Automatic Design of Efficient Deep Neural Networks through Multiobjective Optimization for Medical Image Segmentation

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, modifying and fine-tuning deep learning architectures requires extensive knowledge, time, and computational resources.

There is a growing interest in developing methods that can automatically find the best architecture. These methods, known as neural architecture search (NAS) approaches, are usually modeled as a single-objective optimization problem where the aim is to find a network´s architecture that maximizes 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 designed architectures.

Although NAS has demonstrated great potential in automatically designing neural networks, it remains a computationally expensive and time-consuming process because it requires training and evaluating many potential configurations. Moreover, as the hyperparameter search space expands, the number of potential architectures to be trained increases significantly. Recent work has focused on improving the convergence time of NAS algorithms but most techniques have been applied to single-objective optimization algorithms. Given the much higher complexity of optimizing multiple objectives when designing a deep neural network, there is an increasing need to improve the convergence speed of multiobjective NAS.

A major application of deep learning is medical image segmentation, where deep learning techniques have emerged as powerful tools achieving state-of-the-art results in many medical image datasets. 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 imagining equipment and acquisition protocols. Therefore, neural networks are usually tailored to a specific application, anatomical region, and image modality. Moreover, in contrast with most image segmentation tasks that are performed on 2D images, medical image data is often volumetric requiring expensive 3D operations that result in large and complex architectures. These architectures usually have tens of millions of parameters that consume considerable storage and memory bandwidth, and make 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 a multiobjective evolutionary based algorithm for neural architecture search that increases the convergence speed and improves accuracy by expanding the architecture's configuration space.

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, a multiobjective evolutionary based algorithm that increases the convergence speed by efficiently searching through the search space is proposed for neural architecture search for 3D medical image segmentation. This will be accomplished by guiding the search towards more promising sub-problems and hyperparameter values, and applying an inexpensive surrogate function to estimate an architecture´s performance. Also, the hyperparameter search space will be expanded to explore different levels of an architecture and improve the learning capacity and flexibility of the optimal network configuration. The main research contributions are threefold. First, to propose strategies that reduce the computational time for neural architecture search of multiobjective problems. Second, to present a novel efficient multiobjective evolutionary based algorithm for the joint optimization of the predictive performance and size of an architecture for 3D medical image segmentation. Lastly, to propose a NAS method that simultaneously optimize the cell level structure and network level structure for high-resolution 3D image segmentation.

The broader impact of the proposed research is as follows: (1) automating the design of deep neural networks' architecture and hyperparameters to improve performance and reduce a model´s size; and (2) increase 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. 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 the problem of medical image segmentation, they can also be applied to other applications such as semantic segmentation and object recognition where deep neural networks that optimize multiple objective functions need to be designed.

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]. A 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). NAS can be considered a subfield of auto machine learning (AutoML) and has a significant overlap with hyperparameter optimization and meta-learning [5]. The optimization algorithms applied in NAS are usually based on reinforcement learning [6, 7], Bayesian optimization [8], evolutionary algorithms [9], and gradient-based methods [10, 11]. 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. NAS usually consists of three steps as shown in Fig 1: sample the hyperparameters of a candidate architecture from the search space, train and evaluate the performance of the candidate architecture in the specific task, and guide the search to a more promising architecture. These steps are

repeated iteratively until the performance measure converges to a desired threshold or a maximum number of iterations has been reached. The efficiency of a NAS method is affected by the design and size of the hyperparameter search space, the evaluation of the candidate architectures, and the selected search strategies. To reduce computational burden, recent works have proposed to decrease the hyperparameter search space by optimizing the configuration of a small cell that is repeated to form the final architecture [12]. Also, surrogate functions have been applied to estimate the performance of candidate neural networks without fully training them [10] or using a one-shot architecture search that defines an architecture as a subgraph of a supergraph and shares weight parameters between subgraphs to reduce training time [7], and strategies that direct the search towards the most promising areas of the hyperparameter search space [8]. However, these techniques are usually designed for NAS methods that only aim to optimize one objective function. In multiobjective optimization, a number of scalar optimization subproblems need to be solved simultaneously. Thus, there are three main challenges that need to be addressed: (1) how to include the optimization of all subproblems in the proposed search strategies; (2) how weight sharing can be performed between different-sized architectures and whether reusing weights will result in a reduction of training time; and (3) how to efficiently represent a candidate architecture as a subgraph of a supergraph without being limited by the GPU memory storage requirements.

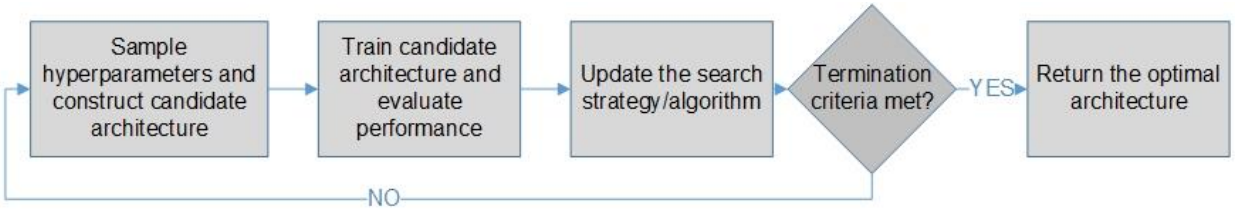

Fig. 1 Traditional neural architecture search process to automatically design a neural network for a specific problem.

Since the deep learning breakthrough in 2012, state-of-the-art networks have greatly increased in size and complexity to achieve better performance. This growth is visible in the size of the networks that have won the renowned ImageNet challenge in the last few years. In 2012, the winner network AlexNet [1] had a total of 62 million parameters while the 2018 winner GPipe [13] had a total of 557 million parameters. Hence, training and deploying current neural networks 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 [14, 15]. Recent works [16, 17, 18, 19] have shown that networks can be significantly reduced in size without a loss of accuracy. Therefore, there is a growing focus on improving both the performance and size of the networks to reduce unnecessary computational and memory usage and enable their use in many computationally-limited environments such as self-driving cars, and mobile applications.

NAS methods that simultaneously optimize the predictive performance and efficiency of the designed architectures through multiobjective algorithms are gaining attention. Kim et al. [20] employed a NSGA-II [21] based algorithm to design convolutional neural networks (CNNs) for image classification that maximize accuracy and minimize inference speed. In [22], Elsken et al. proposed a Lamarckian evolutionary algorithm for multiobjective NAS that optimizes the predictive performance and resource consumption of CNNs for image classification. Liang et al. [23] presented an evolutionary AutoML framework that optimizes the hyperparameter values, network architecture and size of the network of recurrent neural networks for text classification and CNNs for image classification. Very limited work has been presented to address the problem for image segmentation, which is the more difficult task of assigning a label to each pixel in an image to identify an object's class and its boundaries.

Within image segmentation, 3D medical image segmentation is a critical task to assist in clinical diagnosis 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 imagining 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 [24]. 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.

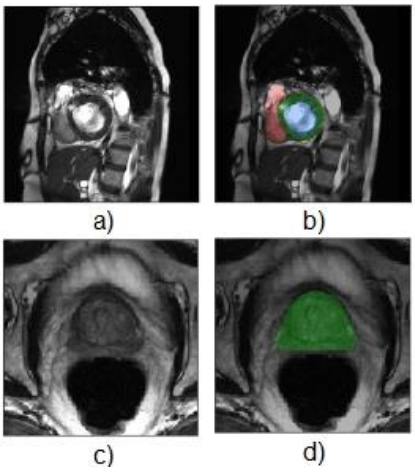

Fig. 2 Examples of segmentation of medical images: (a) Cardiac cine MRI, (b) Segmentation of the right ventricle cavity, left ventricle cavity and left ventricle myocardium, c) Prostate MRI, d) Segmentation of the Prostate.

In contrast with most image segmentation tasks that are performed on 2D images, medical image data is often volumetric requiring the methods to consider the entire volume to perform the segmentation in 3D. 2D CNNs have been developed, which segment each 2D slice independently and then concatenate the results along the third dimension to achieve the 3D segmentation [2, 25, 26]. Although these models are able to capture rich information in one plane, they do not fully exploit the spatial correlation along the z-axis affecting the segmentation accuracy. To overcome this limitation, 3D CNNs have been proposed by replacing 2D convolutions with 3D convolutions and directly processing volumetric information [27, 28, 29]. 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.

For these reasons, this research work 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 complexity; 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 a multiobjective evolutionary based algorithm for neural architecture search that increases the convergence speed and improves the learning capacity and flexibility of the constructed architectures.

2. RESEARCH OBJECTIVES

The main goal of this research is to design self-adaptive FCNs using multiobjective evolutionary based algorithms that can automatically find the optimal architecture for a specific dataset while minimizing the model´s size. This research work focuses on designing architectures for medical image segmentation, but the proposed algorithms are broad enough to be applied to the hyperparameter optimization of architectures for other applications. Furthermore, the self-adaptive network can be implemented in 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:

(1) ***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 [25] with a residual learning framework for a more efficient training. The proposed learning framework uses a multiobjective evolutionary 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 equally to the Bayesian hyperparameter optimization approach.

(2) ***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, 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 accuracy. 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.

(3) ***To design a multiobjective evolutionary based algorithm that increases the convergence speed and improves the learning capacity and flexibility of the constructed architectures.*** A novel multiobjective evolutionary based algorithm that efficiently searches through the hyperparameter search space and scalar optimization subproblems for neural architecture search of 3D medical image segmentation architectures is proposed. For the hyperparameter search, the hyperparameter values that have shown to produce a better average accuracy on the tested architectures will be assigned a higher selection probability during mutation. Similarly when solving the various scalar optimization subproblems, the subproblems that have actively contributed to approximate the Pareto Frontier will have a higher probability of being selected and solved in a generation. Hence, the increase in convergence speed will be achieved by using the performance evaluation during evolution to guide the future search to more promising subproblems and hyperparameter values. Furthermore, a radial basis function model will be applied as an inexpensive surrogate function to estimate a candidate´s architecture performance and decrease the training time. Finally, to improve the learning capacity and flexibility of the constructed architectures, the search space will be increased to include the search for operations in the encoder-decoder building blocks and the hyperparameters for the entire architecture. Therefore, jointly optimizing the hyperparameters related to the cell level and structure level of the optimal network.

3. LITERATURE REVIEW

This section 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 segmentations tasks. Then, neural architecture search is presented emphasizing the most recent methods for the construction of medical image segmentation architectures. Finally, the current challenges that will be addressed in this research are discussed.

3.1 Medical image segmentation techniques

Various methods have been proposed for 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 [30], edge based segmentation and grouping [31], and region based segmentation [32]. 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 [33]. 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 [34].

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 [35], Fourier filter method [36], and fractal texture description method [37]. Texture-based methods have shown to provide better results in medical image segmentation than intensity based methods [33]. 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 [38, 39].

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 [40], liver [41] and prostate [42].

3.2 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 are usually comprised of convolutional and de-convolutional layers as shown in Fig. 3. 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. De-convolutional layers also have trainable filters, but are applied with a fractional stride to increase the size of the feature map.

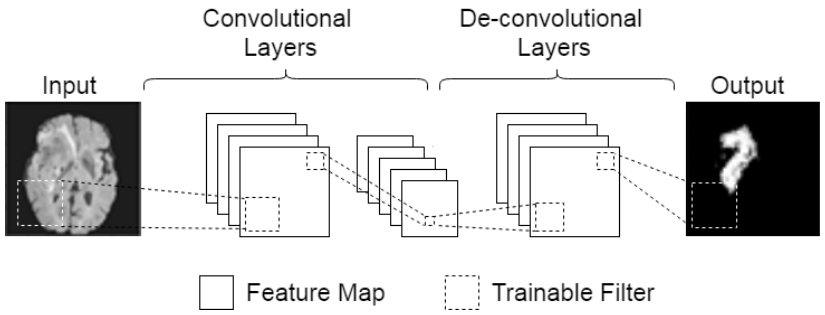

Fig. 3 A Fully Convolutional Network (FCN) comprised of 2 convolutional layers and 1 de-convolutional layer. Each layer has various trainable filters that extract features from the input map.

Similarly, in [43], Eigen *et al.* used a multiscale convolutional network to obtain a global prediction from the entire input image, and later refined the prediction from coarse to fine using other networks. On the other hand, Chen *et al.* [44] proposed the use of an atrous convolution with upsampled filters to increase the receptive size of the filter map. Then, a fully connected conditional random field was added at the final layer of the network to improve localization performance and obtain a detailed segmentation map.

CNNs have been increasingly used for medical image segmentation with the U-Net [25] being the most well-known architecture. Inspired in the FCN, the U-Net consists of a contracting path that extracts the most relevant features, followed by an expanding path that uses de-convolutions to increase the feature map to the size of the segmented image. Additionally, the authors include long skip-connections between opposing layers in the down-sampling and up-sampling 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 [45], SegNet [46], fully convolutional DenseNets [47, 48], and ResU-Net [49].

