Some Metaheuristic Algorithms to Design Deep Neural Networks for Medical Image Detection

Submitted in partial fulfillment of the requirements

for the award of the degree of

DOCTOR OF PHILOSOPHY

Submitted by

Chilukamari Rajesh

(Roll No. 720084)

Under the supervision of

Prof. Ravichandra Sadam

and

Prof. Sushil Kumar

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
TELANGANA - 506004, INDIA
JANUARY 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
TELANGANA - 506004, INDIA

THESIS APPROVAL FOR Ph.D.

This is to certify that the thesis entitled, Some Metaheuristic Algorithms to Design Deep Neural Networks for Medical Image Detection, submitted by Mr. Chilukamari Rajesh [Roll No. 720084] is approved for the degree of DOCTOR OF PHILOSOPHY at National Institute of Technology Warangal.

Examiner

Research Supervisor
Prof. Ravichandra Sadam
Dept. of Computer Science and Engg.
NIT Warangal
India

Research Supervisor
Prof. Sushil Kumar
Dept. of Computer Engg.
NIT Kurukshetra
India

Chairman
Prof. Ch. Sudhakar
Dept. of Computer Science and Engg.
NIT Warangal
India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
TELANGANA - 506004, INDIA

CERTIFICATE

This is to certify that the thesis entitled, Some Metaheuristic Algorithms to Design Deep Neural Networks for Medical Image Detection, submitted in partial fulfillment of requirement for the award of degree of DOCTOR OF PHILOSOPHY to National Institute of Technology Warangal, is a bonafide research work done by Mr. Chilukamari Rajesh [Roll No. 720084] under our supervision. The contents of the thesis have not been submitted elsewhere for the award of any degree.

Research Supervisor
Prof. Ravichandra Sadam
Dept. of Computer Science and Engg.
NIT Warangal
India

Research Supervisor
Prof. Sushil Kumar
Dept. of Computer Engg.
NIT Kurukshetra
India

Place: NIT Warangal

Date: 03 January, 2024

DECLARATION

This is to certify that the work presented in the thesis entitled "*Some Metaheuristic Algorithms to Design Deep Neural Networks for Medical Image Detection*" is a bonafide work done by me under the supervision of Prof. Ravichandra Sadam and Prof. Sushil Kumar was not submitted elsewhere for the award of any degree.

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/date/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Chilukamari Rajesh
(Roll No. 720084)
Date: 03-01-2024

ACKNOWLEDGMENTS

I would like to express my sincere gratitude and appreciation to my supervisors, Prof. Ravichandra Sadam and Prof. Sushil Kumar, for their invaluable guidance throughout the completion of this work. Their continuous support, timely feedback, and constructive discussions have played a pivotal role in helping me achieve my objectives. I am grateful for the ample time they dedicated to reviewing my work and providing insightful suggestions for improvement. Their mentorship not only shaped me as a researcher but also as an individual. I have been inspired by their words, actions, and values, which have demonstrated the qualities of a great teacher and a compassionate human being. Their unwavering dedication and commitment to excellence have left a profound impact on me. I aspire to embody these remarkable qualities throughout my life.

I would like to express my heartfelt gratitude to all the members of my Doctoral Scrutiny Committee (DSC), namely Prof. Ch. Sudhakar, Prof. Sujit Das, Prof. Mettu Srinivas, and Prof. Ravi Kumar Jatoth. Their valuable comments and suggestions during the oral presentations have greatly enriched my research work. I am truly fortunate to have had the opportunity to attend lectures by esteemed professors such as Prof. P. Radha Krishna, Prof. Sanjeevi, Prof. P. Venkata Subba Reddy, Prof. U.S.N. Raju and Prof. Madhavi Reddy Kesari. Their knowledge and expertise have been instrumental in broadening my understanding in the field.

I am immensely thankful to Prof. P. Radha Krishna, Prof. Ravichandra Sadam, and Prof. R. Padmavathy Heads of Dept. of CSE and chairmans of DSC, during my tenure for providing adequate facilities in the department to carry out the oral presentations. I wish to express my thanks to all the esteemed faculty members of the Department of Computer Science and Engineering at NIT Warangal. I would also like to extend my heartfelt gratitude to Prof. N.V. Ramana Rao and Prof. Bidyadhar Subudhi, the Director of NIT Warangal, for their unwavering support and encouragement throughout my research endeavor. I am truly fortunate to have been part of such a remarkable institution and to have received support from all these individuals who have contributed to my academic and personal development.

I would like to express my heartfelt gratitude to my seniors, Vinay Raj, Hemkumar, Amilpur Santhosh, and B. Umamaheswara Sharma, as well as my co-scholars, Medipaly Rampavan, G Punnam Chander, V Dilip Kumar, Prakash, Satya Sekar, Asif M.d, Murukesan, Rajasekar, and A Srinivas for their invaluable support and selfless cooperation throughout my Ph.D. journey. Their presence and assistance have been truly remarkable, and I am deeply thankful for their unwavering support and readiness to help me at any time. In particular, I would like to express a special thanks to Rampavan, Punnam Chander, and Dilip Kumar for their unconditional love, support, and constant presence in both the joyful and challenging moments of my personal life and research work. Their friendship has been a source of strength and motivation, and I am forever grateful to have such amazing individuals by my side. Without the support and camaraderie of these remarkable individuals, my Ph.D. journey would have been much more challenging. I am truly blessed to have such incredible friends in my life, and I will cherish their friendship and support forever.

I am immensely grateful to my mother, Chilukamari Swarupa, my father, Chilukamari Sammaiah, and my brother, Chilukamari Venkatesh, for their unwavering love, support, and prayers throughout my journey. Their constant encouragement and support have been my pillars of strength, providing me with the confidence and motivation to overcome challenges and pursue excellence.

Chilukamari Rajesh

Dedicated to

Family, Friends & Teachers

ABSTRACT

Medical image analysis plays a critical role in modern healthcare for treatment planning, diagnosis, and disease prediction, including brain tumors. It encompasses various modalities, including X-ray, Computed Tomography (CT), and Magnetic Resonance Imaging (MRI), each with unique strengths and applications. Analyzing these complex images pose challenges due to noise, and image quality variability, making them prone to errors and heavily reliant on the knowledge and experience of physicians. Medical imaging tasks, such as denoising, aim to improve image quality for precise diagnosis, while accurate segmentation enables quantitative analysis and visualization of anatomical abnormalities, with early detection of brain tumors crucial for timely intervention and improved patient outcomes. Deep Neural Networks (DNNs) have shown promising results in medical image analysis. However, manually designing DNN models is challenging, tedious, time-consuming, and requires domain-specific knowledge. The increasing number of available training techniques adds complexity to finding the optimal structure, and selecting the suitable hyperparameters for a given task often entails multiple trial-and-error iterations. To address these challenges, Neural Architecture Search (NAS) has emerged as a promising solution, which automates the design of DNNs for specific tasks. Nevertheless, NAS methods need further optimization in designing a search space, constructing a DNN from search space (encoding), and exploring different search strategies for specific tasks. The Metaheuristic Algorithm (MA) based methods in NAS have gained traction for automating the DNN architecture design process. This thesis proposes automatically designing flexible and efficient DNN architectures and hyperparameters using MAs. Integrating metaheuristic optimization with deep learning can lead to adaptable and practical solutions for medical image analysis. Flexible search spaces, advanced techniques, and consideration of computational resources contribute to developing practical solutions. The proposed metaheuristic optimization framework has the potential to revolutionize medical image analysis, enhancing patient care by enabling better diagnosis, treatment planning, and research in medical imaging.

The main objectives of this thesis include: (i) To design a metaheuristic block-based

deep neural network for medical image denoising, (ii) To develop a metaheuristic-based modified U-shaped network for 2D medical image segmentation with denoising, (iii) To develop a metaheuristic based encoder-decoder model for 3D medical image segmentation, and (iv) To design a multi-objective metaheuristic model for detecting brain tumors in 3D medical images.

In this thesis, to achieve the abovementioned objectives, we proposed some metaheuristic-based approaches for medical image analysis tasks, including denoising, segmentation, and brain tumor detection. Firstly, a metaheuristic block-based deep neural network is designed for medical image denoising. The denoising performance is enhanced by utilizing the Differential Evolution (DE) algorithm, which facilitates the exploration of various combinations of network components and hyperparameters within the specified search space. Secondly, a metaheuristic-based modified U-shaped network is developed for 2D medical image segmentation with denoising. A modified U-shaped architecture is used with a flexible search space that allows the optimization of individual blocks. Furthermore, attention blocks are incorporated to enhance segmentation accuracy. The Teaching-Learning-Based Optimization (BTLBO) algorithm is used for optimization, resulting in improved segmentation performance. Thirdly, a metaheuristic-based encoder-decoder model is developed for 3D medical image segmentation. A powerful search space is constructed to optimize the network blocks and training parameters. The Chameleon Search Algorithm (CSA) explores the search space to improve the segmentation performance. Lastly, the third objective is extended to brain tumor detection using a multi-objective optimization approach to optimize detection performance and network size. The search space is expanded to include various blocks and parameters. The Multi-objective Iterative Teaching-Learning-Based Optimization (MO-ITLBO) algorithm is utilized to identify optimal block structures and training parameters. The experimental results of this research demonstrate the effectiveness of metaheuristic optimization techniques in enhancing various tasks of medical image analysis. The proposed models outperform existing methods, offering improved denoising, segmentation, and brain tumor detection performance. These advancements can revolutionize medical image analysis, leading to better patient care, diagnosis, and treatment planning in medical imaging.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

BTLBO	Binary Teaching-Learning-Based Optimization
CNN	Convolutional Neural Network
CSA	Chameleon Swarm Algorithm
CT	Computed Tomography
DE	Differential Evolution
DNN	Deep Neural Network
ED	Encoder Decoder
GA	Genetic Algorithm
MA	Metaheuristic Algorithm
MO-ITLBO	Multi-objective Improved Teaching-Learning-Based Optimization
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
NAS	Neural Architecture Search
PF	Pareto Front
PSNR	Peak Signal to Noise Ratio
RL	Reinforcement Learning

Chapter 1

Introduction

Medical image analysis plays a critical role in modern healthcare for treating, diagnosing, and predicting diseases, such as brain tumors. Medical images differ from natural images as they exhibit heterogeneous appearances due to variations in acquisition protocols, imaging equipment, and the characteristics of the anatomical structures of interest regarding location, size, and shape. Medical imaging encompasses different modalities like X-ray, Computed Tomography (CT), and Magnetic Resonance Imaging (MRI), each with its own strengths and applications. For instance, X-rays use electromagnetic radiation to generate 2D images, commonly utilized for bone and lung examinations. CT scans combine X-rays and computer processing to provide cross-sectional views of bones, organs, and blood vessels. MRIs employ magnetic fields and radio waves to produce detailed images, ideal for visualizing soft tissues and diagnosing conditions in the brain, spinal cord, and joints [1]. 2D medical images, like X-rays and single-slice CT scans, are easy to acquire and suitable for initial screenings, emergency situations, and routine monitoring. In contrast, 3D medical images, like volumetric CT scans or MRI sequences, offer detailed anatomical information, supporting surgical guidance, quantitative analysis, and advanced research. The choice of imaging modality depends on the specific medical condition and the information required by healthcare professionals. Analyzing these complex images poses significant challenges due to various factors, including noise, variability in image quality, and the need for precise segmentation, prediction of pathological structures and tumor detection [2, 3]. However, addressing these tasks in medical images is time-consuming, prone to er-

rors, and heavily reliant on physicians' knowledge and experience. Therefore, developing accurate, reliable, and automatic brain tumor detection algorithms holds immense clinical significance [4]. Two approaches are commonly used in medical image analysis: traditional and deep learning.

Traditional methods used in medical image tasks include threshold-based [5], region-based [6], and machine learning [7] techniques. Threshold-based methods involve assigning pixels to different classes based on an intensity threshold value. Selecting an appropriate threshold value is crucial for accurate segmentation results. Region-based methods use algorithms like watershed and region-growing to identify distinct regions but can be sensitive to noise and introduce holes in extracted regions. Machine learning techniques like Clustering, Support Vector Machines (SVM), and Random Forest (RF) approaches are less suitable for large-scale data [8, 9].

The advancement of deep learning and computer vision algorithms have revolutionized medical image analysis, resulting in significant advancements in various tasks. The Convolutional Neural Networks (CNNs) have emerged as one of the most popular deep learning methods, showing promising results in denoising, segmentation, and tumor detection tasks. Deep Neural Networks (DNNs) have gained significant popularity in various fields due to their ability to learn complex patterns and make accurate predictions in medical image analysis. However, the process of designing a DNN architecture involves determining the number of layers, the number of neurons in each layer, and the connectivity patterns between the layers. These decisions significantly impact the performance and efficiency of the network. A poorly designed architecture may result in suboptimal performance or even failure to learn from the data. In addition to architecture design, selecting appropriate hyperparameters is crucial for training a DNN effectively. Hyperparameters include learning rate, batch size, regularization techniques, activation functions, and others. The choice of hyperparameters can greatly influence the training dynamics and the final performance of the network. With the advancements in deep learning, numerous training techniques and optimization algorithms have been developed. Each technique may have specific requirements and constraints on the network architecture and hyperparameters. Consequently, finding the optimal structure and selecting the best hyperparameters for a given task often

involves multiple iterations of trial-and-error experiments. The iterative process of trying different architectures and hyperparameters adds complexity to the task [10]. Researchers and practitioners need to spend a significant amount of time experimenting and fine-tuning the network to achieve satisfactory results. The manual exploration can be a tedious and resource-intensive process, especially when dealing with large-scale datasets or complex tasks. To address these challenges, automated approaches are needed to design efficient DNN architectures and determine the appropriate hyperparameters for specific task [11].

