workshop on statistical atlases and computational models of the heart*, 2017.
- [117] C. Zotti, Z. Luo, A. Lalande and P.-M. Jodoin, "Convolutional neural network with shape prior applied to cardiac mri segmentation," *IEEE journal of biomedical and health informatics*, vol. 23, pp. 1119-1128, 2018.
- [118] C. Zotti, Z. Luo, O. Humbert, A. Lalande and P.-M. Jodoin, "GridNet with automatic shape prior registration for automatic MRI cardiac segmentation," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.
- [119] C. F. Baumgartner, L. M. Koch, M. Pollefeys and E. Konukoglu, "An exploration of 2D and 3D deep learning techniques for cardiac MR image segmentation," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.
- [120] J. M. Wolterink, T. Leiner, M. A. Viergever and I. Isgum, "Automatic segmentation and disease classification using cardiac cine MR images," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.
- [121] M.-M. Rohe, M. Sermesant and X. Pennec, "Automatic multi-atlas segmentation of myocardium with svf-net," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.


- [122] J. Patravali, S. Jain and S. Chilamkurthy, "2D-3D fully convolutional neural networks for cardiac MR segmentation," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.
- [123] E. Grinias and G. Tziritas, "Fast fully-automatic cardiac segmentation in MRI using MRF model optimization, substructures tracking and B-spline smoothing," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.
- [124] X. Yang, C. Bian, L. Yu, D. Ni and P.-A. Heng, "Class-balanced deep neural network for automatic ventricular structure segmentation," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, 2017.
- [125] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *arXiv preprint arXiv:1905.11946*, 2019.
- [126] A. L. Simpson, M. Antonelli, S. Bakas, M. Bilello, K. Farahani, B. Van Ginneken, A. Kopp-Schneider, B. A. Landman, G. Litjens, B. Menze and others, "A large annotated medical image dataset for the development and evaluation of segmentation algorithms," in *arXiv preprint arXiv:1902.09063*, 2019.
- [127] D. Ulyanov, A. Vedaldi and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," in *arXiv preprint arXiv:1607.08022*, 2016.
- [128] Z. Qiu, T. Yao and T. Mei, "Learning spatio-temporal representation with pseudo-3d residual networks," in *IEEE International Conference on Computer Vision*, 2017.
- [129] S. Liu, D. Xu, S. K. Zhou, O. Pauly, S. Grbic, T. Mertelmeier, J. Wicklein, A. Jerebko, W. Cai and D. Comaniciu, "3d anisotropic hybrid network: Transferring convolutional features from 2d images to 3d anisotropic volumes," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2018.
- [130] X. Cai, Y. Li, Z. Fan and Q. Zhang, "An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 508-523, 2014.
- [131] X. Cai and O. Wei, "A hybrid of decomposition and domination based evolutionary algorithm for multi-objective software next release problem," in *2013 10th IEEE International Conference on Control and Automation (ICCA)*, 2013.
- [132] Q. Zhang, W. Liu and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *2009 IEEE congress on evolutionary computation*, 2009.
- [133] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182--197, 2002.
- [134] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5-32, 2001.
- [135] M. Baldeon and S. K. Lai-Yuen, "Self-Adaptive 2D-3D Ensemble of Fully Convolutional Networks for Medical Image Segmentation," in *arXiv:1907.11587v1*, 2019.
- [136] Q. Yu, D. Yang, H. Roth, Y. Bai, Y. Zhang, A. L. Yuille and D. Xu, "C2FNAS: Coarse-to-Fine Neural Architecture Search for 3D Medical Image Segmentation," in *arXiv preprint arXiv:1912.09628*, 2019.

- [137] L. Wang, R. Chen, S. Wang, N. Zeng, X. Huang and C. Liu, "Nested Dilation Network (NDN) for Multi-Task Medical Image Segmentation," *IEEE Access*, vol. 7, pp. 44676-44685, 2019.
- [138] N. Painchaud, Y. Skandarani, T. Judge, O. Bernard, A. Lalande and P.-M. Jodoin, "Cardiac MRI segmentation with strong anatomical guarantees," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2019.
- [139] L. Zhao, M. Li, D. Meng, X. Li, Z. Zhang, Y. Zhuang, Z. Tu and J. Wang, "Deep convolutional neural networks with merge-and-run mappings," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018.

Appendix A: Copyright Permission

The copyright permissions for Chapter 3 and Chapter 4 is shown below.

← → ↻ elsevier.com/about/policies/copyright

 ELSEVIER About Elsevier Products & Solutions Services

Copyright

Describes the rights related to the publication and distribution of research. It governs how authors (as well as their employers or funders), publishers and the wider general public can use, publish and distribute articles or books.

[Journal author rights](#) [Government employees](#) [Elsevier's rights](#) [Protecting author rights](#) [Open access](#)

Journal author rights

In order for Elsevier to publish and disseminate research articles, we need publishing rights. This is determined by a publishing agreement between the author and Elsevier. This agreement deals with the transfer or license of the copyright to Elsevier and authors retain significant rights to use and share their own published articles. Elsevier supports the need for authors to share, disseminate and maximize the impact of their research and these rights, in Elsevier proprietary journals* are defined below:

For subscription articles	For open access articles
<p>Authors transfer copyright to the publisher as part of a journal publishing agreement, but have the right to:</p> <ul style="list-style-type: none">• Share their article for Personal Use, Internal Institutional Use and Scholarly Sharing purposes, with a DOI link to the version of record on ScienceDirect (and with the Creative Commons CC-BY-NC-ND license for author manuscript versions)• Retain patent, trademark and other intellectual property rights (including research data).• Proper attribution and credit for the published work.	<p>Authors sign an exclusive license agreement, where authors have copyright but license exclusive rights in their article to the publisher**. In this case authors have the right to:</p> <ul style="list-style-type: none">• Share their article in the same ways permitted to third parties under the relevant user license (together with Personal Use rights) so long as it contains a CrossMark logo, the end user license, and a DOI link to the version of record on ScienceDirect.• Retain patent, trademark and other intellectual property rights (including research data).• Proper attribution and credit for the published work.

Is Elsevier an STM signatory publisher? +

Do I need to request permission to re-use work from another STM publisher? +

Do I need to request permission to text mine Elsevier content? +

Can I include/use my article in my thesis/dissertation? –

Yes. Authors can include their articles in full or in part in a thesis or dissertation for non-commercial purposes.

Which uses of a work does Elsevier view as a form of 'prior publication'? +

How do I obtain permission to use Elsevier Journal material such as figures, tables, or text excerpts, if the request falls within the STM permissions guidelines? +

How do I obtain permission to use Elsevier Journal material such as figures, tables, or text excerpts, if the amount of material I wish to use does not fall within the free limits set out in the STM permissions guidelines? +

How do I obtain permission to use Elsevier Book material such as figures, tables, or text excerpts? +

How do I obtain permission to use Elsevier material that is NOT on ScienceDirect or Clinical Key? +

Can I use material from my Elsevier journal article within my thesis/dissertation? –

As an Elsevier journal author, you have the right to Include the article in a thesis or dissertation (provided that this is not to be published commercially) whether in full or in part, subject to proper acknowledgment; see the [Copyright page](#) for more information. No written permission from Elsevier is necessary.

This right extends to the posting of your thesis to your university's repository provided that if you include the published journal article, it is embedded in your thesis and not separately downloadable.