In an effort to incorporate volumetric information, researchers thereafter proposed 2.5D FCNs that input a stack of adjacent slices in the channel dimension and segment the center slice [50], and triplanar CNNs that integrate the prediction of three 2D networks trained on orthogonal planes [51]. Also, 3D information has been integrated by using recurrent neural networks (RNNs) [52], and 3D FCNs that replace 2D convolutions with 3D convolutions [53, 27]. 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.

In an effort 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 [54], 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 volumetric information. In [55] , 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 2D-3D architectures remain considerably big and comparable in size and memory consumption to other 3D CNNs.

3.3 Neural architecture search (NAS)

Neural architecture search (NAS) is the process of automatically designing neural network configurations for a specific problem. NAS can be regarded a subfield of auto machine learning (AutoML) and has a significant overlap with the area of hyperparameter optimization and meta-learning. NAS has three main components, namely the search space, optimization method, and evaluation method. The search space defines the set of feasible architectures that can be represented. Global search spaces have been proposed [56, 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 [12, 57]. In this search

space, the subspace is confined to represent smaller DAGs that are stacked in a predetermined template to form the larger architecture.

The optimization method in NAS determines how the search space is explored and the optimal solution is obtained. The most commonly used methods include reinforcement learning based algorithms [6, 56], evolutionary based algorithms [58, 59], Bayesian optimization [8], and gradient-based methods [60, 11]. 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 [9], but this demands high computational resources. Less expensive proxy-metrics have been proposed such as partial training [61], training on a subset of the data [62], or using images with lower resolution [63]. NAS-specific evaluation methods have also been proposed to reduce convergence time such a network morphism [64], weight-sharing [7], and hypernetworks [65].

3.4 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 find the hyperparameters of a 2D densely connected encoder-decoder baseline FCN. In [67], Weng et al. presented three types of primitive operation sets to construct down-sampling and up-sampling cells for a 2D U-Net backbone network. The cell configurations are updated using the DARTS [11] differential search strategy.

Very recently, NAS approaches that incorporate volumetric information during segmentation have been presented. Isensee et al. [68] presented a self-adapting framework that uses a rule-based approach to determine the pre-processing operations and training parameters of a pool of U-Net architectures (2D U-Net, 3D U-Net and cascaded U-Net). The model or ensemble that has the best performance on the medical dataset is selected for inference. Zhu et al. [69] proposed a differentiable NAS to choose between 2D, 3D and pseudo-3D convolutions on the modules of a predefined FCN template. Similarly, Kim et al. [70] presented a differentiable NAS framework to optimize the encoder, decoder, reduction and expansion cells of a 3D U-Net template architecture.

3.5 Current research challenges

The proposed research aim to address the following challenges: 1) the difficulty of designing accurate and efficient deep neural networks for a specific problem; 2) the computational cost and long-run times 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, hence designing an accurate 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, and size of kernels) that determine the accuracy of the model. Setting the correct values of these parameters is a non-trivial task because of the extensive hyperparameter 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 are 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 problem where performance as well as 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 accurate and efficient 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 [6, 9]. 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. There are three main factors that affect the efficiency of a NAS algorithm: the size of the hyperparameter search space, the number of candidate networks that need to be trained, and the search strategy. 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 or applied for single-objective optimization, it is necessary to design techniques for multiobjective NAS.

Finally, the development of highly complex and diverse architectures of FCNs has led to a dramatic improvement in the segmentation of medical images. However, it remains extremely difficult to adapt a FCN to a new segmentation task. Medical images have different characteristics compared to other types of images that need to 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 volumetric 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 3D medical images.

4. 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 networks can automatically adapt to different applications, increase the usability on computational limited environments, and provide a better understanding of the relation between hyperparameter selection and neural network accuracy. The presented AdaResU-Net and AdaEn-Net have shown to adapt to distinct medical datasets, succeeding in the segmentation of 2D and 3D complex structures like the prostate and heart (left ventricle cavity, right ventricle cavity, and left ventricle myocardium). Therefore, the proposed models are not restricted to only medical segmentation tasks, but can also be used in other non-medical fields that require the segmentation of irregular and complex shaped figures. For example, in aerial imagery segmentation, video understanding, object recognition for self-driving cars, content-based image retrieval and robot navigation.

The MEA algorithm, proposed to adapt the AdaResU-Net and AdaEn-Net to 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 MEA algorithm reframes the commonly used single objective hyperparameter optimization problem to a multiobjective problem, where a model´s generalization error as well as objectives related to model's size can be minimized simultaneously. Given the proliferation of deep learning on embedded applications and mobile devices, the proposed algorithm provides a solution to address the additional constraints in terms of size, memory and power consumption that are required for deep neural networks to be deployed.

Finally, a transcendental contribution on improving the efficiency of the MEA 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). NAS methods have helped democratizing AI by reducing the need for expert knowledge when designing deep neural networks. However, deploying the proposed algorithms still requires high computational resources, which reduces the accessibility to a large group of organizations and researchers. Consequently, developing an algorithm that automatically designs efficient deep neural networks can help improve the accessibility and applicability of deep learning technology to a wider group of people and problems.

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.

5. OBJECTIVE 1: 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 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 [25] with a residual framework [71] 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.

5.1 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 penalty-based boundary intersection (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.

5.1.1 AdaResU-Net architecture

The proposed fixed AdaResU-Net architecture is shown in Fig. 4, which is based on the ResU-Net architecture presented in our previous work [49]. 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 [71], 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.

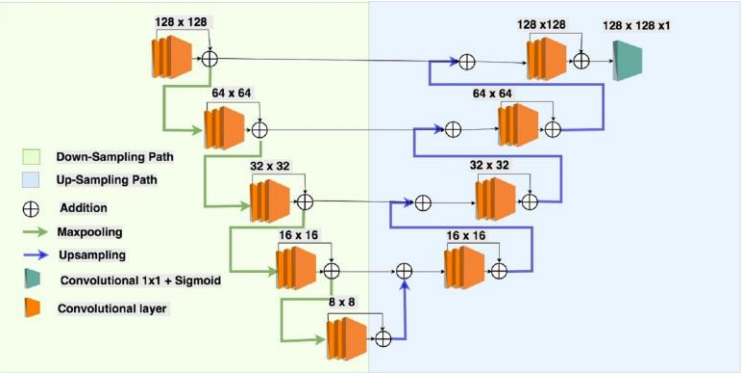

Fig. 4 Fixed AdaResU-Net architecture. Each box represents a dropout or convolutional layer. The arrows represent the max pooling, up sampling and residual operations. The size of the input in each convolutional block is indicated on top of each block.

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 [49], the proposed AdaResU-Net architecture includes dropout layers before the residual blocks to prevent the co-adaptation of feature detectors [72]. Then, the size of the features maps is reduced to decrease computational cost. Finally, the hyperparameters described in Section 4.2 have been left unset for the MEA learning framework to select automatically based on the specific dataset.

5.1.2 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 [73] 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 Sub-Section 4.1, 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 Fig. 5.

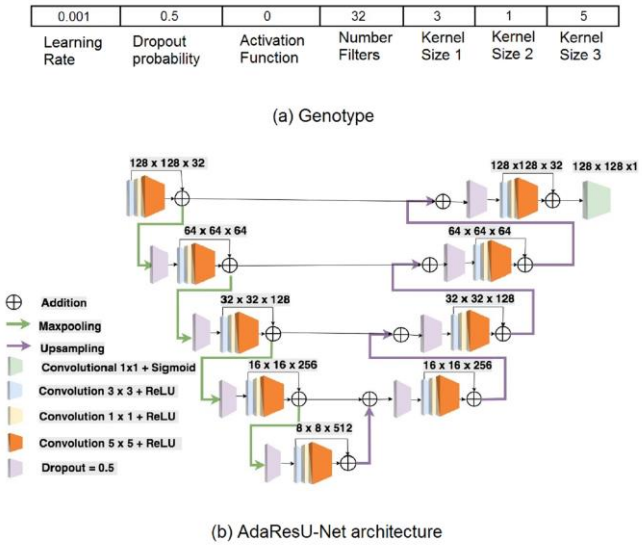

Fig. 5 Example of a genotype decoded into an AdaResU-Net architecture. (a) The genotype, where each gene refers to a hyperparameter. (b) The resultant AdaResU-Net network after using the genotype from (a). Different colors in the convolutional box represent a different kernel size.

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 [25] [53] [42] [74] [75].

5.1.3 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:

$$DSC = \frac{2\sum_1^K \widehat{y_i}(\theta)\, y_i}{\sum_1^K \widehat{y_i}(\theta) + \sum_1^K y_i} \quad (1)$$

Where y_i and $\widehat{y_i}(\theta)$ 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. $\widehat{y_i}(\theta)$ 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 *1-DSC*. We call *1-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:

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

$$subject\ to\ x\ \in \Omega \tag{2}$$

$|\cdot|$ 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.

Fig. 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 (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.

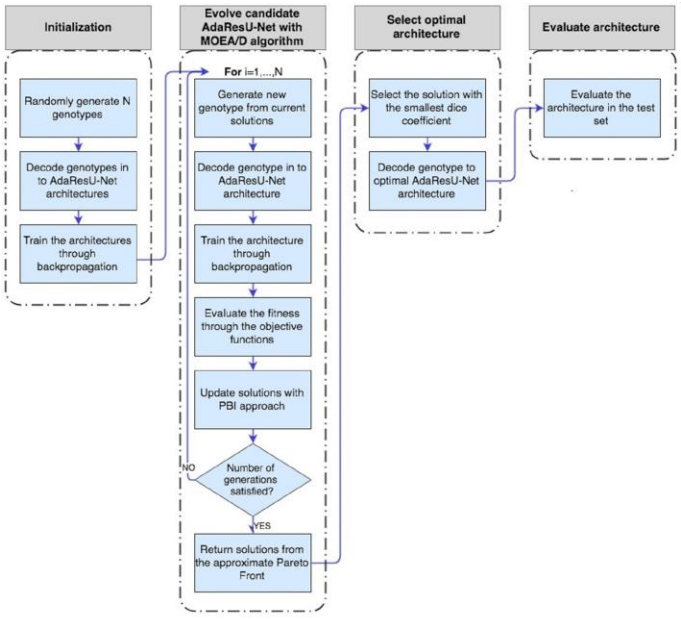

Fig. 6 Overview of the proposed MEA learning framework of the AdaResU-Net.

The proposed MEA algorithm is based on the framework of MOEA/D 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 [77] [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 [21] [79].

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:

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

where

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

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*= ∅, run time as *t*=0 and generation as *g*=1.

Step 1.2) For each weight vector $\lambda^i \; i \in \{1,2 \dots, N\}$ 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 ,…, 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 $FV^i = [\frac{f_1(x_g^i) - z_1^{min}}{z_1^{max} - z_1^{min}}, \dots, \frac{f_M(x_g^i) - z_M^{min}}{z_M^{max} - z_M^{min}}]$ for all $i \in \{1,2 \dots, N\}$.

For *i* =1… *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 $FV_g^y = [\frac{f_1(x_g^y) - z_1^{min}}{z_1^{max} - z_1^{min}}, \dots, \frac{f_m(x_g^y) - z_M^{min}}{z_M^{max} - z_M^{min}}]$.

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 $d_1^y = \frac{\left\|\left(FV_g^y - z^*\right)^T \lambda^j\right\|}{\|\lambda^j\|}$, $d_2^y = \left\|FV_g^y - (z^* - d_1 \frac{\lambda^j}{\|\lambda^j\|})\right\|$, $d_1^j = \frac{\left\|\left(FV_g^j - z^*\right)^T \lambda^j\right\|}{\|\lambda^j\|}$ and $d_2^j = \left\|FV_g^j - (z^* - d_1 \frac{\lambda^j}{\|\lambda^j\|})\right\|$.

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 $m \in \{1,2 \ldots, M\}$.

end for

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

end for
return *NDS*

5.2 Experimental Setup

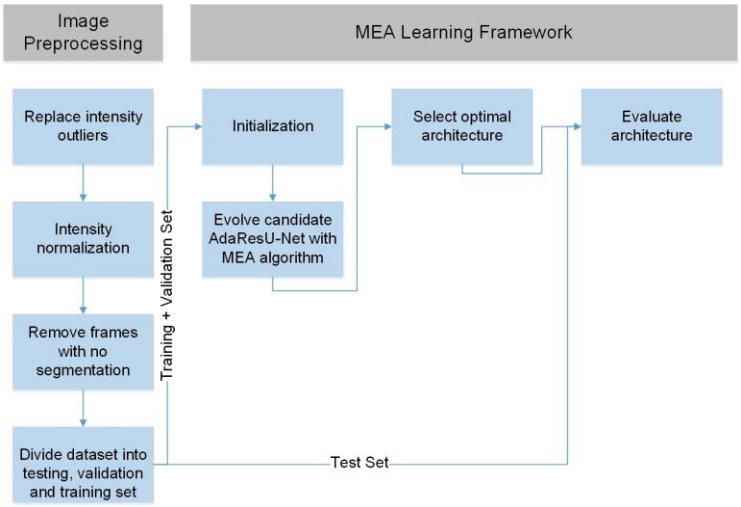

Fig. 7 Experimental process for the prostate and endocardium segmentation.