Neural Architecture Search (NAS) has emerged as a promising solution for architecture search and hyperparameter optimization [12, 13]. NAS aims to discover the best network architecture and hyperparameters for a given task through automated exploration of various architectures. These automated methods help in reducing the human effort required and can potentially discover more effective architectures and hyperparameters than manual design [12]. In general, the NAS can be formulated as follow equations 1.1, 1.2:

$$\mathcal{A}^* = argmin_{\mathcal{A}\in\mathbb{S}}\, f(\mathcal{A}, \omega^*(\mathcal{A})) \tag{1.1}$$

where,

$$\omega^*(\mathcal{A}) = argmin\, \mathcal{L}_T(\mathcal{A}, \omega) \tag{1.2}$$

In equation 1.1, the search space denoted by $\mathbb{S}$ contains all possible architectures from which the optimal architecture $\mathcal{A}^*$ is selected. The search space can be defined based on specific architectural constraints and design choices. The objective function $f(\mathcal{A}, \omega^*(\mathcal{A}))$ evaluates the performance of an architecture $\mathcal{A}$ with its learned parameters $\omega^*(\mathcal{A})$. The objective function can be defined based on the specific task and desired optimization criteria. In equation 1.2, $\omega^*(\mathcal{A})$ represents the optimal learned parameters for a given architecture $\mathcal{A}$ obtained by minimizing the loss function $\mathcal{L}_T(\mathcal{A}, \omega)$, which measures the error between the predictions made by the architecture $\mathcal{A}$ with parameters ω and the ground truth labels or targets of the given task T.

The NAS problem is typically solved using an iterative process that involves evaluating the performance of a set of candidate architectures, selecting the most promising ones, and generating new architectures based on the insights gained from the evaluation. This

process can be guided by different search strategies, such as Reinforcement Learning (RL) [14], Metaheuristic Algorithms (MA) [15], or Gradient-based optimization [16]. The MAs are preferred for NAS in computer vision over RL and gradient-based learning techniques due to their numerous advantages such as flexibility, robustness, and efficiency [13]. For example, MAs offer greater flexibility, allowing for the exploration of a broader range of network structures and configurations. They are robust to factors like random weight initialization and hyperparameter choices, resulting in less susceptibility to suboptimal architectures. Additionally, MAs efficiently search large architectural spaces, avoiding local optima that RL and gradient-based methods may get stuck in due to insufficient guidance from gradients or rewards.

Therefore, MAs are employed as search methods for NAS in this research. The process of metaheuristic algorithms based NAS (MetaNAS) can be summarized as follows:

1. Initialize a population of DNN architectures randomly. Each individual in the population represents a unique DNN architecture.

2. Evaluate each individual's fitness in the population by training and evaluating the performance of the corresponding DNN.

3. Select the best-performing individuals from the population and use them to create a new population by applying MAs operations.

4. Repeat steps 2 and 3 for a fixed number of generations or until convergence is achieved.

MetaNAS is usually modelled as an optimization problem, where the aim is to optimize the fitness function on a given search space. The fitness function can be defined based on the specific task and evaluation metric, such as accuracy, F1 score, mean squared error etc. The operations used in the MAs can be designed to promote diversity in the population and explore the search space efficiently. In the exploration process, the MetaNAS can gradually converge to a set of high-performing DNN architectures for the given task. The architectures discovered by these approaches have outperformed manual-designed architectures in

high-level vision tasks like image classification, segmentation and detection, etc. However, these algorithms need to be further optimized in terms of designing a search space, constructing a DNN from search space (encoding) and exploring different metaheuristic algorithms for specific tasks. Metaheuristic algorithm-based models have been rapidly increasing to automate the DNN architecture design process [13].

1.1 Motivation & Objectives

1.1.1 Motivation

Medical image analysis poses significant challenges in removing noise, accurate segmentation, and tumor detection. Medical images often have noise that can affect image quality and robustness of detection methods [17]. Complex mixed noise, such as photon starvation artifacts, may appear in CT perfusion images due to factors like poor enlightenment, camera sensor cells, and transmission issues. This noise poses a significant challenge for tasks like segmentation, detection, and classification, compromising the performance and robustness of the results. Medical image denoising reduces noise and enhances image quality, leading to a more accurate diagnosis. Accurate segmentation in medical imaging is crucial for precise measurements, quantitative analysis, and enhanced visualization of anatomical abnormalities. 2D medical image segmentation focuses on delineating and identifying specific structures or regions of interest, while 3D medical image segmentation offers a comprehensive understanding of anatomical structures in three-dimensional space. Early detection of brain tumors holds significant importance, as it enables timely intervention and ultimately improves patient outcomes. Tumor detection assists radiologists in expediting the diagnosis process, reducing interpretation time, minimizing potential errors, and facilitating efficient treatment planning for better patient care. Designing DNN architectures manually for these tasks involves finding the optimal network structure and hyperparameters, which is complex and time-consuming, often requiring multiple trial-and-error iterations. Manual exploration requires domain-specific knowledge and is a tedious and resource-intensive process, particularly when dealing with large-scale datasets or complex tasks [18].

The motivation behind this research stems from the need to overcome the limitations of manually building DNNs. The goal is to automatically design efficient DNN architectures and optimize hyperparameters to enhance the performance of denoising, segmentation, and brain tumor detection by leveraging the power of MAs. As per the "no free lunch" (NFL) theorem [19], there is no single optimizer that can effectively handle and outperform all types of optimization tasks, which highlights the importance of using different MAs with broader applicability to address the different natures of optimization problems. Furthermore, the integration of metaheuristic optimization techniques with deep learning approaches can pave the way for more adaptable and practical solutions for medical image analysis. Additionally, exploring flexible search spaces, integrating advanced techniques, and considering computational resources contribute to developing more adaptable and practical solutions for medical image analysis [15]. The development of such a metaheuristic optimization framework has the potential to revolutionize medical image analysis and improve patient care. The results of this research hold great potential for improving diagnosis, treatment planning, and medical research in the field of medical imaging.

1.1.2 Objectives

The following objectives formulated in this thesis are:

Objective 1: To design a metaheuristic block-based method that presents a novel approach to design deep neural network architectures and training parameters specifically for medical image denoising. A comprehensive search space is designed to achieve this with activation functions, network blocks, and optimizers. The Differential Evolution (DE) algorithm facilitates efficient exploration and optimization within the search space.

Objective 2: This objective aims to develop a metaheuristic approach to 2D medical image segmentation. The optimal block structures in the modified U-shaped network are discovered by utilizing the Binary Teaching-Learning-Based Optimization (BTLBO) algorithm. Additionally, the performance of 2D medical image segmentation is enhanced by incorporating denoising as a preprocessing step.

Objective 3: To design a metaheuristic encoder-decoder-based model tailored for 3D med-

ical image segmentation, leveraging insights gained from the previous objective on 2D segmentation. The efficient exploration of the search space and identification of optimal network structures and parameters for the development of high-performance 3D CNN models is accomplished through the utilization of the Chameleon Swarm Algorithm (CSA) algorithm.

Objective 4: This objective extends the previous objective by expanding the search space into developing a metaheuristic-based multi-objective model for brain tumor detection in 3D images. A Multi-Objective Improved Teaching-Learning-Based Optimization (MO-ITLBO) algorithm is employed to simultaneously optimize performance and model size.

1.2 Summary of the contributions

In this section, an overview of chapter-wise contributions to this thesis has been presented. Each subsection presents a summary of the contributions of the corresponding chapter.

1.2.1 To design a metaheuristic block-based deep neural network for medical image denoising.

In this work, a metaheuristic block-based deep neural network is designed for medical image denoising. This objective focuses on developing a specialized architecture tailored for denoising medical images.

To enable a comprehensive exploration of model hyperparameters, a search space is designed with essential components such as network blocks, activation functions, number of filters, batch normalization, number of blocks, and number of convolution layers in a block. Additionally, training parameters, including loss function, optimizers, and learning rate, are incorporated into the search space to facilitate the training of the architecture. These hyperparameters are treated as vector parameters within the evolutionary process, enabling us to explore diverse combinations and configurations. Then the designed search space is explored using the DE algorithm [20] by treating these parameters as vector parameters to search for optimal network parameters for designing CNN architectures.

Furthermore, a transfer learning technique is incorporated to update the search space during the evolutionary process dynamically. The metaheuristic algorithm is guided toward promising regions in the search space using knowledge gained from previously evolved architectures. This dynamic update accelerates the optimization rate of evolution, leading to the discovery of optimal network parameters for medical image denoising. The proposed model is evaluated on four medical image datasets, and the results demonstrate its superior performance compared with existing denoising models. The DE-based optimization framework significantly improves the quality of medical image denoising, with potential applications in various medical fields and improving patient care.

1.2.2 To develop a metaheuristic based modified U-shaped network for 2D medical image segmentation with denoising.

This work aims to improve the performance of 2D medical image segmentation by integrating denoising (from Objective-1) as a preprocessing step and introducing a modified U-shaped architecture. The proposed method combines denoising and segmentation capabilities to accurately identify and delineate structures and regions of interest within medical images.

A modified U-shaped architecture is proposed as the backbone network for the segmentation task, taking inspiration from the encoder-decoder framework of U-Net [21]. In order to enhance the flexibility and adaptability of the U-shaped network, a search space is introduced that enables the optimization of each block within the proposed U-shaped architecture. This search space consists of 32 node operations divided into pre-activation nodes, where the activation function is applied before the convolution operation, and post-activation nodes, where the activation function is applied after the convolution operation.

The search space encompasses different node operations with various convolutional kernel sizes, activation functions, normalization functions, pooling layers, and loss functions. Furthermore, attention blocks are used instead of addition or concatenation operations during upsampling in U-shaped architecture to identify complex structures within the input images and improve segmentation performance. These attention blocks enable the

network to focus on salient features and essential regions during segmentation.

The BTLBO algorithm [22] is employed to optimize the block structures within the U-shaped network by exploring the search space. A cache mechanism that avoids redundant evaluations further enhances the algorithm's efficiency. The proposed model is evaluated and compared with state-of-the-art models on various 2D medical image datasets. The results demonstrate its superiority in terms of performance and indicate its potential for clinical applications.

1.2.3 To develop a metaheuristic based encoder-decoder model for 3D medical image segmentation.

This work focuses on improving the segmentation performance of three-dimensional medical images by employing an encoder-decoder CNN architecture with a metaheuristic optimization process.

The encoder-decoder architecture used in a previous objective is adapted for the 3D segmentation task. Unlike previous approaches that use a uniform block structure throughout the encoder-decoder architecture, this method individually optimizes the blocks within the network using a metaheuristic algorithm. This approach enables improved segmentation performance and the development of customized 3D medical image segmentation models, leveraging the network's flexibility to adapt to diverse medical imaging datasets.

To improve the segmentation performance, a powerful search space is designed with network parameters, such as blocks, activation functions, normalization layers, up-sample layers, as well as training parameters, including optimizers and loss functions. Then, the CSA algorithm [23] is employed to explore different combinations and variations from the designed search space that can effectively enhance the performance of the encoder-decoder architecture for 3D medical image segmentation.

The efficacy of the proposed model is evaluated using two 3D medical image segmentation datasets. The experimental results show its superior performance in comparison to existing models. The framework holds substantial potential for advancing segmentation outcomes, ultimately leading to more precise and reliable analyses for medical profession-

als.

1.2.4 To design a multi-objective metaheuristic based deep learning model for brain tumor detection.

In this work, the Objective-3 work is extended to a detection task with denoising (Objective-1) as a preprocessing step, aiming to improve detection performance and reduce the model size by employing a multi-objective optimization approach for the detection of brain tumors in 3D medical images.

The search space is expanded to include efficient-net, transformer, and convolution meet transformer blocks to facilitate improved brain tumour detection, incorporating top-performing blocks from the previous segmentation task and training hyperparameters. To effectively explore this search space, a multi-objective ITLBO algorithm [24] is employed to identify optimal block structure in the encoder-decoder architecture along with training parameters. This algorithm considers multiple objectives, aiming to simultaneously optimize both the detection performance and network size, resulting in the development of more effective models.

The proposed method is evaluated using two brain tumor datasets and demonstrates superior performance compared to existing models. Through the application of multi-objective metaheuristic optimization, brain tumor detection of 3D medical images is enhanced by simultaneously optimizing detection performance and model size. This advancement can improve patient care by enabling efficient and accurate predictions, particularly in environments with limited resources.