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 [80]. 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 [81] 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 256x256 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 Fig. 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 [82] [83], 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 [84], 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.

5.2.1 Image preprocessing

Some of the challenges when segmenting structures on MRI are the intensity inhomogeneity, noise interference, and distinct resolution [85]. 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.

5.2.2 Learning framework application

Evolution of AdaResU-Net candidates with the MEA algorithm

The MEA learning framework is implemented as described in Section 5.1.3. 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.

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.

Hyperparameter	Range
Learning rate	$[1x10^{-7}, 1x10^{-1}]$
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 1: Hyperparameter search space for generating new AdaResU-Net architecture configurations.

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

Table 2: Parameters for the proposed MEA algorithm.

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: $1x10^{8}$ 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 [86]. 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.

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.

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:

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

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:

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

The Hausdorff distance is defined as:

$$HD\left(Y, \hat{Y}(\theta)\right) = \max\left(h\left(Y, \hat{Y}(\theta)\right), h\left(\hat{Y}(\theta), Y\right)\right) \tag{8}$$

where

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

$\|\cdot\|$ 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 (9).

The mean surface distance is defined as:

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

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 [25], ResU-Net architecture [49], and resulting AdaResU-Net architecture after using a Bayesian optimization (BO) approach [87] 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 [75] [88] [89] [90]. 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 [87]. The BO algorithm is implemented

using the open-source Bayesian optimization toolbox GPyOpt [91]. 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.

5.3 Results

5.3.1 Prostate segmentation

For the prostate dataset segmentation, the AdaResU-Net candidates tested with the MEA algorithm and the obtained Pareto Front are presented in Fig. 8(a). A total of 8 AdaResU-Net architectures constitute the non-dominated frontier, where the selected configuration has 4.6 x 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.

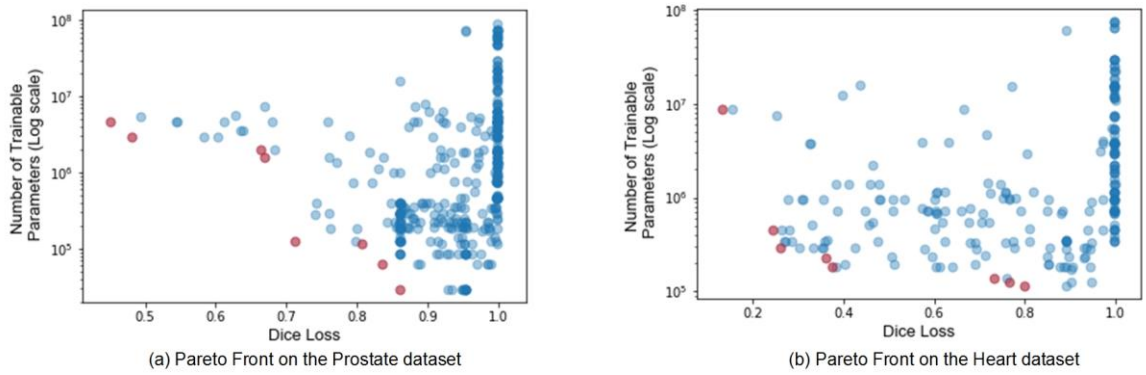

Fig. 8 Tested AdaResU-Net architectures and their corresponding value in the criterion space. The Pareto front is formed by the points in color red (a) Criterion space obtained for the prostate MRI dataset. (b) Criterion space obtained for the heart MRI dataset.

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 x 10^{-5}
AdaResU-Net BO	0.35	5	1	1	ReLU	32	5 x 10^{-5}

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).

In Table 4 and Fig. 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.

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 Fig. 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.

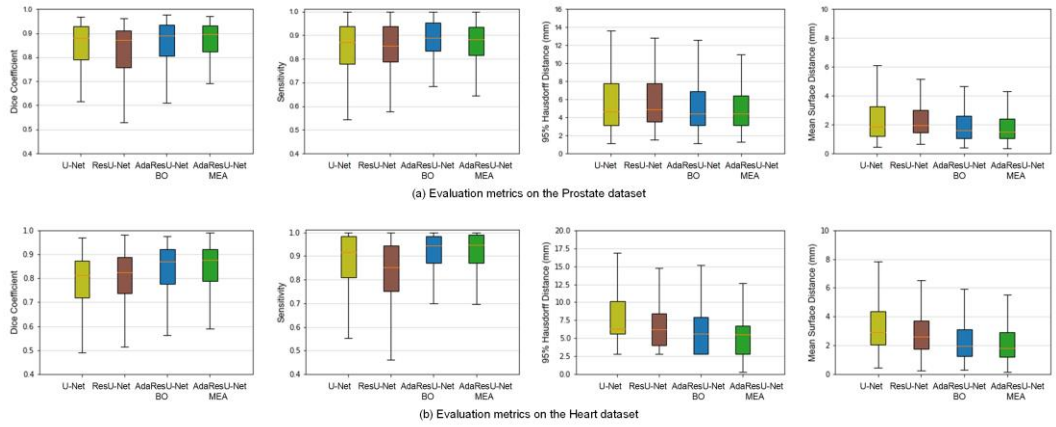

Fig. 9 Evaluation metrics of the AdaResU-Net with the MEA learning framework (AdaResU-Net MEA: rightmost green boxplot), AdaResU-Net with Bayesian optimization (AdaResU-Net BO: middle-right blue boxplot), ResU-Net (middle-left brown bloxplot) and U-Net (leftmost olive boxplot) on the test set. The metrics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

AdaResU-Net MEA achieves a higher mean dice coefficient, Hausdorff distance and mean surface distance than the other models. AdaResU-Net MEA and AdaResU-Net BO have an statistically equal mean sensitivity.

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 [25]	0.827 ± 0.146	0.840 ± 0.143	10.201 ± 15.435	3.244± 3.869	31.0×10^6

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

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

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

A visual illustration of the segmentation results for the prostate MRI are presented in Fig.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.

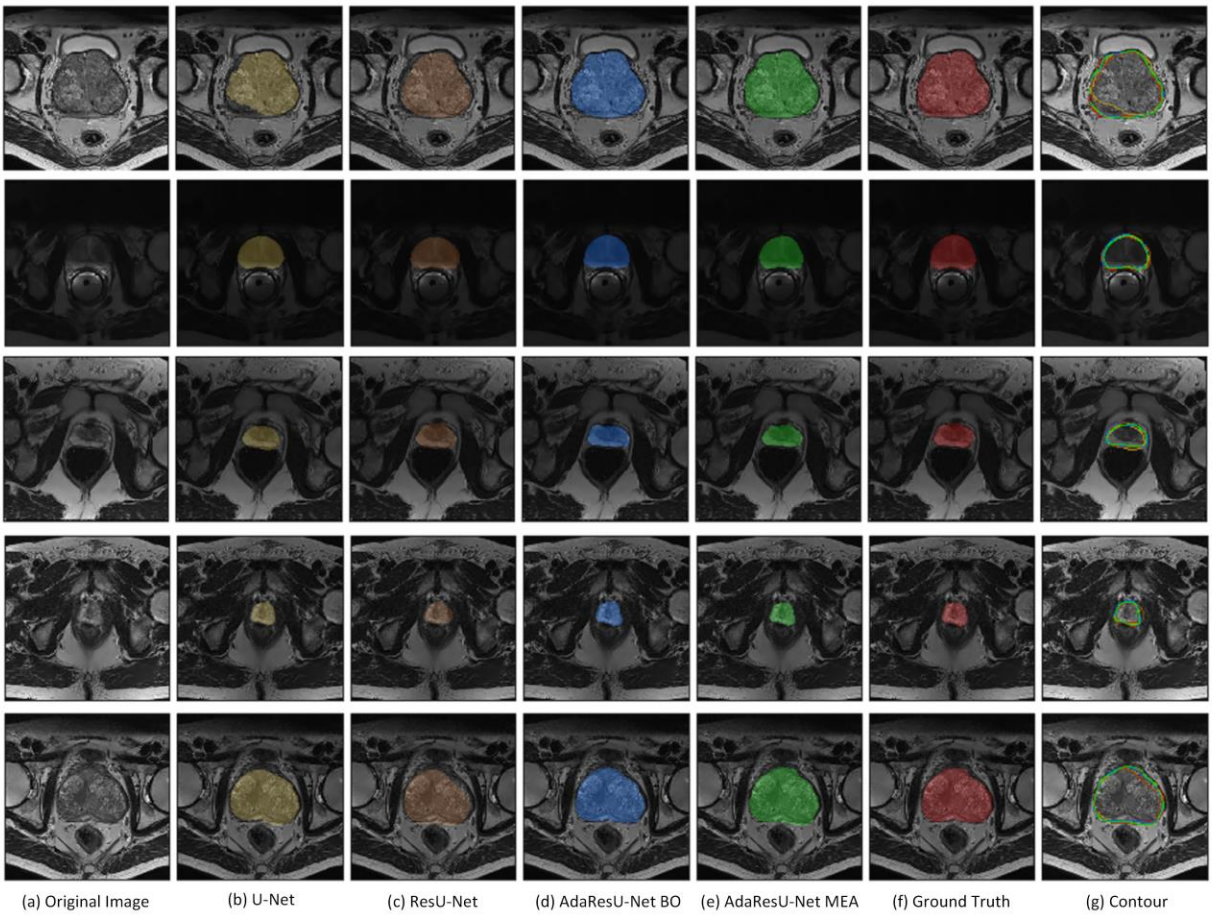

Fig. 10 Segmentation results for different cases of the Prostate MRI test set. (a) Input MR image (b) Image segmented with the U-Net architecture (c) Image segmented with the ResU-Net architecture (d) Image segmented with the AdaResU-Net architecture with Bayesian hyperparameter optimization (e) Image segmented with the AdaResU-Net architecture with our MEA learning 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 MEA (green). The AdaResU-Net MEA and AdaResU-Net BO are able to segment complex prostate shapes and provide smooth contours. The U-Net and ResU-Net suffer of under segmentation and some details of the prostate are not captured.

5.3.2 Endocardium segmentation

The Pareto Front and criterion space for the heart dataset are shown in Fig. 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 x 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.

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 x 10^{-5}
AdaResU-Net BO	0.45	1	5	3	ReLU	32	1 x 10^{-5}

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).

Table 7 and Fig. 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.

As with the prostate dataset, the set of hyperparameters selected with the proposed MEA learning framework and the Bayesian optimization approach are similar. The kernel sizes in most positions, number of filters and learning rate are alike. However, the BO method selects a kernel of size 5 that increases the size of the network and also increments the dropout probability. These differences make our AdaResU-Net MEA to provide better segmentations in the test set. Another interesting finding is that even though the proposed MEA learning framework aims to find the smaller networks, if the data is complex enough, it will search for hyperparameters that increase the capacity of the network. Thus, it will provide the smallest network without affecting the segmentation accuracy.

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 x 10^6**
AdaResU-Net BO	0.820 ± 0.163	**0.900 ± 0.142**	6.950 ± 8.928	2.839 ± 3.845	15.7 x 10^6
ResU-Net	0.791 ± 0.144	0.824 ± 0.159	7.727 ± 7.550	3.076 ± 2.176	10.9 x 10^6
U-Net [25]	0.770 ± 0.152	0.869 ± 0.151	8.616 ± 7.289	3.542 ± 2.292	31.0 x 10^6

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

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

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

In Fig. 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.

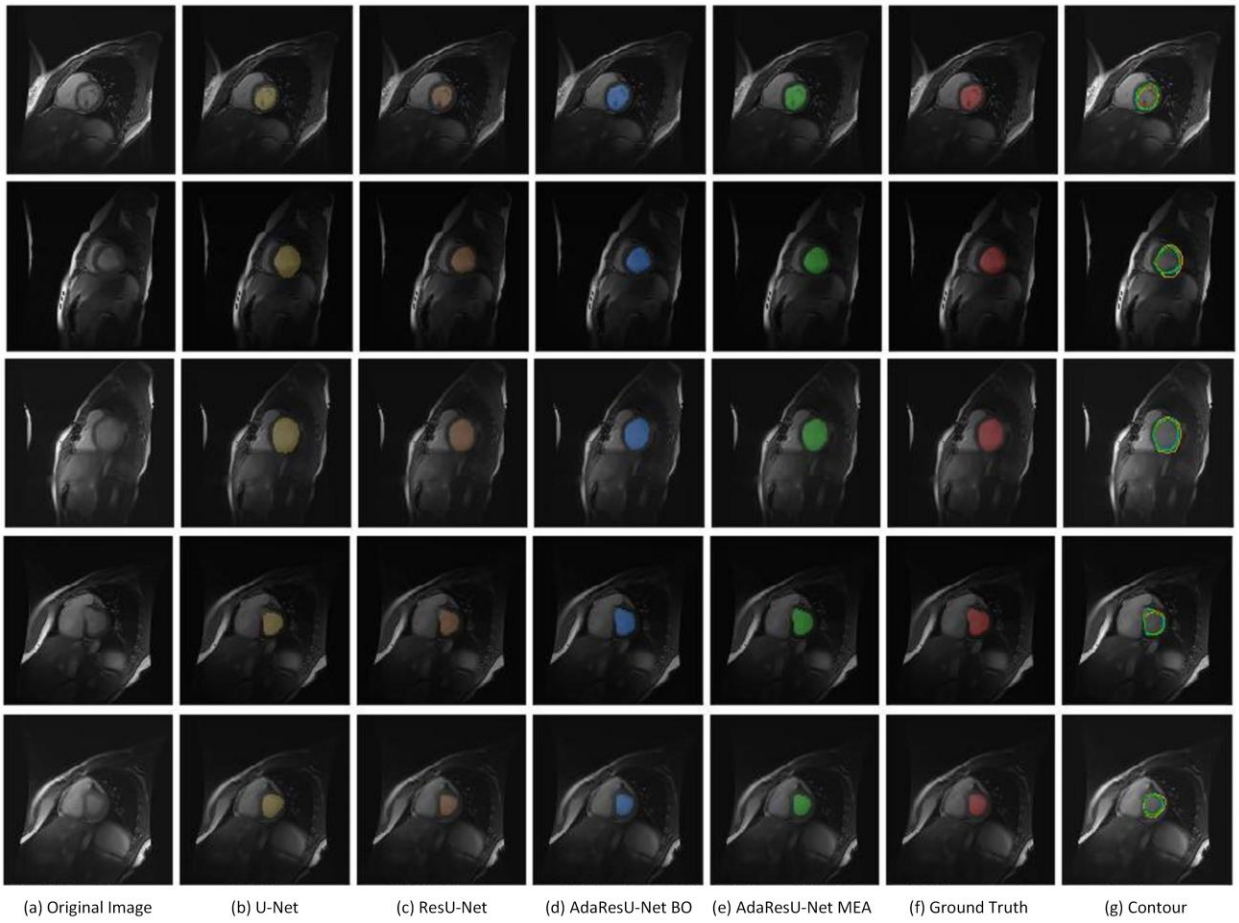

Fig. 11 Segmentation results for different cases of the Heart MRI test set. (a) Original MR image (b) Image segmented with the U-Net architecture (c) Image segmented with the ResU-Net architecture (d) Image segmented with the AdaResU-Net architecture with Bayesian hyperparameter optimization (e) Image segmented with the AdaResU-Net architecture with our MEA learning 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).The AdaResU-Net MEA identifies and segments correctly the endocardium. The shape, size and contour resemble the ground truth.

5.3.3 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. A images 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 previously, are shown in Fig. 12.

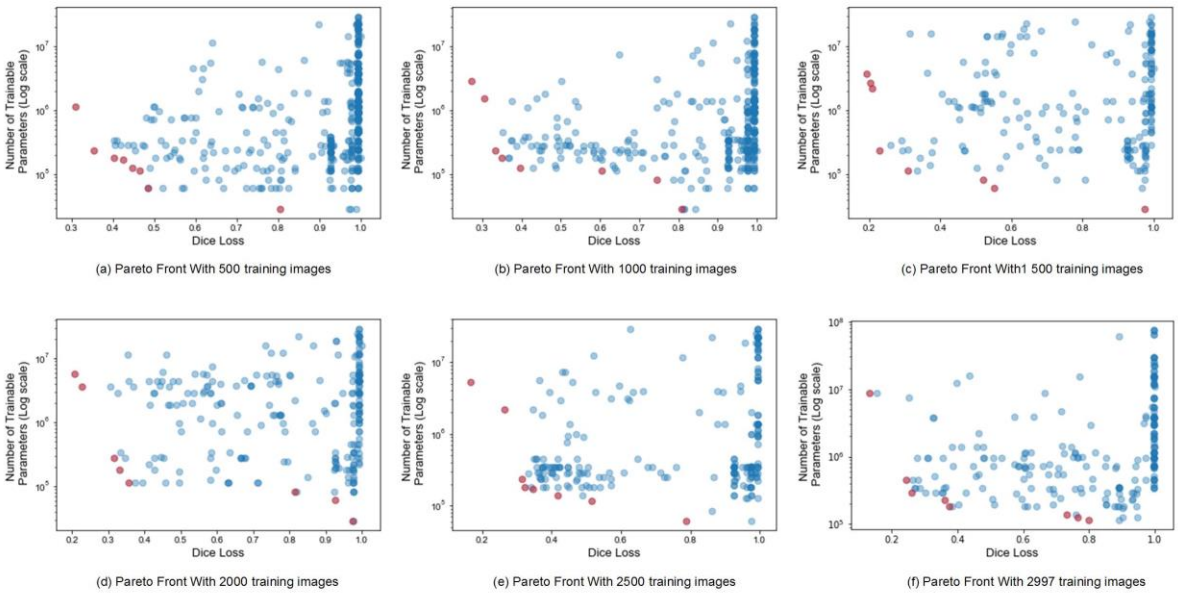

Fig. 12 Pareto front for the heart dataset with varying number of training images. As more training images become available the MEA learning framework finds bigger architectures that have a smaller Dice loss. However, the set of hyperparameters still construct architectures that are smaller than state-of-the-art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

Fig. 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.

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

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.

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

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.

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 Sub-Section 5.2.1, 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 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.

5.3.4 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.

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 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.

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	83,081.67	57,515.25
	Segmentation	1.09×10^{-2}	0.68×10^{-2}	0.97×10^{-2}	0.66×10^{-2}

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. The training time does not include the time it takes the MEA learning framework or Bayesian Optimization to adapt the AdaResU-Net configuration. The architectures were trained for 500 epochs on each dataset. The segmentation row presents the average time it takes the corresponding architecture to segment one 2D image.

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.

5.3.5 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 previous subsections 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 5.2.2.3 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.

Model	Type	Mean Dice Coefficient
AdaResU-Net*	Automatic: 2D CNN	0.877±0.031
Brosch *et* al. [92]	Automatic: Model-based	0.905
Lequan *et* al. [42]	Automatic: 3D CNN	0.894
Tang *et* al. [93]	Semi-Automatic: Graph Cuts	0.893 ± 0.031
Sciolla *et* al. [94]	Automatic: 3D/2D CNN	0.893
Jia *et* al. [95]	Automatic: 3D CNN	0.889

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	0.862±0.076
Barba-J *et* al. [96]	Semi-Automatic: Active Contour	0.912±0.026
Ehrhardt *et* al. [97]	Semi-Automatic: Atlas-based	0.86
Santiago *et* al. [98]	Automatic: Active Shape	0.794±0.081

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.

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 [93], 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 [96], 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.

5.4 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 in 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 Fig. 5 (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 5.3.3, 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 1x1 convolutional layers have shown to reduce the dimension of an architecture while increasing the depth and improving feature representation [99]. 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 [100] . 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.

It may be considered that 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 [101] [102], we believe the AdaResU-Net can also be used in other non-medical segmentation tasks while providing smaller architectures.

6. OBJECTIVE 2: 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 for medical image segmentation that incorporates volumetric information while optimizing both the performance and model's size is proposed. The presented **ada**ptive **en**semble, 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 [103]. 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 [80], and cardiac segmentation from the MICCAI ACDC challenge [104]. 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.

6.1 Methods

The AdaEn-Net is an adaptive 2D-3D ensemble network that is constructed in two phases as shown in Fig. 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 [105]. 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 quantified through the validation loss, training loss and epoch of the minimum loss. 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.

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 using a majority voting from the *k* predicted segmentations.

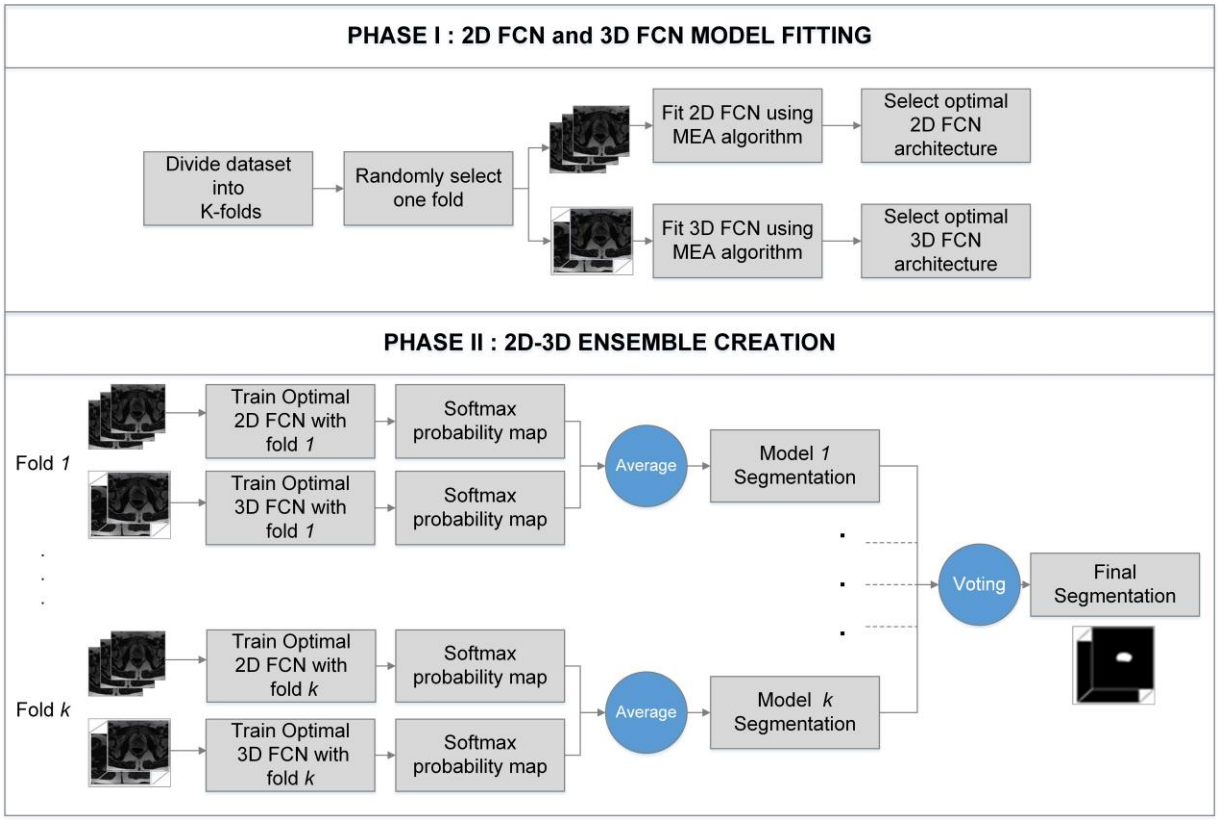

Fig. 13 Overview of the proposed framework for the 2D-3D FCN ensemble construction. In Phase I, the MEA algorithm is applied to find the optimal 2D FCN and 3D FCN for the specific segmentation task. In Phase II, the optimal 2D and 3D architectures are trained with each of the *k* folds from the dataset and their softmax probability maps are averaged. The final segmentation is achieved by using the majority voting from 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 image segmentation.

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

FCN Structure

A schematic view of the FCN architecture is shown in Fig. 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 Fig. 14(a), each module consists of three convolutional stages, each composed of a zero-padded convolutional layer, batch normalization layer [106], and an activation function layer. The first and last convolutional stages are linked through a residual connection [71], 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 [107, 25]. Furthermore, to prevent overfitting, a spatial dropout layer [108] is included before the residual blocks (except for the first residual block). Regular dropout assumes independence between feature map activations [109]. 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.

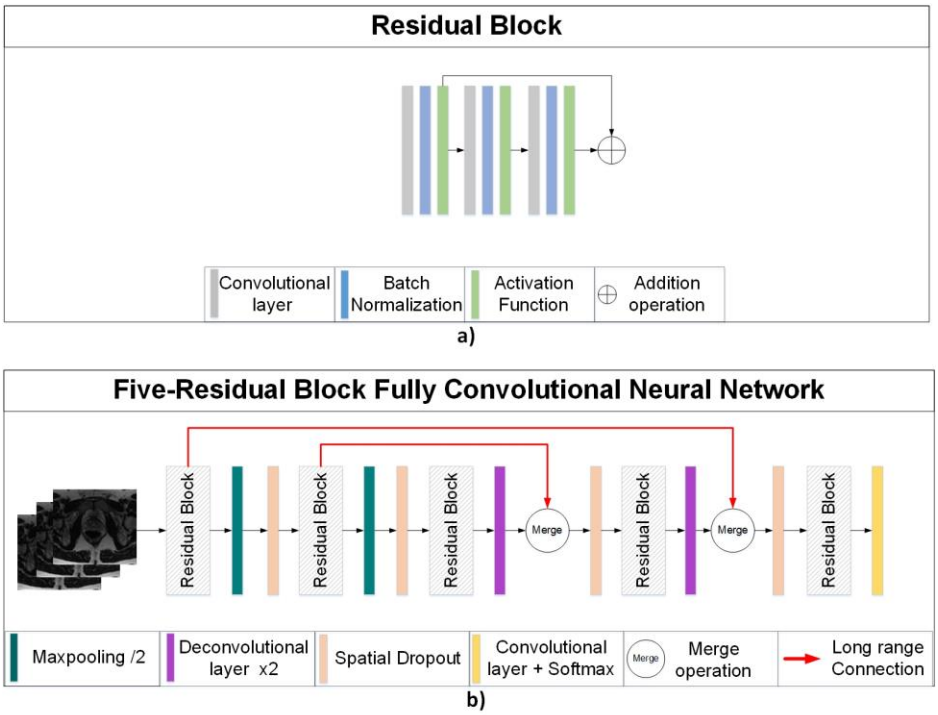

Fig. 14 Schematic view of the FCN architecture, which has an encoder-decoder structure. a) The residual block is the basic building module of the FCN architecture. The number of residual blocks are defined based on the specific segmentation task. b) An example of a five-Residual Block FCN.