1.3 Organization of the Thesis

The main focus of this thesis is to design a metaheuristic framework to design deep neural networks for 3D medical image detection. The thesis consists of seven chapters, including an introduction, a literature survey, and a conclusion. The content of each of these chapters is described briefly below:

Chapter 1: This chapter provides a comprehensive introduction to the thesis topic and briefly outlines the objectives that are pursued throughout the research.

Chapter 2: In this chapter, an overview of state-of-the-art works in the field of medical image analysis is provided, with a specific focus on brain tumor detection and other related tasks, including denoising, segmentation, and 3D segmentation with NAS.

Chapter 3: In this chapter, a metaheuristic block-based model employing the DE algorithm for medical image denoising is introduced. Transfer learning technique is incorporated for dynamically updating the search space to accelerate the rate of evolution and optimization. Comprehensive experimental evaluations demonstrate that the proposed model effectively reduces noise and artifacts, significantly improving the quality of medical images.

Chapter 4: This chapter proposes a metaheuristic-based framework for 2D medical image segmentation by incorporating denoising as a preprocessing step. A modified U-shaped network is introduced, which integrates attention blocks and cache mechanisms. The optimal block structure within the network is determined using the metaheuristic BTLBO algorithm. Experimental results demonstrated that the proposed approach has improved the segmentation performance compared to existing 2D medical image segmentation models.

Chapter 5: This chapter presents a metaheuristic encoder-decoder-based model designed specifically for 3D medical image segmentation, leveraging insights gained from previous work on 2D segmentation. The flexibility and performance of the model are enhanced by introducing a comprehensive search space encompassing various blocks within the encoder-decoder CNN architecture. The CSA algorithm efficiently explores the search space and identifies optimal network structures and parameters for creating highly accurate 3D CNN models. The comparative analysis demonstrates that the proposed metaheuristic model significantly improves the performance of 3D medical image segmentation.

Chapter 6: This chapter extends the third objective by introducing efficient-net, transformer, and convolution meet transformer blocks into the search space for brain tumor detection in 3D images. Then, a MO-ITLBO algorithm is utilized to optimize both tumor detection performance and model size of a 3D CNN model. The results demonstrate that the proposed approach enhances detection performance and reduces model size, which is

particularly beneficial for applications with limited computational resources.

Chapter 7: This chapter provides conclusions of the thesis, essential outcomes of the contributions, and the scope for future expansion of the research conducted in this thesis.

Chapter 2

Related Work

A comprehensive literature review of different tasks in medical image analysis is presented in this chapter. The literature related to medical image denoising is discussed in Section 2.1, while Section 2.2 covers the literature on 2D medical image segmentation. Furthermore, Section 2.3 focuses on the literature related to 3D medical image segmentation and detection. Finally, a summary is provided in Section 2.4.

2.1 Medical image denoising

Various traditional methods have been proposed in the literature for medical image denoising. In the Spatial Domain, techniques like the Median Filter [25], Bilateral Filter [26], and Non-local Means Filtering [27] have been utilized. However, these methods often result in image blurring and compromise image resolution. In contrast to Spatial Domain Filters, Transform Domain Filters in image denoising, first transform the given noisy image to another domain, and then apply denoising procedure on the transformed image. Transform Domain Filters, such as Wavelet-based thresholding [28], Wavelet Sub-band Coefficient Mixing (WSM) [29], and the Contourlet Transform [30], offer improved edge preservation but require prior knowledge of the noise present in the image. George et al. [31] conducted a study where they explored the use of various wavelet transforms to enhance the quality and reduce noise levels in medical images.

Convolutional Neural Networks (CNNs) have emerged as a prominent approach for

automatic feature extraction in medical image analysis. Agostinelli et al. [32] utilized adaptive multi-column deep neural networks for medical image denoising. Vincent et al. [33] demonstrated the effectiveness of stacking denoising autoencoders to create deep networks, where the output of one denoising autoencoder serves as input for the subsequent one. Gondara et al. [34] developed denoising autoencoders employing convolutional layers tailored specifically for medical image denoising. He et al. [35] introduced the concept of residual learning, which involves the use of residual connections, into deeper CNNs to improve the performance of denoising tasks. Yuan et al. [36] proposed a denoising approach for hyperspectral images that combines deep CNNs, residual learning, and multiscale knowledge. Gholizadeh et al. [37] incorporated dilated convolutions to expand the receptive field and reduce network depth for CT image denoising. Most of the methods mentioned above rely on improved CNN architectures for medical image denoising. Therefore, the design of network architectures plays a crucial role in achieving effective denoising outcomes.

In the domain of medical image analysis, Neural Architecture Search (NAS) techniques have also been employed. For example, Mortazi et al. [11] optimized individual layers within building blocks and hyperparameters while maintaining a relatively fixed block structure. Rundo et al. [38] combined histogram equalization with a Genetic Algorithm (GA) [39] to improve image histogram frequency for image enhancement in medical imaging systems. Elhoseny et al. [40] proposed a bilateral filter for medical imaging denoising based on Dragonfly (DF) and Modified Firefly (MFF) algorithms utilizing a CNN classifier for the classification task of denoised images as normal or abnormal. Liu et al. [41] proposed a network evolution approach utilizing GA to design CNN network structures for medical image denoising automatically. However, their approach focused solely on residual blocks, potentially limiting the exploration of other recent blocks beyond the residual block paradigm. Hence, there is a growing need to automatically generate efficient denoising models based on CNN, incorporating optimal layer connections, and appropriate hyperparameter combinations while considering a various types of blocks.

2.2 2D Medical image segmentation

Traditional 2D medical image segmentation methods encompass various strategies such as threshold-based, region-based, clustering, and supervised methods. In threshold-based method [5], an intensity threshold is computed based on MRI scans, and each pixel is assigned to different classes based on a comparison between its gray value and the threshold. This approach assumes that pixels within a specific range belong to the same category [42]. However, this method struggles to segment complex images accurately. Selecting an appropriate threshold is crucial as it directly impacts the quality of the segmentation results. Region-based segmentation methods employ algorithms like watershed and region-growing to identify distinct regions [6, 43]. These techniques offer advantages such as simplicity in calculations, high accuracy, and effective extraction of regional features. However, they are susceptible to noise and may introduce holes in the extracted regions [44].

Clustering techniques, including k-means and fuzzy c-means, partition the input image into multiple segments [7, 45]. However, these methods' computational demands and time complexity make them less suitable for large-scale data processing. In supervised approaches such as support vector machines and Random Forest, labeled data is required for accurate segmentation [46, 47]. Although these algorithms tend to outperform other traditional methods, they still have limitations in brain tumor segmentation. For instance, selecting appropriate hyperparameters for a support vector machine can be highly challenging, and Random Forest may not perform optimally with low-dimensional tumor data [44].

Deep learning algorithms, specifically CNNs, have shown exceptional performance in pixel-wise medical image segmentation. One standard architecture in this domain is the U-Net, introduced by Ronneberger et al. in 2015 [21]. The U-Net architecture has revolutionized medical image segmentation by employing an encoder to extract features from input data and a decoder to generate segmentation masks. Notably, U-Net-based networks incorporate skip connections, enabling them to learn highly effective representations [48]. Dual U-Net [49] contains two encoders for extracting spatial and context data with a unique module to merge the data from two encoder pathways. Francia et al. [50] recommended

integrating U-Net and U-Net with residuals, where the U-Net can extract the crude features and then U-Net with residuals can capture additional features using multi-scale fusion. This technique requires more complicated image preprocessing, and feature extraction is performed using two U-Nets, which increases the computational cost. The dense residual network (DRNet) was suggested by Guo et al. [51] by using the U-Net skip connection to link several residual blocks with dense-residual blocks and to correlate various dense-residual blocks for retinal vessel segmentation. But, these blocks are prone to overfitting with short datasets. Li et al. [52] proposed the IterNet model using numerous U-Net structures where each structure has its segmentation outputs and loss function. As a result, the network becomes four times deeper than the traditional U-Net due to the iterative network structure.

Weng et al. [15] introduced the NAS U-Net model for finding block structures in a fixed U-Net-based network for medical image segmentation. Hassanzadeh et al. [53] proposed EvoU-Net, an evolutionary model utilizing GA to evolve deep CNNs for medical image segmentation, discovering both network structures and training parameters. Sun et al. [54] presented EvoCNN, utilizing GA with a novel crossover strategy to generate evolutionary deep CNNs for image classification. Fan et al. [55] developed an autoencoder-based network whose configuration parameters are optimized through evolutionary algorithms to improve the segmentation results. Wang et al. [56] proposed a PSO-based approach for image classification, focusing on optimizing a single network block consisting of dense layers while using fixed hyperparameters for convolutional and pooling layers. However, this approach may not fully capture image classification's complexity and could overlook essential interactions between different network layers. Hassanzadeh et al. [57] introduced an evolutionary genetic algorithm-based model combining residual blocks and dense blocks in a U-Net for medical image segmentation. Wei et al. [58] incorporated an environmental selection mechanism in GA that considers population elitism and diversity to enhance search efficiency and avoid premature convergence, aiming to improve segmentation precision. However, it is important to note that most of the existing models optimize a single block and depend on fixed hyperparameters. Consequently, there has been a growing interest in developing automated and efficient DNN models for medical image segmentation

tasks.

2.3 3D Medical image segmentation and Detection

Designing CNN models for 3D medical image segmentation poses greater complexity compared to 2D segmentation, primarily due to the larger volume of data and the substantial increase in the number of parameters involved. Furthermore, medical image data is often volumetric, and analyzing 3D images necessitates the consideration of the entire volume for segmentation. To tackle this challenge, researchers have suggested various approaches. Initially, Çiçek et al. [59] extended U-Net architecture for 3D medical image data and proposed a 3D U-Net directly dealing with 3D medical data. Then, Milletari et al. [60] proposed a similar architecture, V-Net. It is well known that residual connections can avoid vanishing gradients and accelerate network convergence, and it is thus easy to design deeper network structures that can provide better feature representation. Christ et al. [61] proposed a cascaded fully convolutional network (FCN) for automatically segmenting the liver and lesions in CT images. Chen et al. [62] presented a 3D CNN architecture called VoxResNet, designed with residual learning for volumetric brain segmentation. Shakeri et al. [63] used a 2D CNN architecture to detect tumors from a series of brain slices and applied a 3D CRF algorithm for post-processing to implement volumetric homogeneity. Yu et al. [64] utilized mixed residual connections to build volumetric ConvNets for 3D prostate segmentation. Lee et al. [65] proposed 3DRUNet, a neural network model that achieves superhuman accuracy in segmenting neurons in electron microscopy images in the SNEMI3D Connectomics Challenge.

Furthermore, Li et al. [66] introduced H-DenseUNet, a hybrid densely connected UNet architecture that integrates 2D and 3D functionalities for accurate liver and tumor segmentation in CT volumes. Oktay et al. [67] proposed an Attention U-Net model by integrating an attention mechanism into the U-Net architecture to enhance the model's ability to focus on relevant image regions for medical image segmentation. Kolavri et al. [68] proposed an optimized 3D dense-U-Net model, which integrates densely connected layers into the U-Net architecture for spine and brain segmentation. Myronenko et al. [69] introduced

SegResNet, which uses an autoencoder regularization method for detecting brain tumors in 3D images. A 3D deep learning-based method is proposed by Li et al. [70] for the automatic segmentation of the liver from MRI and CT images. Nevertheless, this method requires both MRI and CT images for training, which may not be available in some clinical settings. Liu et al. [71] proposes an ensemble of 3D deep neural networks for the automatic segmentation of the liver in CT images. Hatamizadeh et al. [72] introduced the integration of swin transformers, a self-attention mechanism, into a U-Net architecture for brain tumor prediction in 3D MRI images. Demoustier et al. [73] proposed residual 3D U-Net by incorporating residual connections and localization techniques with Localization (R3D-UNet-Loc) for brain tumor detection. However, due to their large parameter size, these 3D networks suffer from high computational costs and GPU memory usage.

Regarding NAS approaches, there are two categories: algorithms that search for 2D architectures and produce the final segmentation by processing each slice independently [63, 74, 75, 76] and algorithms that search for 3D segmentation architectures [77, 78, 79, 80]. Nevertheless, converting 2D image segmentation models to 3D often results in subpar segmentation performance. These works focused on either optimizing the best number of cells and their connection while keeping the configuration of the cell fixed or the best configuration for the cell while arbitrarily setting the number of cells and their connections. However, these methods mainly focus on optimizing accuracy without considering additional objective functions or constraints imposed by the application.

Furthermore, Isensee et al. [81] presented a self-adapting framework that utilizes a rule-based approach to determine the training parameters and pre-processing operations for a pool of U-Net architectures, facilitating medical image segmentation. Kim et al. [77] proposed a differentiable NAS method specifically designed for optimizing the architecture of a 3D U-Net for accurate segmentation of 3D medical images. Zhu et al. [79] proposed V-NAS for volumetric medical image segmentation, exploring a hybrid design search space of 2D, 3D, and pseudo-3D convolution operations. Calisto et al. [82] proposed AdaEn-Net, which is an evolutionary model designed for both 2D and 3D networks. These networks trained separately and then combined them in an ensemble. Hassanzadeh et al. [83] developed an evolutionary 2D convolutional neural network transformed into

3D networks for optimal evolutionary 3D deep networks in medical image segmentation. Yu et al. [84] presented a coarse-to-fine NAS that addresses network structure through a two-stage optimization problem. However, the sequential optimization of the two problems makes their approach time-consuming. Kaur et al. [75] proposed a GA-based model for the segmentation of 2D and 3D medical images. However, their approach requires additional search time due to the exploration of both 2D and 3D architectures. Hassanzadeh et al. [74] introduced attention-based CNNs using a GA to tune the CNN hyperparameters for 2D and 3D medical image segmentation. However, their approach does not fully exploit volumetric information, potentially limiting segmentation accuracy. Qian et al. [85] proposed Hasa, a hybrid architecture search method using a NAS-based framework to automatically design an optimal architecture, incorporating a pre-trained backbone model for echinococcosis classification and ovary segmentation in ultrasound images. Chu et al. [86] proposed a differentiable search algorithm for designing a unified network for 3D medical image classification. Nevertheless, their approach's extensive joint search space results in high computational costs.

In [87], a multi-objective evolutionary-based algorithm was introduced to adapt a semi-fixed 2D FCN architecture. However, the segmentation is performed slice-wise and does not fully utilize the available volumetric information. Similarly, in [88], an ensemble of 2D and 3D FCNs was automatically designed using a multi-objective evolutionary-based algorithm, which also requires searching for both 2D and 3D architectures and can be time-consuming. Calisto et al. [89] proposed a random forest surrogate-assisted multi-objective evolutionary algorithm for 3D medical image segmentation optimization, which is computationally expensive and takes significant time to converge. Zhang et al. [90] proposed a multi-objective evolutionary algorithm-based method for 3D medical image segmentation. This method combines with a reinforcement learning-based controller to automatically search for an optimal neural architecture. Hu et al. [91] proposed an automated approach for designing deep learning methods using multi-objective evolutionary optimization for medical image segmentation. Zanaty et al. [92] proposed a multi-objective optimization approach for brain tumor detection in MRI scans using DCNNs. This method optimizes accuracy and robustness through a weighted combination of two loss functions.

Guo et al. [93] proposed a multi-objective evolutionary algorithm for designing a 3D CNN for medical image detection. Zhao et al. [94] presented a multi-objective NAS approach for 3D medical image segmentation that automatically searches for an optimal network architecture achieving high accuracy and efficiency. However, it requires a large amount of computational resources, limiting its practical applicability. To address this, there is a growing demand for the automatic development of deep learning models for 3D medical imaging tasks with minimal resource requirements.

2.4 Summary

This chapter presents a brief literature review of the advancements made in medical image processing and analysis, specifically focusing on medical image denoising, segmentation on both 2D and 3D images, and detection. It begins by discussing the limitations of traditional denoising methods and then presents various CNN architectures and techniques used to enhance performance in these tasks. Then explores the use of NAS techniques for optimizing network structures and training parameters in medical imaging, highlighting different optimization algorithms and techniques for finding optimal architectures.

Moreover, the chapter emphasizes the significance of accurate and efficient network architectures in medical image analysis. It emphasizes the importance of an appropriate search space for successful evolutionary CNN parameter optimization. In this regard, the following chapters (chapters 3, 4, 5, and 6) delve into more detailed solution mechanisms for denoising, 2D and 3D segmentation, and brain tumor detection tasks.

Chapter 3

To design a metaheuristic block-based deep neural network for medical image denoising

In this chapter, a metaheuristic block-based model that utilizes Differential Evolution (DE) algorithm for medical image denoising is proposed.

Chapter Organization: Section 3.1 provides the background knowledge related to the proposed model. Section 3.2 presents the proposed methodology. Section 3.3 discusses the experimental results and analysis. Section 3.5 provides the summary of the work.

3.1 Background

This section provide background knowledge relevant to the proposed model, covering topics such as differential evolution, residual block, dense block, and transfer learning. Differential evolution is a popular optimization algorithm commonly used for global optimization problems. Residual blocks are fundamental building blocks in deep neural networks, particularly in the context of residual neural networks (ResNets). Dense blocks, on the other hand, are building blocks commonly employed in DenseNet architectures. Transfer learning is a technique in machine learning where knowledge learned from one task or domain is applied to a different but related task or domain. Understanding these concepts is essential

for comprehending the proposed model and its components, as they form the basis for the design and optimization strategies employed in the research.

3.1.1 Differential Evolution

In 1997, Storn and Price [20] developed the DE algorithm. DE is a vector-based stochastic search algorithm and is used to solve various combinatorial optimization problems. The advantage of the DE is that it obtains global optimum by using a few control parameters and fast convergence [95]. The DE produces optimal solutions than other metaheuristic algorithms [96, 97, 98], and it can be applied to any difficult optimization problems despite multi-modal, noisy, and multi-dimensional spaces. DE could include domain knowledge for all combinatorial optimization steps to dictate the search methodology. Generally, neural network combinations need to incorporate constraints like domain-specific knowledge for search techniques. Differential evolution has been successfully applied in a wide range of applications, including neuroevolution, digital signal processing, and image processing, such as segmentation [99], classification [100], fusion [101], denoising [102], and registration [103]. Therefore, DE is chosen over other heuristic algorithms for the evolution of neural networks in medical image denoising.

In the DE algorithm, there are four main operations: Initialization, Mutation, Crossover, and Selection. The Initialization step involves randomly initializing a population of N_p individual vectors with K dimensions. Each dimension of an individual vector is assigned a random value between the predefined lower bound $X_{j,low}$ and upper bound $X_{j,high}$, as shown in Equation 3.1.

$$X_{i,j}^{t} = X_{j,low} + r * (X_{j,high} - X_{j,low}), \tag{3.1}$$

where r is a random value generated between [0,1], i is the individual $(0, 1, \ldots N_p - 1)$, j is a dimension $(0, 1, \ldots K - 1)$ in the current population; t is index of generation; For instance, $X_{i,j}^{t}$ is j^{th} dimension of i^{th} individual vector in a t^{th} generation. During the Mutation step, a donor vector V_i is calculated by combining three randomly selected and distinct individuals, X_{a_1}, X_{a_2}, and X_{a_3}, with the current individual X_i. The donor vector

V_i^t at t^{th} generation is determined using Equation 3.2.

$$V_i^t = X_{a_1}^t + F * (X_{a_2}^t - X_{a_3}^t); \text{ where } a_1, a_2, a_3 \in \{0, 1, \ldots N_p - 1\}, \tag{3.2}$$

where the scaling factor constant F is chosen in between $[0, 1]$, a_1, a_2, and a_3 are three individuals which are randomly selected and distinct from the current individual i (i.e., $a_1 \neq a_2 \neq a_3 \neq i$) in a population. In the Crossover step, a trial vector U_i^t is generated by mixing the donor vector V_i^t and the target vector X_i^t. The crossover operation is controlled by the crossover probability C_r, which determines the extent to which the trial vector's parameters come from the donor vector. Equation 3.3 describes the crossover process:

$$U_{i,j}^t = \begin{cases} V_{i,j}^t, & \text{if } R_v \leq C_r \text{ or } j = \bar{j}; \text{ where } \bar{j} \in \{0, 1, \ldots K - 1\} \\ X_{i,j}^t, & \text{otherwise.} \end{cases}, \tag{3.3}$$

where R_v is a real number chosen randomly in between 0 to 1, C_r is the crossover probability $(0 \leq C_r \leq 1)$ controls how much of the trial vector's parameters come from the donor vector. Furthermore, the donor vector parameter with the randomly generated index $\bar{j}$ is always inherited by the trial vector, which is different from the trial vector, with at least one parameter being compared to it.

Finally, in the Selection step, the trial vector U_i^t is compared to the target vector X_i^t based on their fitness scores. If the fitness score of the trial vector is lower than that of the target vector, the trial vector replaces the target vector in the population. Otherwise, the target vector remains unchanged for the next generation. Equation 3.4 represents the selection process, where the fitness function f is used to evaluate the fitness scores of the vectors.

$$X_i^{t+1} = \begin{cases} U_i^t, & \text{if } f(U_i^t) \leq f(X_i^t), \\ X_i^t, & \text{otherwise.} \end{cases}, \tag{3.4}$$

This iterative process continues until a certain stopping condition is met.

3.1.2 Residual Block

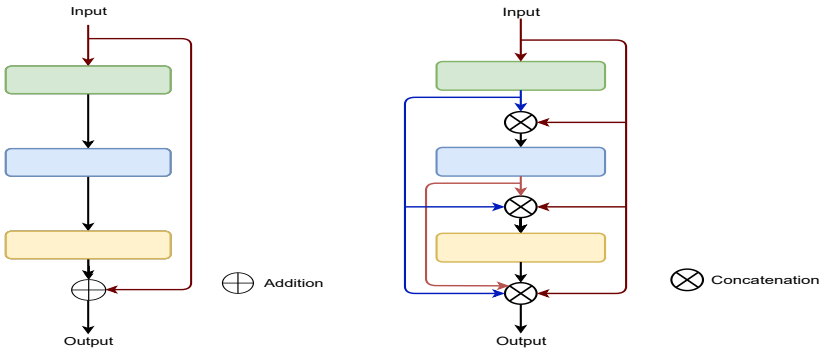
A Residual Neural Network [35] is a type of neural network that is formed by residual blocks, which uses skip or shortcut connections between layers. These connections directly link the input of a block or layer to its output, allowing the network to "skip" over certain layers. By doing so, shortcut connections enable the flow of information from earlier layers to subsequent layers, facilitating the propagation of gradients during training. This helps alleviate the vanishing gradients problem [104] and promotes better gradient flow, allowing to effectively train deeper architectures. Figure 3.1a illustrates the structure of a residual block, in which the input and output feature maps of the same size are added together to produce the block's output. Several recent studies, including Sharif et al. [105], and Jifara et al. [106] used a residual network approach for denoising the medical images. The proposed approach incorporated residual block as one type of building block in a network for medical image denoising.

3.1.3 Dense Block

A Dense Network [107] is a deep neural network constructed by Dense blocks, where each block has multiple layers. To reuse extracted feature maps in this network, each layer's output feature maps are concatenated with the outcomes of all preceding levels in the block, and the output feature maps can be transmitted to the further layers as shown in Figure 3.1b. Stacking many feature maps of varying quality generated by shortcut connections may help the network to reuse features, increase propagation quality, and solve the vanishing gradients problem [104]. Recently, Jia et al. [108] and Park et al. [109] used dense networks for the image denoising process. The proposed approach also integrated a dense block as another type of building block in a network for medical image denoising.

3.1.4 Transfer Learning

The transfer learning technique, as described by Torrey and Shavlik in 2010 [110], refers to the process of leveraging knowledge gained from previous tasks and applying it to enhance learning in other related tasks. This approach enables the utilization of pre-existing trained



(a) Residual Block (b) Dense Block

Figure 3.1: An example structure of Residual and Dense block.

models, hyperparameters, and learned weights from source tasks, which can be transferred and adapted to new tasks. In neural networks, transfer learning optimizes architectures by leveraging knowledge and representations from previous tasks, resulting in more efficient models for target tasks. This approach reduces the need for extensive training on large datasets from scratch, leading to improved performance, reduced training time, and better generalization of new tasks.

3.2 Methodology

In this section, a detailed explanation of the search space is provided, which encompasses the range of possible parameters explored by the DE algorithm. Subsequently, the encoding process is describes the representation of vectors as DNN models. Furthermore, the vector parameter splitting technique and the proposed DEvoNet method are discussed.

3.2.1 Search space and Encoding

Search space: The search space includes the combination of Dense and Residual blocks to design deep neural networks for medical image denoising. When a network contains many residual or dense blocks, it can be challenging to identify the ideal network architecture for the specific task. Manually designing such a complex structure takes a long time and involves a lot of trial-and-error processes [111]. To solve this issue, a block-based model

is proposed for automatically designing a optimal network architecture. The proposed method aims to find the optimal layer pattern in the residual or dense blocks and network training parameters using DE. Table 3.1 lists the search space with 11 parameters and their respective ranges for designing and training a network in the proposed work.

Table 3.1: Hyperparameters and their range to build a network.