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. The MEA algorithm is applied to find the hyperparameters values that optimally fit the FCN to the specific image dataset. The number of residual blocks, kernel sizes and number of filters completely define the depth and width of the neural network. Therefore, the search algorithm is able to find the appropriate architecture while simultaneously minimizing its size.

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.

Table 15: Set of hyperparameters and corresponding search space 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	$0.05 * n_p$	$n_p \in [0,1,..,14] \subset \mathbb{Z}$	$n_p \in [0,1,..,14] \subset \mathbb{Z}$
Learning Rate	-	$[1 \times 10^{-8}, 9 \times 10^{-3}]$	$[1 \times 10^{-8}, 9 \times 10^{-3}]$

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 the following function:

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

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 Fig. 14(b).

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 as follows:

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

Where n_f is set to an integer value to force the number of filters in NF_1 to be a power of 2.

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, the number of filters for residual block i (NF_i) is computed as follows:

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

Hence, 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 using Equation (13).

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. Displayed in Table 14 are the hyperparameters that define the 2D FCN and 3D FCN architectures along with their corresponding search space.

As observed in Table 14, 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.

MEA algorithm

The adaptation of the 2D FCN and 3D FCN to a medical image dataset is performed by applying the MEA algorithm [105] summarized in Algorithm 2. The adaptation process is modeled as a multi-objective 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. 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.

In this work, the nine hyperparameters shown in Table 16 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 16, 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 as:

$$Dice = \frac{2\sum_i \hat{y}_i y_i}{\sum_i \hat{y}_i + \sum_i y_i} \tag{14}$$

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:

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

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 in all segmented substructures. In this work, the function $C - MCDice$ is defined as the multi-class Dice loss.

The loss function the MEA algorithm uses to quantify the expected segmentation error 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. Thus, the loss function is defined as follows:

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

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 that represent local optimal solutions increases the robustness of the final prediction and provides a better approximation to the true global optimal solution [110]. 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. As shown in Fig. 15, candidate architectures FCN^1 and FCN^2 have similar average validation *MCDice* during the E training epochs. However, FCN^1 has already achieved the maximum performance, while FCN^2 could improve with additional training. 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.

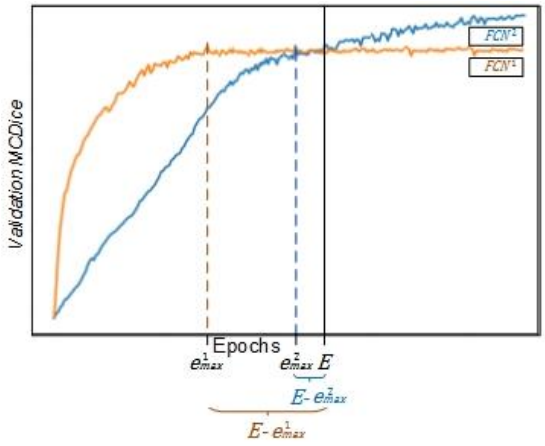

Fig. 15 Validation multi-class Dice coefficient (*MCDice*) for the candidate architectures FCN[1] and FCN[2] during the training epochs. Candidate architectures are partially trained for E epochs and e^i_{max} represents the epoch with the maximum validation *MCDice* for architecture *i.* Both candidate architectures have a similar average validation *MCDice* during the E training epochs. However, FCN[1] has already achieved the maximum performance, while FCN[2] could improve with additional training. The distance between e^i_{max} and E ($E - e^i_{max}$) can help distinguish this behavior.

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 reach 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 multi-objective optimization problem is defined as:

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

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

$$subject\ to\ x\ \in \Omega \quad (19)$$

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.

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 [76] 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 saved as the set of non-dominated points. In each generation afterwards, for each point of the non-dominated set, 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 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 points and Pareto Front are updated. After a specified number of generations, the algorithm terminates and the set of non-dominated solutions and Pareto Front are returned. Further details of the MEA algorithm are presented in [105]. 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.

Algorithm 2: Summary of the MEA algorithm for evolving architectures

Input:

- A Multi-Objective Hyperparameter Optimization Problem with *M* objective functions.
- Hyperparameter search space (*S*)
- Population size (*N*)
- Neighborhood size (*T*)

- Maximum number of generations (*G)*

Output:

- Optimal hyperparameters for the FCN architecture

1. Generate the initial population $\{x_1^1 \dots x_1^N\}$ of size *N* by randomly sampling the hyperparameters from *S*.
2. Train the *N* architectures. For each architecture calculate the *M* objective functions $F^j = [f_1(x_1^j), \dots, f_M(x_1^j)] \ \forall \ j \in \{1,2 \dots, N\}$. Include all solutions in the set of non-dominated solutions. Include initial solutions to the Pareto Front.
3. **While** $g < G$ **do**:
   - 3.1 **For** $i = 1 \dots N$ **do:**
     - 3.1.1 Select two solutions x_g^k and x_g^l from the neighborhood of i. Create new solution x_g^y by applying crossover and mutation operators to the two solutions.
     - 3.1.2 Train the FCN architecture with the hyperparameter values from solution x_g^y. Calculate the *M* objective functions $F^y = [f_1(x_g^y), \dots, f_M(x_g^y)]$
     - 3.1.3 Update the neighboring solutions of i using the Penalty based Intersection Approach (PBI).
     - 3.1.4 If no non-dominated solution dominates F^y, add F^y to the set of non-dominated solution and x_g^y to the Pareto Front. Eliminate any point and solution dominated by F^y.
4. **Return** the solution with smallest expected segmentation error in the set of non-dominated solutions

In the present work, the mutation operator is applied independently to each component of the hyperparameter vector with a probability that decreases monotonically. A high value 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, to ensure convergence we opt to reduce the mutation probability up to a minimum value. Also, solutions that are considerably far from the current solution are avoided to prevent turning the optimization algorithm into a random search. By combining the characteristics from the mutation functions in [111] and [112], the mutation probability $p(g)$ at the beginning of each generation g is computed as:

$$p(g) = max\left(\left[1 - \frac{\ln(g)}{\ln(G)}\right], \frac{1}{9}\right) \tag{20}$$

Where G is the maximum number of generations, and g is the current generation.

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 [105]. 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	$max\left(\left[1-\frac{\ln(g)}{\ln(40)}\right],\frac{1}{9}\right)$

6.1.2 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 until convergence with the ADAM optimizer [76]. Data augmentation is also applied during training to improve generalization. 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 Fig. 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:

$$\hat{y}^{i}{}_{ENS} = \underset{c}{\text{argmax}} \sum_k H_k^{i,c} \tag{21}$$

where

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

$\hat{y}^{i}{}_{ENS}$ is the predicted class label for pixel i using the ensemble network, $H_k^{i,c}$ is an indicator function, $\hat{y}^{i}{}_{m,k}$ 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 [113, 114, 115]. Ensemble diversity is attained when the classifiers make different errors on new data points. Meanwhile, an accurate classifier has a smaller error rate than random guessing [116]. 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 [116, 117, 118]. As discussed previously, 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 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 [118].

6.2 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 [80]. 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 [104]. The AdaEn-Net is implemented using Python 3.6 and Keras library [86]. 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.

6.2.1 Prostate MR Image Segmentation Challenge (PROMISE12)

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 1×1×1.5 mm and setting them to a fixed size of 128×128×64 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 96×96×16.

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 hyperparameters for both architectures as well as the number of parameters in each model are presented in Table 16.

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 [119]. 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 hyperparameters to construct the 2D FCN and 3D FCN for the two 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	$4x10^{-4}$	$5x10^{-5}$	$4x10^{-4}$	$9x10^{-5}$
Number of Parameters	$1.6\ x10^{6}$	$3.9\ x10^{6}$	$1.5\ x10^{6}$	$3.3\ x10^{6}$

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 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 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 displayed are the mean Dice similarity coefficient (DSC), sensitivity, 95 percentile Hausdorff distance (95 HD), and average boundary distance (ABD). Additionally, a one-tailed paired t-test with a 95% confidence level is applied to compare the performance of all pairs of architectures. The p-values for each paired t-test is presented in Table 19. The statistically significant results are indicated in bold on both tables.

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	ABD
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	$2.2x10^{-1}$	$4.9x10^{-1}$	$2.3x10^{-1}$	$1.4x10^{-1}$

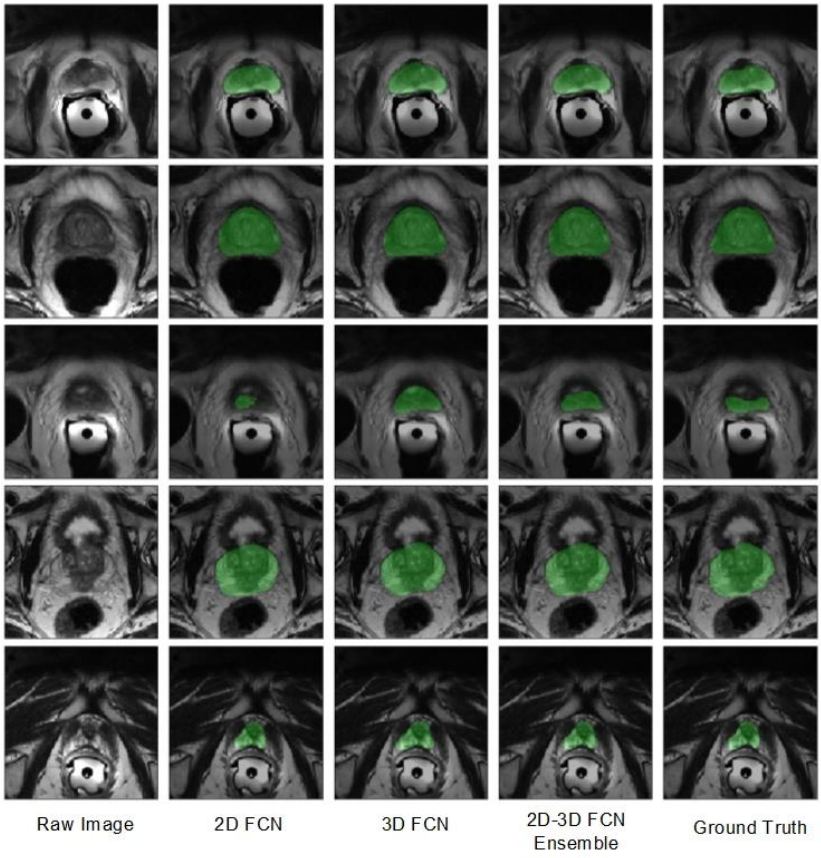

Fig. 16 Examples of prostate segmentation using the optimal 2D FCN, optimal 3D FCN, and 2D-3D FCN ensemble on the validation dataset. The prostate region is denoted in color green.

The experimental results show that the differences in 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 performs statistically better 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 Fig. 16. The examples show that the 3D FCN tends to over segment the prostate but provides a better definition in irregular areas. The 2D FCN appears to under segment certain regions while having smoother contours. Meanwhile, the 2D-3D FCN ensemble inherits the advantages of both models and produces segmentations with better spatial distribution and delineation. The performance improvement is particularly visible in the segmentation example found on the third row. 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.

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 [80] . 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 July 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 and Table 21. 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 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: Quantitative results of the proposed method (AdaEn-Net) and five competing models on the PROMISE12 challenge dataset. Only methods that have submitted a methodology description are displayed.

Model	**DSC[%]**			**95 HD [mm]**			**ABD [mm]**			**aRVD[%]**		
	Whole	**Apex**	**Base**	**Whole**	**Apex**	**Base**	**Whole**	**Apex**	**Base**	**Whole**	**Apex**	**Base**
AdaEn-Net	91.45	88.10	89.54	4.08	3.58	4.43	1.34	1.40	1.53	5.45	14.35	10.04
HD_Net	91.35	91.35	89.82	3.93	3.60	4.24	1.36	1.34	1.54	5.10	5.10	7.00
Bowda-Net	91.41	91.41	89.56	4.27	3.44	4.48	1.35	1.29	1.54	6.04	6.04	9.12
Sunrise2014	90.58	90.58	89.65	4.95	3.90	4.10	1.59	1.47	1.48	5.81	5.81	5.94
nnU-Net (I)	91.61	91.61	90.29	4.00	3.79	4.05	1.31	1.46	1.45	7.13	7.13	8.29
nnU-Net (II)	91.56	91.56	89.59	4.17	3.77	4.42	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 [68] 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 makes 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 considerably smaller (1.6 x10^6 parameters for the 2D FCN and 3.9 x10^6 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.