Hyperparameter	**Hyperparameter Range**
Type of block	[Residual (0), Dense (1)]
Number of blocks	[2, 3, 4, 5, 6, 7]
Number of conv layers	[1, 2, 3, 4]
Filter size	[3×3, 5×5, 7×7]
Number of filters	[8, 16, 32, 64, 128]
Activation	[ReLU, Tanh, Leaky ReLU]
Optimizer	[Adam, Adadelta, Adamax, SGD, Adagrad]
Batch normalization	[False (0), True (1)]
Learning rate	[0.1, 0.01, 0.001]
Loss function	[Mean Absolute Error, Mean Squared Error]
Batch size	[8, 16, 32, 64, 96]

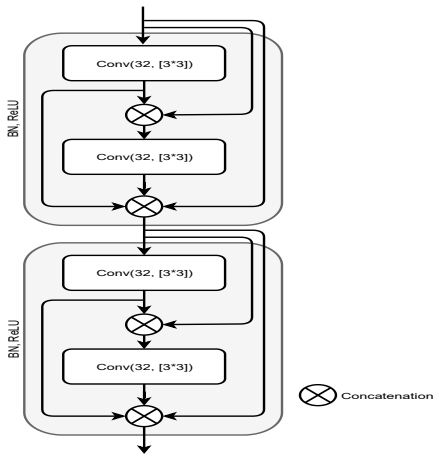
Encoding: In a population, each vector corresponds to a network during the evolution process. The maximum number of blocks is seven, and one to three convolution layers can be present in each block. A block-based approach is used to configure the type of block, activation function, number of filters, and batch normalization. However, each of these parameters can affect the whole block. For example, when Batch Normalization (BN) [112] is enabled in a block, it implies that the BN will be applied after every two convolution layers. A dense block vectortype example shown in Figure 3.2a with its corresponding phenotype shown in Figure 3.2b and residual block vectortype (see Figure 3.2c) with the corresponding phenotype (see Figure 3.2d) is shown in Figure 3.2.

As shown in Figure 3.2a, 3.2c, the parameters show the type of block (TOB), whether it is dense or residual, the number of dense/residual blocks (NOB) present in a network, convolution layers per block (NOC), filter size (FS), number of filters (NOF), activation function (ACT), and batch normalization (BN) in the convolution layer respectively. In addition, the remaining respective four parameters, such as learning rate (LR), optimizer

(OPT), loss function (LF), and batch size (BS), are used to train the network.

TOB	NOB	NOC	FS	NOF	ACT	OPT	BN	LR	LF	BS
1 (Dense)	0 (2)	1 (2)	0 (3*3)	2 (32)	0 (ReLu)	0 (Adam)	1 (True)	2 (0.001)	1 (MSE)	1 (16)

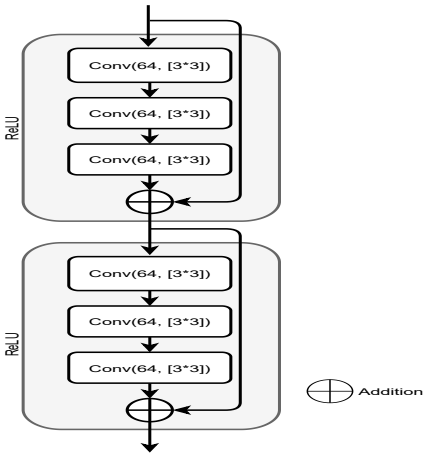
(a) Dense Block Vectortype



(b) Dense Block Phenotype

TOB	NOB	NOC	FS	NOF	ACT	OPT	BN	LR	LF	BS
0 (Residual)	0 (2)	2 (3)	0 (3*3)	3 (64)	0 (ReLu)	0 (Adam)	0 (False)	1 (0.01)	0 (MAE)	2 (32)

(c) Res Block Vectortype



(d) Res Block Phenotype

Figure 3.2: Example Vectortype with its Phenotype of Dense and Residual block.

3.2.2 Vector parameter splitting

In the proposed work, the network's construction parameters, such as type of block, number of blocks, number of conv layers, filter size, number of filters, batch normalization, activation function and training parameters, including the loss function, optimizer, learning rate, batch size are represented by vector parameters. The universal-vector-parameter set (θ) encompasses all hyperparameters, which is further divided into a fine-vector-parameter set (θ_f) and a complementary-vector-parameter set (θ_c). The θ_f includes hyperparameters

selected from existing CNN models like DnCNN [113], while θ_c consists of the universal-vector-parameter set excluding θ_f. The initial population in the DE is initialized based on θ_f, and the DE operations are constructed based on θ_c. Combining this approach with the DE allows for the early identification of optimal CNNs in the search process. The fittest vector parameters take precedence over preserving network structures, ensuring the transmission of potential vector parameters to the population and the examination of the fittest network structures in the early generations. The vector parameters from the top-performing evolved CNNs in the previous evolutionary environment are used to generate θ_f. This method accelerates the identification of optimal network structures, eliminating the need to search the entire search space. As a result, the proposed method captures and transmits this knowledge to subsequent generations.

3.2.3 DEvoNet

The flowchart of the proposed DEvoNet is given in Figure 3.3. The initial population is initialized based on θ_f. Each individual in the population represents a solution for NAS (i.e., DNN architecture), and each individual can be encoded as a DNN architecture, as shown in Figure 3.2. The DNN architectures are then trained and evaluated on the dataset, returning fitness values (Peak Signal-to-Noise Ratio (PSNR) (Equation (3.5))). The PSNR [114] is a metric used in medical image denoising to assess the quality of the denoised image. PSNR provide a quantitative measure of the similarity between the denoised image and the original image. A higher PSNR value indicates a higher level of similarity, implying that the denoised image preserves more details and closely resembles the original image. Following the fitness evaluation of the initial population, the DE operations are performed on the population. Mutation generates donor vectors (Equation (3.2)), then crossover generates trial vectors (Equation (3.3)). The trial vectors are converted to their corresponding DNN architectures, and the networks are trained for a fixed number of epochs. The networks return their PSNR values. The DE selection operation is then conducted, comparing the fitness values of the trial and target vectors (Equation (3.4)). The vector with the highest fitness value is selected and remains in the population. The networks with the highest

PSNR values are carried forward to future generations. Additionally, the maximum fitness vector parameters are added to θ_f, and the complementary-vector-parameter set is updated by $\theta_c = \theta - \theta_f$. The DE operations are performed using θ_c. The DEvoNet process continues until the termination criterion is reached, which corresponds to a specified number of generations T. Finally, the top five individuals with the highest fitness values in the population are selected as the output. It is important to note that the proposed model is a single objective method, where each vector is individually examined based on its fitness value. The objective of the proposed model is to discover networks with the maximum PSNR value using the DE algorithm.

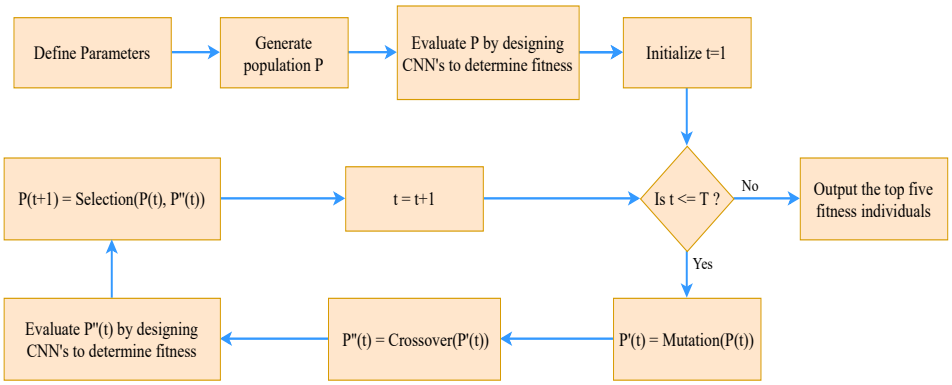


Figure 3.3: Flowchart of the proposed method.

Due to the large search space, DE requires significant computational resources, making it challenging to evaluate performance on large datasets. Therefore, a transfer learning strategy is adopted in this work to transfer the discovered parameters across different sizes or methods in the training data. To facilitate the transfer of network population structures among various-sized datasets, the hyperparameter sets are dynamically updated. The objective is to train a CNN on a small subset D_s to explore optimal combinations of hyperparameters. For this purpose, a total of 250 random images are chosen from the training data. Notably, the entire image, rather than patches or slices, is considered during the evolution. The hyperparameters observed from D_s are then transferred to the larger training set D_l, where the transferred hyperparameters are utilized for further evolution. The effective construction of the neural network is achieved using the fine-vector-parameters θ_f, which have been learned from the small dataset D_s. The utilization of transfer learning accelerates the

evolution rate of the optimization process.

3.3 Experiments

This section provides a description of the datasets, noise levels, and evaluation metrics employed to assess the proposed model. The qualitative and quantitative results of the proposed model are then compared with the existing models.

3.3.1 Datasets

In the experiments, four publicly available medical image datasets, namely CT [115], CheXNet [116], MRI [117], and BraTs2019 [118], were chosen to evaluate the model. The CT dataset comprises 10,000 gray-scale cerebral perfusion chest CT images of size 512×512. This CT image data is collected by the China National Center for Bioinformation (CNCB) as part of the China Consortium of Chest CT Image Investigation (CC-CCII) from various hospital groups across China [115]. The CheXNet dataset consists of chest X-ray images with a size of 384×384, containing information on suspected and positive COVID-19 patients [116]. The MRI dataset comprises 7,022 brain tumor images of size 256×256 [117]. Lastly, the BraTs2019 dataset contains brain tumor MRI images that were cropped to a size of 384×384 [118]. Random image selection is performed to create a diverse training dataset to address overfitting issues. The small training dataset, denoted as D_s, is composed of 250 randomly selected images from each dataset. For each dataset, the training, testing, and validation sets were split into percentages of 60, 25, and 15, respectively, ensuring no overlap between the sets.

3.3.2 Noise Simulation

Unlike natural images, medical images contain signal-dependent noise, making it difficult to eliminate these noises using the conventional natural image denoising techniques currently available in the literature. The present noise is the result of numerous noise sources, including capturing sensors, device processing, and data transmission media, such a noise

creation process is too intricate to handle. Gaussian noise is a statistical concept related to the distribution of noise values. It describes a random process in which noise values follow gaussian distribution. Most of the works [119, 120] assumed the noise present in a medical image as mixed Gaussian–Poisson noise. So, in the experiments, noise standard deviation σ is varied at values of 15, 24, 36, and 52 to investigate the performance of the proposed method.

3.3.3 Experimental Setup

Based on the promising results presented in the literature [121, 113], feasible vector parameters were selected from the CNN hyperparameters. A constrained scenario is adopted, where θ consisted of 11 sub-vector types, as detailed in Table 3.1 in section 3.1. The initial fine-vector-parameter set θ_f is initialized with the following values: type of block = [0, 1], number of convolutional layers = [2, 3], number of blocks = [3, 4, 5], number of filters = [16, 32, 64], filter size = [3×3, 5×5], activation = [ReLU, Tanh], batch normalization = [False, True], optimizer = [Adagrad, Adam, SGD], learning rate = [0.1, 0.01], loss function = [Mean Squared Error, Mean Absolute Error], and batch size = [16, 32]. The model is implemented using the TensorFlow Keras [122], and computations were performed on a system equipped with a 2.2GHz Intel Xeon®CPU, 16 GB of RAM, and a Quadro P5000 graphics card.

Four datasets were considered during experiments, and the model is trained separately on each dataset. Based on preliminary experiments, the population size N_p is initialized as 40, and the total number of generations T_g is set to 10. Additionally, a pre-termination condition (Tc) is defined to terminate the algorithm when the top five vectors remain unchanged for consecutive generations. During the initialization stage, 40 networks were initialized as potential solutions derived from state-of-the-art methods. Relatively large population size is chosen to increase diversity and facilitate the discovery of optimal networks. From the second generation onwards, the population size is halved, selecting the top 20 vectors, and training continues for nine generations. To save time, networks were trained for up to 8 epochs on small datasets D_s and 12 epochs on large datasets D_l during

the training process. The crossover rate (C_r) and scaling factor (F) were set to 0.7 and 0.5, respectively, based on experiments conducted during the DE process. Zero-padding is performed to maintain consistent feature map sizes during each convolution. The evolved networks were evaluated using the Peak Signal-to-Noise Ratio (PSNR) (Equation (3.5)), which served as the fitness function during the evolution process and is employed to analyze networks.

$$PSNR = 10 * \log_{10} * \left(\frac{R^2}{MSE} \right), \tag{3.5}$$

where MSE is the mean squared error between denoised and original images and R indicates maximum pixel value. Additionally, structural similarity index (SSIM) [123] (Equation (3.6)) is also employed to compare the proposed model with existing models,

$$SSIM = \frac{(2 * \mu_Y * \mu_{\bar{Y}} + k_1)(2 * \sigma_{Y\bar{Y}} + k_2)}{(\mu_Y^2 + \mu_{\bar{Y}}^2 + k_1)(\sigma_Y^2 + \sigma_{\bar{Y}}^2 + k_2)}, \tag{3.6}$$

where μ_Y, $\mu_{\bar{Y}}$ are the means of the original and test image, k_1 and k_2 both are constants, σ_Y^2 and $\sigma_{\bar{Y}}^2$ denotes the variances, $\sigma_{Y\bar{Y}}$ represent the covariance between Y and $\bar{Y}$.