6.2.2 Automated Cardiac Diagnosis Challenge (ACDC)

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 mm^2, 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 1.56×1.56×10 mm per voxel and set to a fixed size of 192×192×10 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 144×144×10 and the 2D FCN with slices of size 192×192.

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 previously, 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 hyperparameters 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 [103, 120, 121].

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 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 for the predicted segmentation of the right ventricle cavity (RVC), left ventricle cavity (LVC) and left ventricle myocardium (LVM). Table 22 presents the results using 5-fold cross validation. Also, a one-tailed paired t-test with 95% confidence level is applied to statistically measure the difference in performance. The p-values for the t-test are presented in Table 23. Statistically significant results are indicated in bold on both tables.

Table 22: Evaluation metrics for the predicted segmentation of the right ventricle cavity (RVC), left ventricle cavity (LVC), and left ventricle myocardium (LVM) in the training set 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 23: 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	$\mathbf{3.6x10^{-4}}$	$3.4x10^{-1}$	$5.1x10^{-2}$	$\mathbf{1.2x10^{-2}}$	$1.8x10^{-1}$	$\mathbf{3.7x10^{-2}}$
2D-3D FCN ensemble	3D FCN	$\mathbf{9.0x10^{-19}}$	$\mathbf{3.2x10^{-9}}$	$\mathbf{3.3x10^{-21}}$	$\mathbf{6.9x10^{-12}}$	$\mathbf{2.2x10^{-4}}$	$\mathbf{1.1x10^{-2}}$
2D FCN	3D FCN	$\mathbf{1.5x10^{-15}}$	$\mathbf{3.7x10^{-8}}$	$\mathbf{1.5x10^{-15}}$	$\mathbf{5.7x10^{-8}}$	$\mathbf{1.4x10^{-4}}$	$\mathbf{7.4x10^{-3}}$

The 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 mm^2] 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 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 boarders [122]. 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.

Examples of segmentations produced by the 2D FCN, 3D FCN, and 2D-3D FCN ensemble are presented in Fig.17. The major visible drawbacks of the 3D FCN segmentations are discontinuities and irregular contours especially on the delineation of the RVC and LVM. The 2D FCN has an improvement over the 3D FCN; however, there is still presence of false negatives in the RVC segmentation. The 2D-3D ensemble FCN complements the true positives of both models and provides a smooth and continuous segmentation. This is particularly observed on the first two rows of Fig. 17.

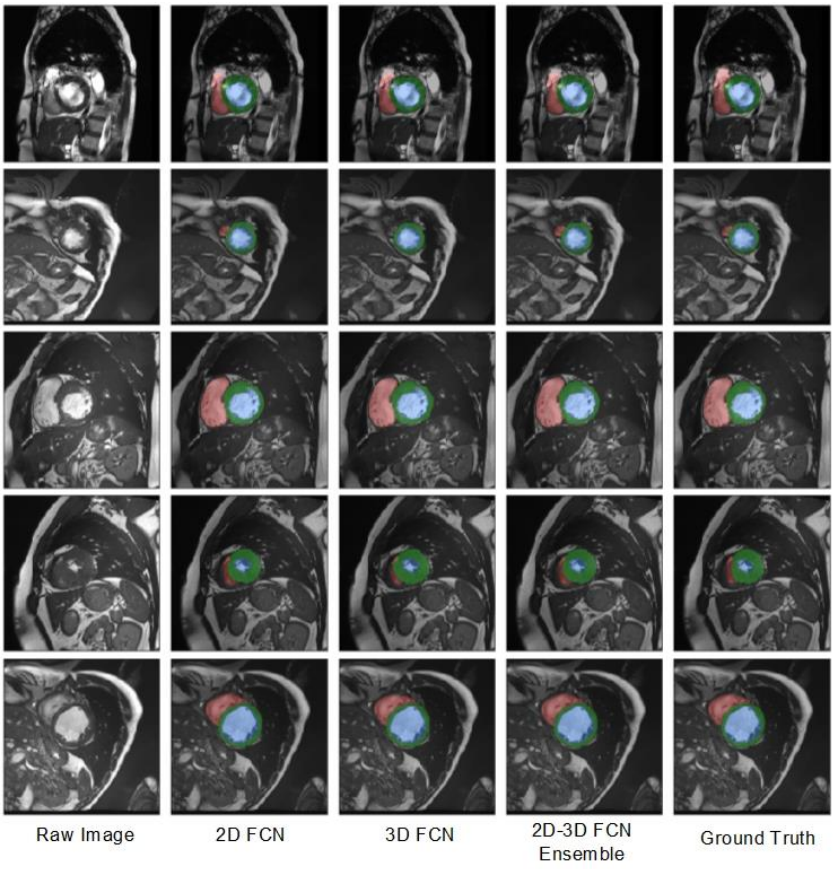

Fig. 17 Examples of cardiac segmentation using the optimal 2D FCN, optimal 3D FCN, and 2D-3D FCN ensemble on the validation dataset. The red regions denote the right ventricle cavity, blue regions the left ventricle cavity, and green regions the left ventricle myocardium.

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 24, 25 and 26, respectively. The top performing groups of the challenge, as of July 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 adjusted to the segmentation task.

Table 24: 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 (Ours)	Automatic	93.6	88.4	10.183	12.234	0.899	-2.120	0.989	2.550
Fabian Isensee [123]	Automatic	**94.6**	**90.4**	**8.835**	**11.376**	**0.925**	-2.966	**0.991**	**2.136**
Clement Zotti [124]	Manual	93.4	88.5	11.052	12.650	0.869	-0.872	0.986	2.372
Clement Zotti [125]	Manual	94.1	88.2	10.318	14.053	0.872	-2.228	0.991	-3.722
Mahendra Khened [48]	Manual	93.5	87.9	13.994	13.930	0.858	-2.246	0.982	-2.896
Christian Baumgartner [126]	Manual	93.2	88.3	12.670	14.691	0.851	1.218	0.977	-2.290
Jelmer Wolterink [127]	Manual	92.8	87.2	11.879	13.399	0.852	-4.610	0.980	3.596
Marc-Michel Rohe [128]	Manual	91.6	84.5	14.049	15.926	0.781	-0.662	0.983	7.340
Shubham Jain [129]	Manual	91.1	81.9	13.517	18.729	0.791	6.784	0.945	5.634
Ilias Grinias [130]	Manual	88.7	76.7	19.041	24.249	0.756	**-0.192**	0.916	11.910
Xin Yang [131]	Manual	78.9	77.0	30.285	31.089	0.576	8.832	0.789	47.322

Table 25: 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 (Ours)	Automatic	95.8	90.3	**5.592**	8.644	0.981	0.490	**0.997**	3.07
Fabian Isensee [123]	Automatic	**96.5**	**93.3**	5.608	**6.300**	**0.992**	0.338	**0.997**	1.590
Clement Zotti [124]	Manual	96.4	91.2	6.180	8.386	0.990	-0.476	**0.997**	3.746
Mahendra Khened [48]	Manual	96.4	91.7	8.129	8.968	0.989	-0.548	**0.997**	**0.576**
Christian Baumgartner [126]	Manual	96.3	91.1	6.526	9.170	0.988	0.568	0.995	1.436
Jelmer Wolterink [127]	Manual	96.1	91.8	7.515	9.603	0.988	-0.494	0.993	3.046
Marc-Michel Rohe [128]	Manual	95.7	90.0	7.483	10.747	0.989	**-0.094**	0.993	4.182
Clement Zotti [125]	Manual	95.7	90.5	6.641	8.706	0.987	-1.186	**0.997**	9.640
Shubham Jain [129]	Manual	95.5	88.5	8.212	10.929	0.971	1.734	**0.997**	9.864
Ilias Grinias [130]	Manual	94.8	84.8	8.898	12.934	0.970	-1.736	0.992	2.454
Xin Yang [131]	Manual	86.4	77.5	47.873	53.050	0.926	1.496	0.894	12.232

Table 26: 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 (Ours)	Automatic	87.3	89.5	8.197	8.318	0.988	-1.790	0.989	-2.100
Fabian Isensee [123].	Automatic	**89.6**	**91.9**	**7.609**	**7.148**	**0.989**	-3.402	0.986	-4.053
Clement Zotti [124]	Manual	88.6	90.2	9.586	9.291	0.980	1.160	0.986	-1.872
Mahendra Khened [48]	Manual	88.9	89.8	9.841	12.582	0.979	-2.572	**0.990**	-2.873
Shubham Jain [129]	Manual	88.2	89.7	9.757	11.256	0.986	-4.464	0.989	-11.586
Christian Baumgartner [126]	Manual	89.2	90.1	8.703	10.637	0.983	-9.602	0.982	-6.861
Clement Zotti [125]	Manual	88.4	89.6	8.708	9.264	0.960	-7.804	0.984	-12.405
Jelmer Wolterink [127]	Manual	87.5	89.4	11.121	10.687	0.971	**0.906**	0.963	**-0.960**
Marc-Michel Rohe [128]	Manual	86.7	86.9	11.536	13.034	0.955	5.130	0.967	-3.373
Ilias Grinias [130]	Manual	79.9	78.4	12.300	14.567	0.890	-1.682	0.950	-19.625

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. 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 previous experiments. 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 [124, 125], extraction of the region of interest using a Fourier analysis technique based on the heartbeat frequency [48], and rigid alignment by using the barycenter of the LV and RV [128]. 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. [123] 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.

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 27.

Table 27: 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 (Ours)	Evolutionary Algorithm	**91.0**	**93.0**	**88.4**	**11.2**	**7.1**	**8.3**
Mortazi and Bagci [66]	Policy gradient algorithm	86.8	92.8	84.9	14.3	8.9	10.7

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 27, 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 2D-3D FCN ensemble. In terms of computational time, the method proposed by Mortazi and Bagci [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.

6.3 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. It achieved comparable results with state-of-the art CNNs manually tailored for the segmentation task in hand. Furthermore, our model performed better than a top-ranked ruled-based adaptive framework in the prostate dataset while being considerably smaller and achieving better results than a NAS reinforcement learning based algorithm on the cardiac dataset. Differently from the competing manually and automatically designed networks, the implemented MEA algorithm also aims to design architectures with minimum 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 [132]. 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. Thus, 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 multi-objective 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 [133]. This indicates that NAS specialized in medical image segmentation is a promising area of research and the presented work contributes towards this development.

Finally, the proposed model has some limitations. Finding the optimal FCN requires significant time, especially when adapting the 3D FCN. Designing a neural network, either manually or automatically, is generally a time-consuming process. We are working on improving the convergence time by applying a surrogate loss function and sharing parameters between architectures. This is expected to improve the ability of our approach in identifying simple and efficient architectures faster.

7. OBJECTIVE 3: To design a multiobjective evolutionary based algorithm that increases the convergence speed and improves the learning capacity and flexibility of the constructed architectures

Automatically designing deep neural networks through the application of optimization algorithms is an emerging area of research due to the ability to find innovative configurations and produce highly accurate and efficient architectures. Designing a deep neural network is usually treated as a black box optimization process where an architecture is constructed, trained and its performance evaluated on the validation set. Given that training an architecture is costly and many feasible architectures need to be evaluated to effectively analyze the correlation between the tested hyperparameters and the performance function, automatically designing a deep neural network is time-consuming and requires multiple high-memory GPUs. To overcome this challenge, recent works have proposed embedding the hyperparameter search space from a discrete to a continuous space [10, 11], reducing the search space from the entire network to a building block that is repeated to construct the final network [12], or applying surrogate functions to estimate an architecture's validation loss [134]. These techniques are usually developed to optimize a single-objective function problem and do not consider additional objective functions during the architecture search. Considering that most deep learning applications need accurate but efficient architectures, and optimizing multiple objectives requires more iterations to approximate the Pareto Frontier, automatically designing deep neural networks for multiobjective problems is a more complex task where convergence speed is critical. In this research work, this challenge will be addressed by proposing an efficient multiobjective evolutionary based algorithm that aims to improve the search strategy and convergence speed. The proposed algorithm will use past performance to guide the search towards more promising sub-problems and hyperparameter values. Then, an inexpensive surrogate function will be trained to estimate a candidate´s architecture performance. Finally, to improve the learning capacity and flexibility of the constructed architectures, the search space will be expanded to include the search for operations in the encoder-decoder building blocks and the hyperparameters for the entire architecture (number of filters, number of blocks). This is expected to result in a NAS algorithm that can efficiently search for the whole architecture of a deep neural network while minimizing other objective functions such as the model's size.

7.1 Methods

As previously discussed, neural architecture search involves three aspects: the search space, search strategy, and performance estimation strategy. In the following sections, the representation of the CNN and search space is

presented. Afterwards, the search strategies developed for multiobjective NAS problems are introduced. Finally, the performance estimation strategy for evaluating candidate architectures is described.

7.1.1 Search Space

The search space defines the possible child architectures that can be generated given the group of unset hyperparameters and search range for each hyperparameter. As shown in Fig. 18, there are two main search spaces: building block and network. In the proposed work, the search space will include the operations to construct the building blocks that are part of the encoder-decoder path, and hyperparameters that define the width and depth of the network architecture.

Building block search space

For designing the encoder-decoder building blocks, a cell-based search space is employed following previous successful NAS algorithms [7, 12, 134, 135] The major advantages of this cell-based search space are that the size is drastically reduced to decrease convergence time, the resulting architectures built by stacking the cells are more easily transferred to other datasets, and repeating building blocks has proven to be a useful design principle in many hand-made architecture [5]. As shown in Fig. 18, the building block will be represented as a directed acyclical graph (DAG), in which a node represents an operation and an edge the flow of feature maps between operations. In the proposed work, three decisions variables characterize a node: the type of convolutional operation applied in the node, the kernel size for the convolutional operation, and the input to the node.

The set of possible convolutional operations are 2D convolutions, 3D convolutions, and Pseudo-3D (P3D) convolutions. Selecting between these types of convolutional operations allows the NAS algorithm 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. Searching between 2D, 3D and P3D convolutional operations has already been presented by Zhu et al. [69]. However, differently from their fixed cell structure, the proposed work adds flexibility to the construction of the block by searching for the optimal connection between nodes, which determines how the information will flow within the block, and the kernel size of the convolutional operation. The selection of the latter, also allows the algorithm to determine the appropriate width of the architecture. The set of kernel sizes for the convolutional operations will be 3, 5 and 7. On the other hand, the set of possible inputs to a node is the set of all former nodes' outputs plus the block's input feature. For example in Fig. 18, the set of possible inputs to node 3 are the block's input feature, the output of node 1 and the output of node 2 (shown as a dotted line). Hence, the NAS algorithm will determine the optimal flow of feature maps in a block. Finally, the output of the block is the concatenation of the output feature maps of all nodes. The number of nodes in a building block is a hyperparameter that requires high computational resources to test. Therefore, following the most recent work of NAS in medical image segmentation architectures [67, 70], the building block will be composed of 4 nodes. This number of nodes has shown to provide good performing models while reducing the computational time. The hyperparameters for the encoder-decoder building block will be encoded in a vector of size 4*3=12, where the value 3 refers to the three decision variables that need to defined for each node (type of convolutional operation, kernel size and input node).

Network search space

One important disadvantage when only searching for the cell-level structure of an architecture is that the macro structure is either manually engineered for the task [67, 69, 70] or a second hyperparameter optimization problem needs to be solved [12, 134]. To overcome these problems, the search is not only confined to the building block structure but also includes the hyperparameters that define the depth (number of decoder and encoder building blocks) and width (number of filters) of the architecture as shown in Fig. 18. Furthermore, adding these hyperparameters will allow to jointly optimize for the performance and efficiency of an architecture to prevent

over-parametrization. Although it is expected that the addition of these hyperparameters will increase the search time, it is also anticipated that these additional hyperparameters will reduce the time of hand-engineering the macro-structure or executing a second hyperparameter search.

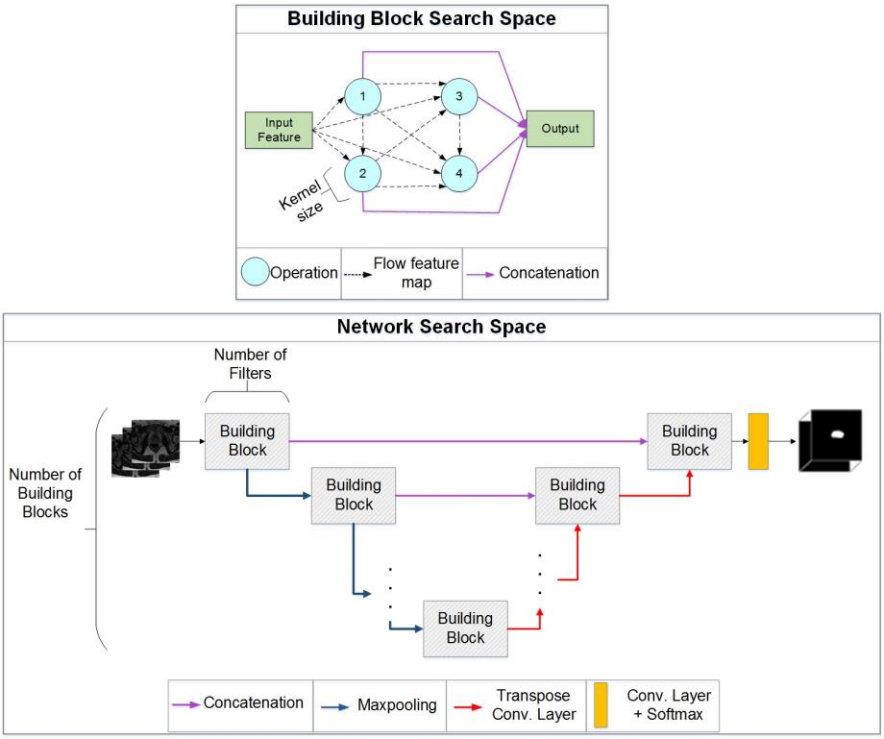

Fig. 18 Overview of the architecture search space.

7.1.2 Search Strategy

The search strategy refers to the method utilized to generate the new child architectures. In evolutionary algorithms, each architecture is encoded as a string (genotype) and the child architectures are obtained by applying cross-over and mutation operations to the parent´s genotype. However, the evolution process is usually slow because random mutation is utilized instead of guiding the search to areas that have shown good results in past iterations. Specifically, in the current MEA algorithm, the mutation probability is independent for each component in the hyperparameter vector and within that component, the selected value is randomly chosen from all possible values in the search space. However, by analyzing the performance of previously tested candidate architectures, a correlation can be found between hyperparameter values and architecture performance. Therefore, the aim is to increase the probability of selecting hyperparameter values that have shown a good performance on the tested architectures and reduce the probability of selecting hyperparameter values that have obtained a poor performance. For this objective, each hyperparameter value will be scored based on the average performance that candidate architectures have on the validation set. Let $S_{h_{ij}}$ be the score for hyperparameter *i* with value *j* where *i* refers to a hyperparameter component (i.e., number of filters, convolutional operation) while *j* refers to a specific value assigned to the hyperparameter component (i.e., if the hyperparameter component is a convolutional operation then the possible hyperparameters values are 3D convolution, 2D convolution or P3D convolution). Then, $S_{h_{ij}}$ can be defined in iteration N:

$$S_{h_{ij}} = \sum_{n=1}^{N-1} \frac{\mathrm{I}(\hat{f}_n) * Acc(\hat{f}_n)}{\mathrm{I}(\hat{f}_n)}$$

Where $Acc(\hat{f}_n)$ is the accuracy of candidate architecture $\hat{f}_n$ tested in iteration n, and $\mathrm{I}(\hat{f}_n)$ is an indicator function such that:

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

Where h_{ij} is the hyperparameter component *i* with value *j*. The probability of selecting hyperparameter value *j* ($P_{h_{ij}}$) during mutation is defined as:

$$P_{h_{ij}} = \frac{S_{h_{ij}}}{\sum_{j \epsilon J} S_{h_{ij}}}$$

Where J is the set that contains all the values of the hyperparameter component *i*. This probability increases as the score or average performance of the hyperparameter h_{ij} increases.

Similar to objective 2, the mutation probability will have a high value at the beginning of the search and decrease monotonically after each generation. This allows the algorithm to initially explore different areas from the search space and afterwards converge to a solution. Also, all hyperparameter values in a component will be initialized with a uniform selection probability ($P_{h_{ij}}$) and updated after each generation. In this manner, all hyperparameter values will have a similar probability of being selected during the initial generations to promote exploration, and significantly differentiate at late generations for favor exploitation.

The other strategy that will be implemented is leading the search to the most promising subproblems. The MOEA/D [77] solves a multiobjective optimization problem (MOP) by decomposing it into a number of scalar optimization subproblems. A single-objective function in each subproblem is obtained by applying a nonlinear weighted aggregation function to the M objective functions $f_1(x)$ ... $f_M(x)$ in the MOP. During evolution, the MOEA/D keeps one solution for each subproblem. A new solution for a subproblem is generated by randomly selecting two solutions from the neighborhood and applying genetic operators. Each subproblem is solved once in each generation and the number of subproblems solved is the same as the size of the population. However, works have shown that based on the characteristics of the MOP problem, some subproblems discover more non-dominated points during the search process than others [136]. Therefore, to exploit the most promising areas, subproblems that have actively contributed to approximate the Pareto Frontier will have a higher probability of being selected to be solved. Hence, a subproblem can be selected more than once for generating a solution at a generation. The probability of choosing subproblem *i* at generation g (PS_{ig}) is determined by:

$$PS_{ig} = \frac{PND_{ig}}{\sum_{i=1}^{N} PND_{ig}}$$

Where PND_{ig} is the proportion of non-dominated solutions generated by subproblem *i* during the previous *L* generations, and *N* is the number of subproblems. Furthermore, the proportion of non-dominated solutions generated by subproblem *i* in the previous *L* generations is computed as:

$$PND_{ig} = \frac{\sum_{k=g-L}^{g-1} ND_{ig}}{\sum_{i=1}^{N} \sum_{g-L}^{g-1} ND_{ig}} + \epsilon, (i = 1,2, \dots, N)$$

Where ND_{ig} is the number of non-dominated solutions generated by subproblem *i* during the previous *L* generations, and $\epsilon = 0.002$ to ensure all $PND_{ig} > 0$. Computing this probability to guide the search of the

MOEA/D has also been used by [136, 137] and has shown to improve the algorithmic performance in terms of convergence speed and quality of the final solution set in comparison with the NSGA-II and original MOEA/D.

7.1.3 Performance Evaluation Strategy

Finally, the performance evaluation strategy refers to the method used to measure the performance of the generated child models. In this work, an inexpensive surrogate function will be applied to estimate the performance of the candidate architectures. Surrogate-based optimization is a well-known strategy used to obtain near optimal solutions when optimizing expensive black-box functions. The main objective is to apply an inexpensive function to approximate the value of an expensive to evaluate function. This is the case of NAS optimization, in which evaluating a candidate architecture requires the expensive process of training it. Therefore, a radial basis function (RBF) model will be used as an inexpensive surrogate to predict an architecture´s validation accuracy. RBFs have shown to have excellent approximation properties independent of the size of the input vector [138]. Hence, it can work in the optimization of high or low dimensional hyperparameter search spaces. Furthermore, RBFs have shown to be successful at approximating the error function of neural networks [139] and also in assisting the MOEA/D algorithm to reduce the computational complexity in the optimization of the Zitzler Deb-Thiele [140] (ZDT) benchmark [141].

7.2 Experiments and Evaluation

The proposed NAS algorithm will be tested in the segmentation of prostate images from the PROMISE12 challenge [80] and cardiac images from the MICCAI ACDC challenge [104]. These datasets will allow the comparison of the proposed algorithm, in terms of search efficiency, with the MEA algorithm. Also, the segmentation performance of the optimal architectures can be evaluated against state-of-the-art networks that have submitted results to the challenges.

The proposed search space and strategies to speed the convergence of the NAS algorithm will be implemented sequentially to assess the performance improvement. First, the proposed search space will be introduced and the MEA algorithm applied to find the optimal architecture. The segmentation performance of the found optimal architecture will be compared against the performance of the ensemble found in objective 2. This will allow to evaluate if the proposed search space increases the accuracy of the constructed architecture. The metrics used to evaluate the accuracy include the dice similarity coefficient, sensitivity, mean surface distance and 95 percentile Hausdorff distance. A one-tailed paired t-test will be applied to verify if the mean performance is statistically different.

Secondly, the new efficient evolutionary based algorithm (with the presented search strategies and RBF surrogate function) will be implemented to find the optimal architectures on the proposed search space. The increase on convergence speed can be assessed by comparing the computational time and number of generations required by the efficient evolutionary based algorithm to obtain the optimal solution compared to the MEA algorithm (tested previously on the same search space). The experiments will be performed on the prostate images from the PROMISE12 dataset as it provides a representative set of images acquired in different clinical settings and the resolution and size of the dataset is small enough to allow various experiments to be executed.

7.3 Expected Outcomes

The expected outcomes of this research objective include: 1) proposing an efficient algorithm for multiobjective neural architecture search that can simultaneously optimize various objective functions while designing deep neural networks, and 2) proposing a NAS method that simultaneously optimize the cell level structure and network level structure for medical image segmentation.

8. DISCUSSION

In this research, deep neural networks that automatically adapt to new datasets and simultaneously search for efficient architectures have been proposed. Furthermore, multiobjective evolutionary based algorithms have been presented to solve the complex hyperparameter optimization problem in deep neural network design.