3.3.4 Experimental Results

As mentioned previously, the proposed evolutionary procedure is iterated for 10 generations for each dataset, and the final denoised networks are selected based on the acquired results. The top five networks were chosen for further analysis. In contrast, the finest networks underwent training for 12 epochs, but additional retraining is required. The weight initialization of the network played a role in the results since it is randomized. Consequently, each network underwent training for 35 epochs to determine the optimal architecture. Subsequently, the optimal networks were selected for each dataset. The PSNR values of the top 5 networks trained for 12 and 35 epochs for each dataset are presented in Table 3.2. The proposed model identified denoised networks with PSNR values of 35.50 for CheXNet, 35.42 for the MRI dataset, 34.89 for the CT dataset, and 36.85 for the BraTs2019 dataset when σ=24. These results highlight the capability of the proposed framework to discover networks with high PSNR values, exhibiting diverse features across different datasets.

Table 3.2: The top five networks PSNR values when trained on 12 and 35 epochs for each dataset.

Network	CT		CheXNet		MRI		BraTs2019	
	12	35	12	35	12	35	12	35
Network1	33.95	34.62	35.02	35.37	35.02	35.29	36.41	36.85
Network2	34.01	34.84	34.94	35.50	34.92	35.24	36.30	36.56
Network3	33.76	34.39	34.92	35.27	34.81	35.42	36.33	36.81
Network4	34.20	34.89	34.90	35.36	34.75	35.30	36.29	36.71
Network5	33.96	34.72	34.99	35.42	34.62	35.41	36.26	36.83

The proposed evolutionary model discovers the unique structures of the network for each dataset along with its training parameters. For each dataset, the top five network structure vectortypes are shown in Table 3.3. For instance, the first network structure vectortype of the CheXNet dataset indicates that the network is developed by six residual blocks where two convolution layers in each block with relu as activation, 3×3 filter size, 64 filters, the loss function is the mean absolute error, Adam optimizer with learning rate 0.01, and batch size is 16. Moreover, every network has a different set of training parameters.

Table 3.3: The top five network structure vectortype with respect to dataset.

Dataset	Network	TOB	NOB	NOC	FS	NOF	ACT	OPT	BN	LR	LF	BS
CT	Network1	1	1	1	0	3	0	1	1	2	0	1
	Network2	1	2	1	0	3	0	0	1	2	1	0
	Network3	1	1	1	0	3	0	0	0	2	0	2
	Network4	0	3	0	0	3	0	0	0	2	0	0
	Network5	1	3	1	0	3	1	2	1	2	1	1
CheXNet	Network1	0	4	1	0	3	0	1	0	1	0	2
	Network2	1	3	1	0	2	1	0	0	2	1	0
	Network3	0	3	1	0	3	0	0	1	2	0	1
	Network4	1	4	2	0	3	0	0	0	2	1	0
	Network5	0	3	0	0	3	1	0	1	2	0	2
MRI	Network1	1	3	1	0	3	0	0	0	2	0	0
	Network2	0	4	1	0	3	0	0	1	2	0	0
	Network3	0	3	0	0	3	0	1	0	2	0	2
	Network4	0	3	1	0	3	1	2	0	2	1	1
	Network5	1	2	1	1	2	0	0	0	1	1	1
BraTs2019	Network1	0	3	0	0	3	0	1	0	2	0	2
	Network2	0	4	1	1	3	0	2	1	1	1	1
	Network3	1	3	0	0	3	0	0	0	2	0	0
	Network4	0	4	1	0	3	1	1	1	2	0	1
	Network5	0	3	2	0	3	1	0	0	2	1	2

A comparison of both qualitative and quantitative results of a proposed model named

DEvoNet with existing methods, including DnCNN [113], FFDNet [124], CBDNet [125], RIDNet [126], EvoNet [41]. The quantitative results of PSNR and SSIM are summarised in Table 3.4. Based on the obtained results, the DEvoNet model performs significantly better than existing denoising models, where the proposed model achieves higher SSIM and PSNR values at different noise levels. Accurate and reliable visual information is of utmost importance in medical imaging for effective diagnosis and analysis. Therefore, evaluating the quality of denoised images becomes crucial in this context. To assess the performance of the proposed model, visual comparisons are made with existing models on four datasets: CheXNet, CT, MRI, and BraTs2019. The denoised results are showcased in Figures 3.4, 3.5, 3.6, and 3.7, respectively, under a noise level of $\sigma = 24$. These figures provide a visual representation of the improvement achieved by the proposed model in comparison to existing models on different medical imaging datasets. In these figures, the red boxes of the original image are scaled up to below blue box images for better visual comparison. The evolutionary framework's flexibility of the proposed model allows for identifying a better combination of network components that is suitable for the target task.

3.4 Experimental Analysis

In this section, a comprehensive analysis of the results obtained during the evolutionary process is presented.

3.4.1 DE Parameters

The success of evolutionary optimization techniques heavily relies on the probabilities of crossover rate and scaling factor. The determination of optimal values for these parameters involves a trial-and-error process and may vary depending on the specific problem [127]. Different crossover rate and scaling factor values have been defined and utilized in the literature based on the architecture model. Through an extensive survey, several combinations of crossover rate and scaling factor were identified, and experimentation is conducted to evaluate their impact. The results revealed that certain values yielded improved outcomes. The performance is examined at various rates (e.g., 0.3, 0.5, and 0.7) for the scaling fac-

Table 3.4: Average PSNR and SSIM of models: FFDNet, CBDNet, RIDNet, DnCNN, EvoNet, and DEvoNet at different σ = 15, 24, 36, and 52

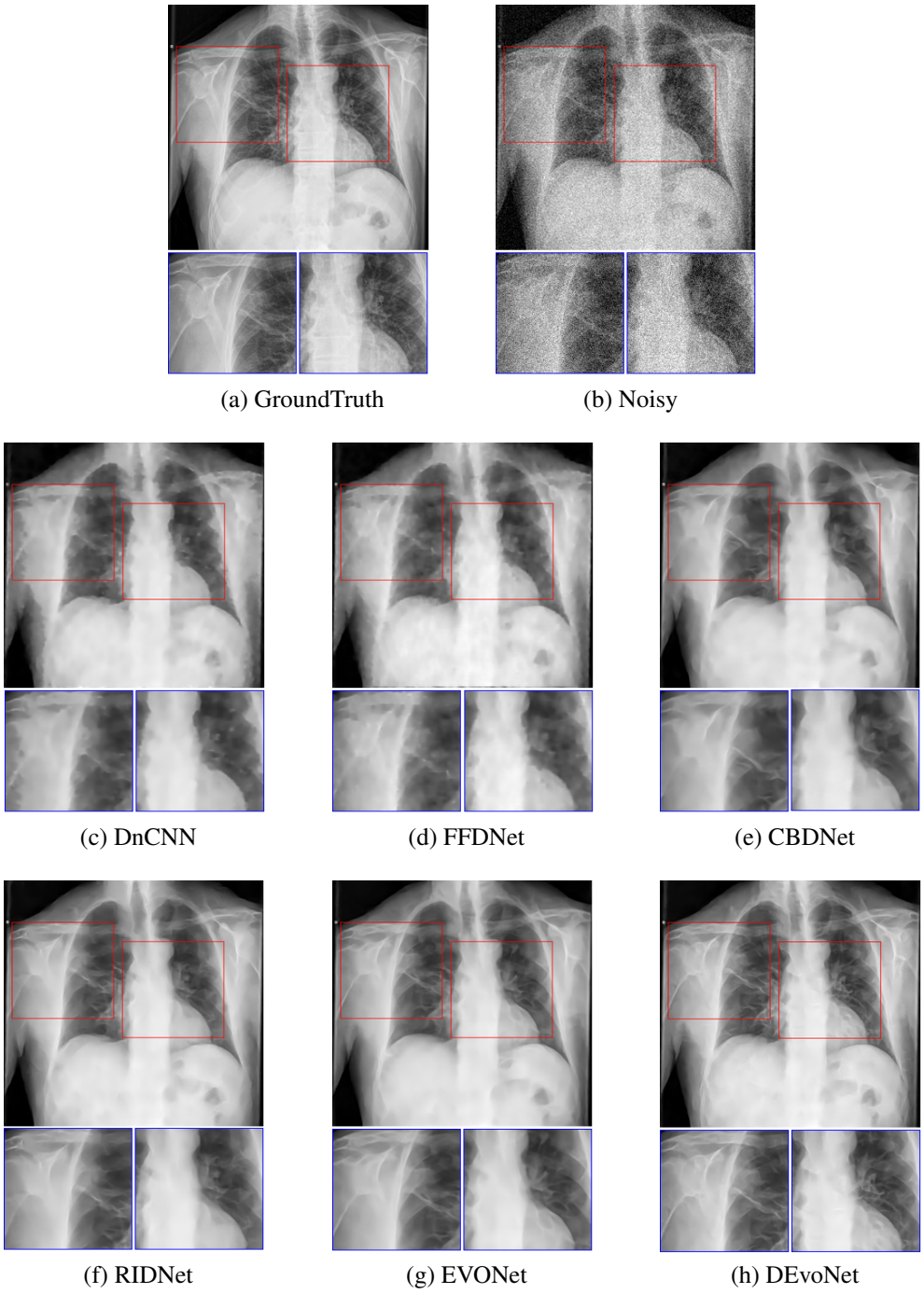
Model	σ	CT		CheXNet		MRI		BraTs2019	
		PSNR (dB)	SSIM	PSNR (dB)	SSIM	PSNR (dB)	SSIM	PSNR (dB)	SSIM
DnCNN [113]	15	34.42	0.90	35.32	0.90	35.71	0.92	36.26	0.92
	24	33.73	0.88	33.27	0.89	33.16	0.91	34.37	0.90
	36	31.56	0.86	30.21	0.87	32.93	0.88	31.41	0.88
	52	30.01	0.84	28.87	0.82	30.65	0.84	30.93	0.86
FFDNet [124]	15	34.45	0.90	35.99	0.89	35.87	0.92	36.11	0.93
	24	33.42	0.88	33.54	0.88	33.54	0.91	34.11	0.92
	36	32.04	0.87	30.18	0.86	32.64	0.88	33.17	0.89
	52	31.11	**0.86**	29.97	0.84	30.92	0.85	32.13	0.87
CBDNet [125]	15	35.81	**0.91**	37.16	0.92	37.73	0.93	37.14	**0.95**
	24	34.88	0.89	35.22	0.90	35.38	0.92	36.76	**0.94**
	36	33.46	0.87	33.03	0.86	33.49	**0.90**	34.16	0.90
	52	31.74	**0.86**	31.90	0.84	31.69	0.86	32.52	0.88
RIDNet [126]	15	35.94	**0.91**	37.26	**0.93**	37.71	**0.94**	37.13	**0.95**
	24	34.04	0.89	35.41	0.90	35.36	**0.92**	36.60	**0.94**
	36	33.27	0.87	34.15	**0.89**	33.48	0.89	34.32	0.90
	52	32.04	**0.86**	32.61	**0.87**	31.75	0.86	32.49	0.88
EvoNet [41]	15	35.60	**0.91**	37.25	**0.93**	37.64	**0.94**	37.09	**0.95**
	24	34.11	0.89	35.48	**0.91**	34.91	0.90	36.82	**0.94**
	36	33.16	0.87	34.09	**0.89**	33.34	0.89	34.87	**0.91**
	52	32.01	**0.86**	32.64	**0.87**	31.77	**0.87**	32.62	**0.89**
DEvoNET (Ours)	15	**35.97**	**0.91**	**37.27**	**0.93**	**37.78**	**0.94**	**37.18**	**0.95**
	24	**34.89**	**0.90**	**35.50**	**0.91**	**35.42**	**0.92**	**36.85**	**0.94**
	36	**33.49**	**0.88**	**34.19**	**0.89**	**33.60**	**0.90**	**34.89**	**0.91**
	52	**32.11**	**0.86**	**32.74**	**0.87**	**31.79**	**0.87**	**32.71**	**0.89**

Note: The best values for each dataset are highlighted in bold.

tor (Figure 3.8a) and crossover rates (Figure 3.8b). High scaling factors resulted in faster search speeds within the search area, but optimal networks may not be found. Conversely, excessively low scaling factors led to network structures quickly converging to local optima instead of the global optimum. Based on these experiments, the values of 0.5 for the scaling factor (F) and 0.7 for the crossover rate (C_r) were selected.

3.4.2 Network Structure Parameters

The parameters utilized to construct an optimal network architecture are examined. The type of block (such as residual block and dense block), number of blocks, convolution layers in each block, activation function, and batch normalization can contribute to network construction. For instance, as per results shown in Table 3.3, the top five networks produced for the CT dataset have more dense block based networks, while the top five networks produced for the BraTs2019 dataset have less dense block based networks. Moreover, the



(a) GroundTruth (b) Noisy

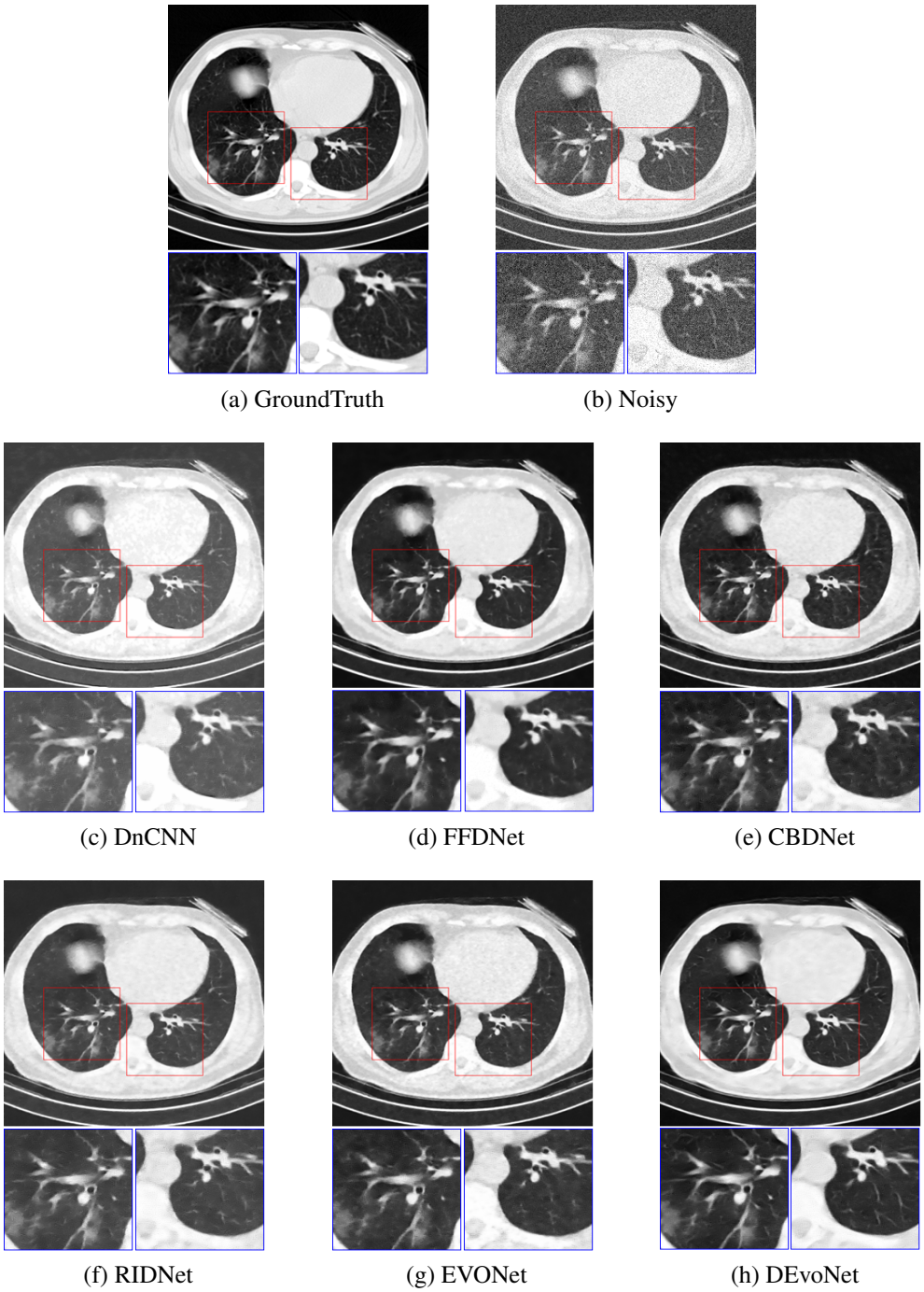
(c) DnCNN (d) FFDNet (e) CBDNet

(f) RIDNet (g) EVONet (h) DEvoNet

Figure 3.4: CheXNet dataset visual results when noise level (σ) = 24.

top five networks used 11 residual and 9 dense blocks.

Convolution has the most important role in constructing networks in all four datasets. In addition, the number of filters, filter size, and the number of convolution layers play a prominent role in feature extraction. The number of filters and filter size for the convolution



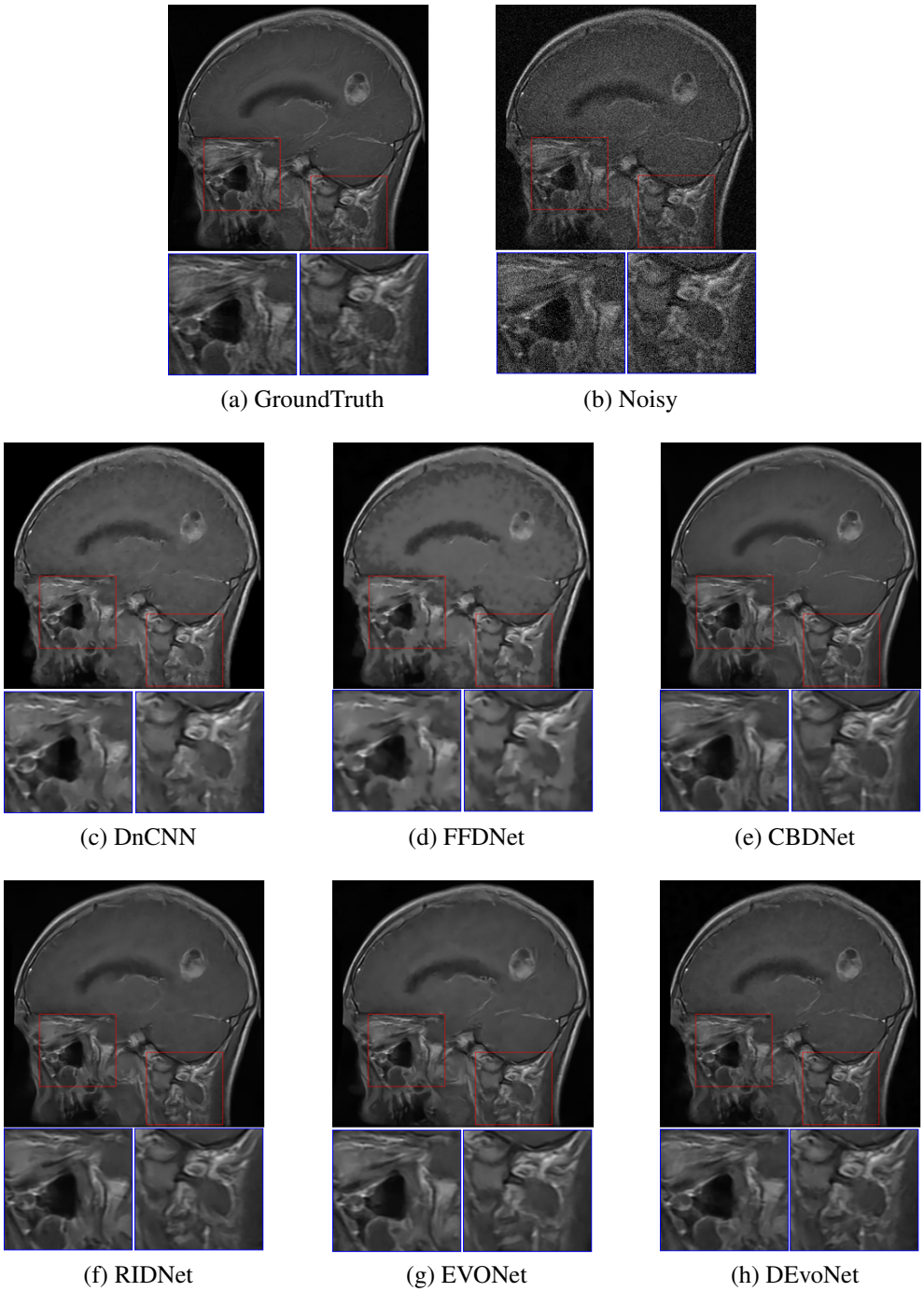
(a) GroundTruth (b) Noisy

(c) DnCNN (d) FFDNet (e) CBDNet

(f) RIDNet (g) EVONet (h) DEvoNet

Figure 3.5: CT dataset visual results when noise level (σ) = 24.

layer were also determined independently in the proposed model. Based on the obtained results, 37 convolution layers were used by the top five networks in that 35 convolution layers used 3×3 filter size and 64 filters, remaining convolution layers utilized 32 filters and 5×5 filter size. This diversity demonstrates the complexity of determining the best



(a) GroundTruth (b) Noisy

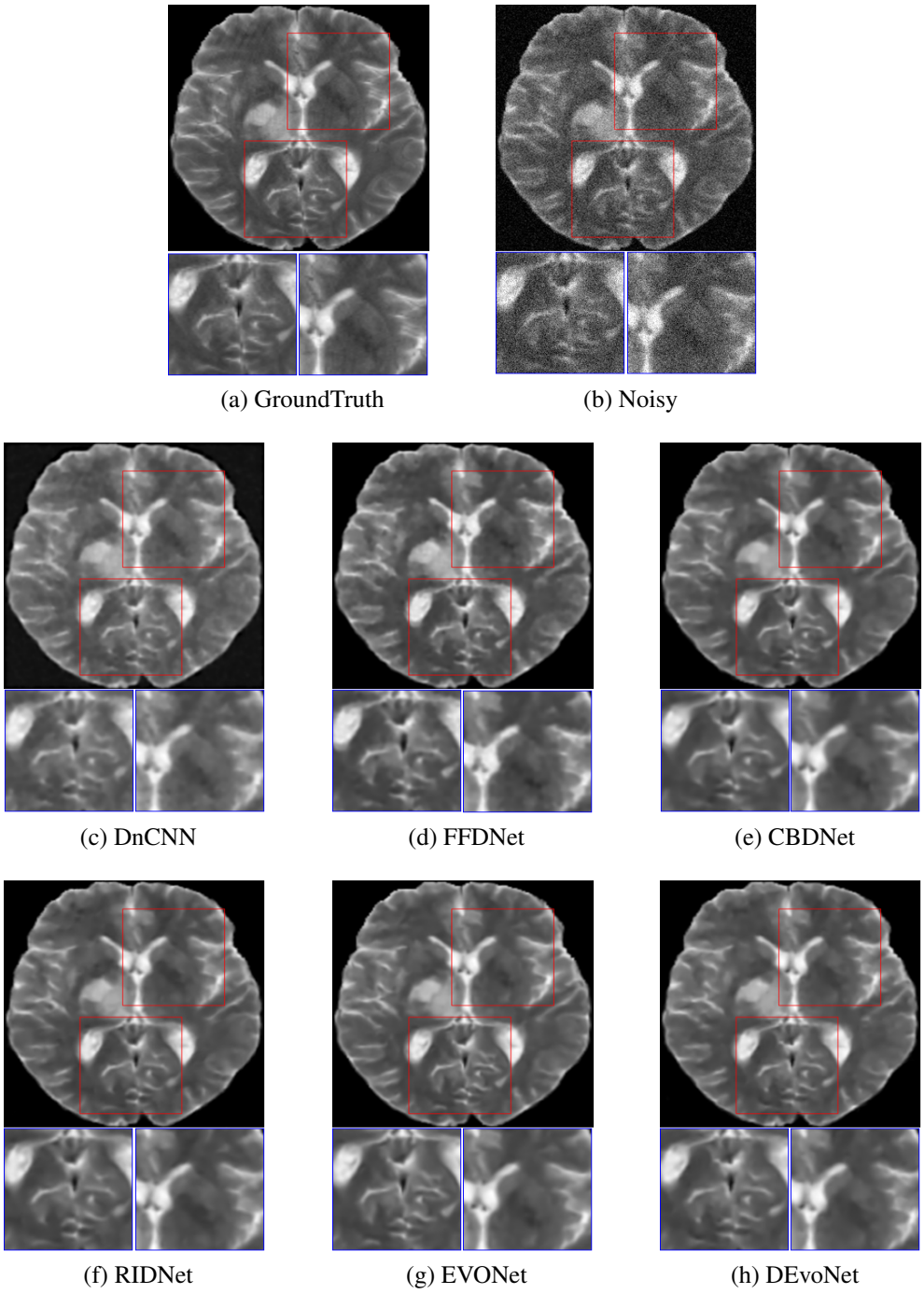
(c) DnCNN (d) FFDNet (e) CBDNet

(f) RIDNet (g) EVONet (h) DEvoNet

Figure 3.6: MRI dataset visual results when noise level (σ) = 24.

block combination to construct a network.