For the first objective, a CNN for 2D medical image segmentation (AdaResU-Net) that automatically adapts to new datasets and reduces the network size has been presented. The experiments show that the AdaResU-Net architecture with the proposed MEA learning framework (AdaResU-Net MEA) 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 U-Net and ResU-Net in terms of the mean Dice coefficient, 95 percentile Hausdorff distance, and mean surface distance. Also, the results of AdaResU-Net with MEA learning framework and the AdaResU-Net with a Bayesian Optimization approach suggest that the proposed algorithm results in similar or better performance compared to the well-known Gaussian Process Bayesian hyperparameter optimization method, while providing considerably smaller architectures. In terms of the components of the AdaResU-Net, the addition of residual connections help provide a better holistic shape and spatial distribution of the segmented region. This is allows the architecture to have a statistically smaller 95 percentile Hausdorff distance and mean surface distance than the U-Net. Furthermore, implementing the proposed MEA learning framework to tune the hyperparameters 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 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. Also, 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). Finally, 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.

In the second research objective, a self-adaptive 2D-3D ensemble of fully convolutional networks (AdaEn-Net) which incorporates volumetric information while optimizing both the performance and model's size is proposed for 3D medical image segmentation. 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 performed better than a top-ranked ruled-based adaptive framework in the prostate dataset while being considerably smaller and achieving better results than a NAS reinforcement learning based algorithm on the cardiac dataset. Differently from the competing manually and automatically designed networks, the implemented MEA algorithm also aims to design architectures with minimum size. 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 (Tan & Le, 2019). The experiments showed that balancing the dimensions of the architecture is necessary when constructing the 3D FCN.

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. Although the 3D FCN is statistically less accurate than the 2D FCN on both datasets, the final ensemble successfully incorporates the information from the two 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 the work, which is that the proposed ensemble framework can develop a diverse set of models with less correlated errors.

For the third research objective, an efficient multiobjective evolutionary based algorithm for deep neural network design is proposed by guiding the search to more promising areas of the search space and the application of surrogate-based optimization. Additionally, the search space will be increased to include the cell level and structure level optimization in the architecture search. The proposed algorithm will help improve the democratization of AI by reducing the amount of computational resources needed to run the design process. Furthermore, it is expected to obtain more accurate and efficient architectures by increasing the flexibility of the hyperparameter search space, which also improves the application of the models into real-world problems.

Finally, 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. Moreover, the proposed algorithms aim to increase 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.

9. DISSERTATION TIMELINE

	Task	F17	Sp18	Sm18	F18	Sp19	Sm19	F19	Sp20	Sm20
1	**Objective 1:** Design AdaResU-Net Implement AdaResU-Net Evaluate Model									
2	**Objective 2:** Design AdaEn-Net Implement AdaEn-Net Evaluate Model									
3	**Objective 3:** Design NAS algorithm Implement NAS algorithm Evaluate Model									
4	**Proposal Defense**									
5	**Dissertation Defense**									

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] T. Elsken, J. H. Metzen and F. Hutter, "Neural Architecture Search: A Survey," *Journal of Machine Learning Research,* pp. 1-21, 2019.

[6] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578,* 2016.

[7] 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.

[8] K. Kandasamy, W. Neiswanger, J. Schneider, B. Poczos and E. P. Xing, "Neural architecture search with bayesian optimisation and optimal transport," in *Advances in Neural Information Processing Systems*, 2018.

[9] 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.

[10] R. Luo, F. Tian, T. Qin, E. Chen and T.-Y. Liu, "Neural architecture optimization," in *Advances in neural information processing systems*, 2018.

[11] H. Liu, K. Simonyan and Y. Yang, "Darts: Differentiable architecture search," in *arXiv preprint arXiv:1806.09055*, 2018.

[12] 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.

[13] Y. Huang, Y. Cheng, D. Chen, H. Lee, J. Ngiam, Q. V. Le and Z. Chen, "Gpipe: Efficient training of giant neural networks using pipeline parallelism," in *arXiv preprint arXiv:1811.06965*, 2018.

[14] A. Canziani, A. Paszke and E. Culurciello, "An analysis of deep neural network models for practical applications," *arXiv preprint arXiv:1605.07678,* 2016.

[15] M. Denil, B. Shakibi, L. Dinh and N. De Freitas, "Predicting parameters in deep learning," in *Advances in neural information processing systems*, 2013.

[16] 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.

[17] 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.

[18] K. Ahmed and L. Torresani, "Connectivity learning in multi-branch networks," in *arXiv preprint arXiv:1709.09582*, 2017.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] J. Liang, E. Meyerson, B. Hodjat, D. Fink, K. Mutch and R. Miikkulainen, "Evolutionary neural automl for deep learning," in *arXiv preprint arXiv:1902.06827*, 2019.

[24] K. K. Roy and A. Phadikar, "Automated Medical Image Segmentation: A Survey," in *Proc. of Int. Conf. on Computing, Communication & Manufacturing*, 2014.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] N. Ramesh, J. Yoo and I. Sethi, "Thresholding based on histogram," in *IEEE Proc Vision Image Signal Proc*, 1995.

[31] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine learning,* pp. 679-698, 1986.

[32] N. Sharma and A. Ray, "Computer aided segmentation of medical images," in *Proc. of Int. Conf. on Mathematical Biolog*, 2006.

[33] N. Sharma and L. M. Aggarwal, "Automated medical image segmentation techniques," *Journal of medical physics,* vol. 35, p. 3, 2010.

[34] 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.

[35] 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.

[36] R. O. Duda, P. E. Hart, D. G. Stork and others, Pattern classification, Wiley New York, 1973.

[37] 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.

[38] 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.

[39] H. M. Moftah, 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.

[40] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin and H. Larochelle, "Brain tumor segmentation with deep neural networks," *Medical image analysis*, pp. 18-31, 2017.

[41] F. Lu, F. Wu, P. Hu, Z. Peng and D. Kong, "Automatic 3D liver location and segmentation via convolutional neural network and graph cut," *International journal of computer assisted radiology and surgery*, 2017.

[42] L. Yu, X. Yang, H. Chen, J. Qin and P.-A. Heng, "Volumentric ConvNets with Mixed Residual Connections for Automated Prostate Segmentation from 3D MR Images," in *AAAI*, 2017.

[43] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[44] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, pp. 834-848, 2018.

[45] 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.

[46] 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.

[47] S. Jegou, M. Drozdzal, 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.

[48] 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.

[49] M. Baldeon-Calisto and S. Lai-Yuen, "ResU-Net: Residual Convolutional Neural Network for Prostate MRI segmentation.," in *IISE Annual Conference*, Orlando, 2018.

[50] 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.

[51] 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.

[52] 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.

[53] 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.

[54] 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.

[55] 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.

[56] 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.

[57] 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.

[58] 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.

[59] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary computation*, pp. 99-127, 2002.

[60] 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.

[61] 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.

[62] 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 Laudardale, 2017.

[63] 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.

[64] E. Thomas, H. M. Jan and H. Frank, "Simple And Efficient Architecture Search for Convolutional Neural Networks," in *ICLR*, 2017.

[65] A. Brock, T. Lim and J. M. a. W. N. Ritchie, "SMASH: one-shot model architecture search through hypernetworks," in *arXiv preprint arXiv:1708.05344*, 2017.

[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] 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: Self-adapting framework for u-net-based medical image segmentation," *arXiv preprint arXiv:1809.10486*, 2018.

[69] Z. Zhu, C. Liu, D. Yang, A. Yuille and D. Xu, "V-NAS: Neural Architecture Search for Volumetric Medical Image Segmentation," in *arXiv preprint arXiv:1906.02817*, 2019.

[70] 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 *arXiv preprint arXiv:1906.05956*, 2019.

[71] 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.

[72] 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.

[73] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural networks: Tricks of the trade*, Berlin, Springer, 2012, pp. 437-478.

[74] O. Oktay, E. Ferrante, K. Kamnitsas, M. Heinrich, W. Bai, J. Caballero, S. A. Cook, A. de Marvao, T. Dawes, D. P. O'Regan, B. Kainz, B. Glocker and D. Rueckert, "Anatomically constrained neural networks (ACNNs): application to cardiac image enhancement and segmentation," *IEEE Transactions on Medical Imaging*, vol. 37, pp. 384-395, 2018.

[75] 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.

[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. 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.

[80] 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.

[81] 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.

[82] 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.

[83] 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.

[84] S. Onal, X. Chen, S. K. Lai-Yuen and S. Hart, "Automatic vertebra segmentation on dynamic magnetic resonance imaging," *Journal of Medical Imaging,* 2017.

[85] 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.

[86] F. Chollet, "Keras," 2015. [Online]. Available: https://github.com/keras-team/keras.

[87] J. Snoek, L. Hugo and R. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems*, 2012.

[88] P. F. Christ, F. Ettlinger, 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.

[89] 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," *arXiv preprint arXiv:1709.00201,* 2017.

[90] C. Wang, X. Zhang, Y. Chen and K. Lee, "vU-net: accurate cell edge segmentation in time-lapse fluorescence live cell images based on convolutional neural network," *bioRxiv: 191858.,* 2017.

[91] J. Gonzalez and Z. Dai, *GPyOpt: A Bayesian Optimization framework in Python,* http://github.com/SheffieldML/GPyOpt, 2016.

[92] 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.

[93] 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.

[94] B. Sciolla, M. Martin and P. Delachartre, "Multi-pass 3D convolutional neural network segmentation of prostate MRI images}," 2017.

[95] 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.

[96] 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.

[97] 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.

[98] 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.

[99] 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.

[100] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation,* pp. 61-70, 2011.

[101] 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.

[102] Z. Zhang, Q. Liu and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters,* 2018.

[103] K. He, X. Zhang, S. Ren and J. Sun, "Identity mappings in deep residual networks.," in *European Conference on Computer Vision*, 2016.

[104] 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.

[105] M. Baldeon and S. Lai-Yuen, "Adaresu-net: Multiobjective adaptive convolutional neural network for medical image segmentation," *Neurocomputing,* 2019.

[106] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167,* 2015.

[107] G. Lin, A. Milan, C. Shen and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[108] 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.

[109] N. Srivastava, G. K. A. Hinton, I. Stuskever and R. Salakhutdinov, "Dropout: A simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research* , 2014.

[110] T. G. Dietterich, "Ensemble learning," *The handbook of brain theory and neural networks,* vol. 2, pp. 110-15, 2002.

[111] 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.

[112] 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.

[113] G. Giacinto and F. Roli, "Design of effective neural network ensembles for image classification purposes," *Image and Vision Computing,* vol. 19, pp. 699-707, 2001.

[114] A. Kumar, J. Kim, D. Lyndon, M. Fulham and D. Feng, "An ensemble of fine-tuned convolutional neural networks for medical image classification," *IEEE journal of biomedical and health informatics,* vol. 21, pp. 31-40, 2016.

[115] 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.

[116] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis & Machine Intelligence,* pp. 993-1001, 1990.

[117] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Advances in neural information processing systems*, 1995.

[118] B. Parmanto, P. W. Munro and H. R. Doyle, "Improving committee diagnosis with resampling techniques," in *Advances in neural information processing systems*, 1996.

[119] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *arXiv preprint arXiv:1605.07146*, 2016.

[120] M. Telgarsky, "Benefits of depth in neural networks," in *arXiv preprint arXiv:1602.04485*, 2016.

[121] S. Liang and R. Srikant, "Why deep neural networks for function approximation?," in *arXiv preprint arXiv:1610.04161*, 2016.

[122] C. Petitjean, M. A. Zuluaga, W. Bai, J.-N. Dacher, D. Grosgeorge, J. Caudron, S. Ruan, I. B. Ayed, M. J. Cardoso, H.-C. Chen and others, "Right ventricle segmentation from cardiac MRI: a collation study," *Medical image analysis,* vol. 19, pp. 187-202, 2015.

[123] 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.

[124] 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.

[125] 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.

[126] 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.

[127] 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.

[128] 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.

[129] 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.

[130] 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.

[131] 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.

[132] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *arXiv preprint arXiv:1905.11946*, 2019.

[133] E. Real, A. Aggarwal, Y. Huang and Q. V. Le, "Regularized evolution for image classifier architecture search," in *arXiv preprint arXiv:1802.01548*, 2018.

[134] 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.

[135] E. Real, A. Aggarwal, Y. Huang and Q. Le, "Aging Evolution for Image Classifier Architecture Search," in *AAAI Conference on Artificial Intelligence*, 2019.

[136] 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.

[137] S.-Z. Zhao, P. N. Suganthan and Q. Zhang, "Decomposition-based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 3, pp. 442-446, 2012.

[138] M. D. Buhmann, "Radial basis functions," *Acta numerica*, vol. 9, pp. 1-38, 2000.

[139] I. Ilievski, T. Akhtar, J. Feng and C. A. Shoemaker, "Efficient Hyperparameter Optimization for Deep Learning Algorithms Using Deterministic RBF Surrogates.," in *AAAI*, 2017.

[140] E. Zitzler, K. Deb and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary computation*, vol. 8, pp. 173-195, 2000.

[141] S. Zapotecas Martinez and C. A. Coello Coello, "MOEA/D assisted by RBF networks for expensive multi-objective optimization problems," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, 2013.