Furthermore, the output of each node concerning its input can be specified using activation functions, which is essential for neural network convergence and speed. Besides, it



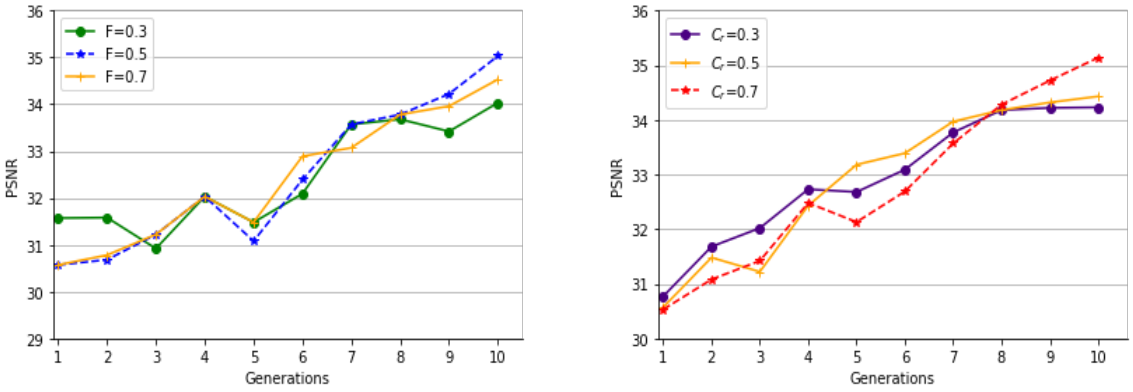
(a) GroundTruth (b) Noisy

(c) DnCNN (d) FFDNet (e) CBDNet

(f) RIDNet (g) EVONet (h) DEvoNet

Figure 3.7: BraTs2019 dataset visual results when noise level (σ) = 24.

can prevent the neural network from convergence in some cases. Therefore, the convolution layer should use an optimal activation function. In the evolutionary process, each dataset produces a different number and type of activation functions in the evolution process, as



(a) Scaling Factor vs PSNR

(b) Crossover rate vs PSNR

Figure 3.8: Change in the performance with different values of scaling factor and crossover rate.

shown in Figure 3.9. However, the obtained results demonstrate the diversity of network architectures and the relevance of the automatic construction of the networks.

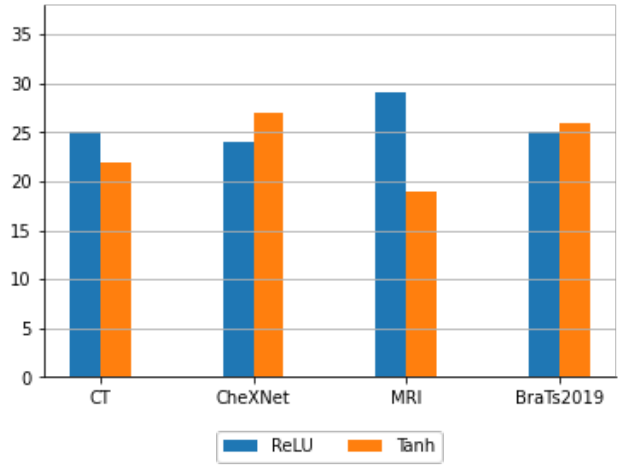
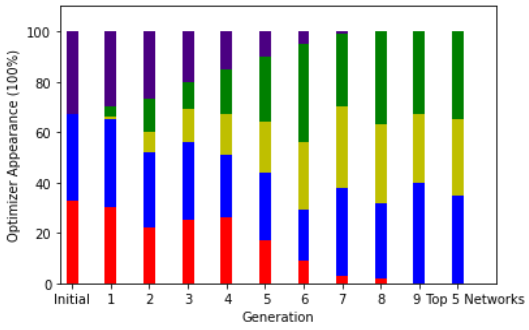


Figure 3.9: The frequency of activation functions in network evolution.

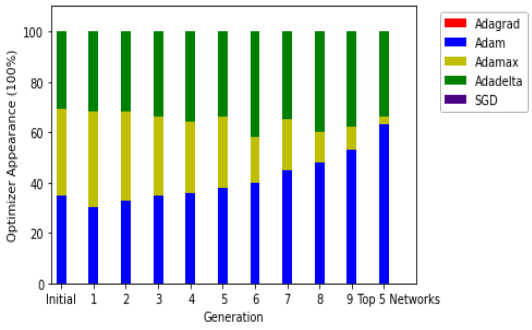
3.4.3 Training Parameters of Network

The optimizer with the corresponding learning rate is chosen automatically during the evolution of the proposed method. Figure 3.10 shows the evolution of optimizers for each generation when training on a small dataset and after transferring them to a larger training dataset. As can be seen, the low-performance vector parameters such as SGD and Adagrad

are gradually eliminated over generations. Simultaneously, high-performance vector parameters such as Adamax and Adadelta are discovered by DE operations. The optimal vector parameters such as Adamax, Adadelta, and Adam are retained over generations when the initialization set is transferred from a small training set Figure 3.10a to a larger training set Figure 3.10b. Throughout evolution, top-performing vector parameters like Adam and Adadelta have dominated the optimizer function vector parameters. It also demonstrates that CNN hyperparameters (vector parameters) and structures learned on a small dataset can be transferred to a large dataset. Furthermore, for training, the majority of networks used a small learning rate (0.001). The loss functions (such as mean absolute error and mean squared error) and batch size are the next two training parameters discovered using the evolutionary method.



(a) The appearance of optimizers on the small datasets D_s



(b) The appearance of optimizers after transferred to large dataset D_l from small datasets D_s

Figure 3.10: The appearance of optimizers in the evolution process.

3.5 Summary

In this chapter, a metaheuristic block-based deep neural network named DEvoNET is designed by integrating a differential evolution algorithm to automatically identify the optimal network structure and training parameters for medical image denoising. The incorporation of the differential evolution algorithm enables DEvoNET to efficiently explore the search space and discover high-performing configurations. A transfer learning approach is employed to further accelerate the evolutionary process, leveraging the knowledge gained

from previous evolved architectures. The experimental results demonstrate the superiority of the proposed DEvoNET model over existing denoising models, emphasizing its potential in automatically constructing networks for various denoising applications. This capability holds promise for enhancing the quality of medical image denoising and ultimately improving patient care by providing clearer and more accurate medical images. In the next chapter, the proposed model will be utilized as a preprocessing step in 2D medical image segmentation.

Chapter 4

To develop a metaheuristic based modified U-shaped network for 2D medical image segmentation with denoising

In this chapter, a metaheuristic based modified U-shaped network is developed for 2D medical image segmentation using the BTLBO algorithm, along with the incorporation of denoising (Chapter 3) as a preprocessing step.

Chapter Organization: The proposed methodology is presented in Section 4.1. The experimental results are provided in Section 4.2. The experimental analysis is discussed in Section 4.3. Lastly, Section 4.4 presents the summary of the work.

4.1 Proposed Method

In this section, the modified U-shaped network, the search space and attention blocks are presented. Subsequently, the proposed BTU-Net method with the caching technique is discussed.

4.1.1 Modified U-shaped network

Generally, a simplified hierarchical architecture, known as a U-shaped (encoder-decoder) network, is employed in the image segmentation process to extract more details and multiscale features compared to other networks. The U-shaped network consists of downsampling (encoder) and upsampling (decoder) parts. Feature extraction is achieved using pooling layers as the network progresses from higher to lower resolutions in the downsampling section. Conversely, deconvolution is employed in the upsampling part to reconstruct the coarse tone segmentation. A specific block, known as the bottleneck (bridge), connects these two parts. The rationale behind selecting a U-shaped network as the backbone of the proposed work is its ability to achieve improved segmentation results, even with limited datasets.

While the basic U-Net utilizes fixed block structures, which may not always be suitable for different segmentation tasks, the proposed model incorporates different block structures that are chosen from a designed search space. In the U-Net structure, employing multiple pooling layers during downsampling can result in the loss of spatial data. During upsampling, addition or concatenation is utilized to recover this information loss. As attention blocks may autonomously learn to focus on target structures without any further supervision, four attention block structures have been employed, as shown in Figure 4.2 based on [67]. These attention blocks are used to recover spatial information and then transfer them during upsampling in place of addition or concatenation operations. The choice of attention block parameters is also included in the search space of the proposed work. Figure 4.1 illustrates the modified U-shaped (encoder-decoder) network structure, which consists of 9 blocks and attention blocks.

4.1.2 Search space and Encoding

In the previous work, as mentioned in 3.2.1, only residual and dense blocks were utilized. However, in this work, the proposed approach has been expanded by incorporating various nodes with different kernel sizes, activations, and normalization layers. The focus is on optimizing the node structure within each block of the encoder-decoder architecture. Addi-

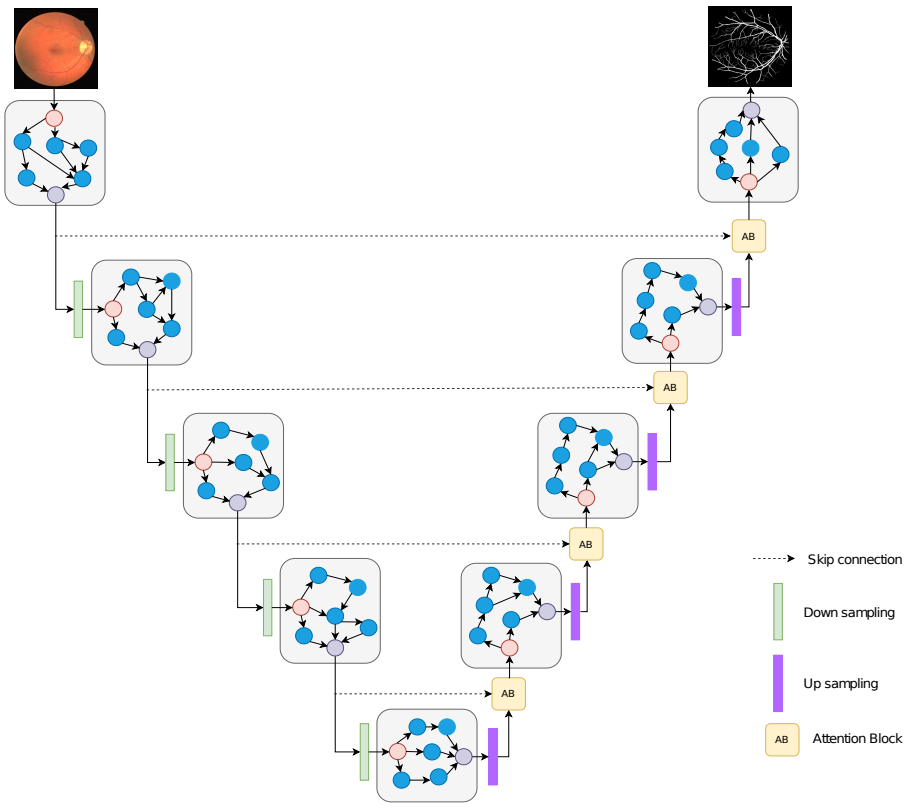


Figure 4.1: The proposed modified U-shaped network.

tionally, to streamline the optimization process, less impactful parameters such as learning rate and batch size are eliminated by fixing their values.

Search space: The search space encompasses various building blocks and parameters that enable the generation of diverse types of DNNs. A larger search space, in theory, encompasses more neural network architectures. However, there is a trade-off between efficiency and search space. A large search space requires more resources and search time. In this work, the search space is restricted to a specific number of operations and these operations are chosen from the state-of-the-art model components [128, 53, 58]. The proposed search space contains 32 node operations as depicted in Table 4.1. Furthermore, these 32 nodes are divided into pre-activation nodes (where the activation function performs before the convolution operation) and post-activation nodes (where the activation function performs after the convolution operation). The node sequence operations in search space were designed with different convolutional kernel sizes (1×1, 3×3, and 5×5), activation functions (ReLu [129] and Mish [130]), normalization functions (batch normalization [112], and instance

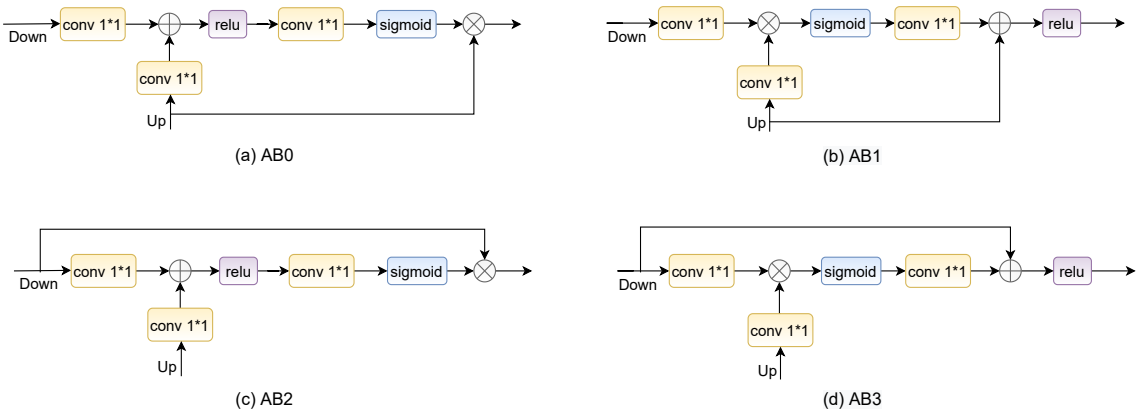


Figure 4.2: Attention blocks [67]

normalization [131]). Additionally, pooling layers for the downsampling (Averagepool and Maxpool), upsampling (bilinear upsampling and transposed convolution [132]), and loss functions (Focal loss [133], DiceBCE loss (a mix of Dice loss and Binary Cross Entropy loss) [134]) are added into the search space.

Table 4.1: The node sequence operations.

Post-activation nodes				Pre-activation nodes			
ID	Conv	Norm	Act	ID	Norm	Act	Conv
00000	1×1	-	Relu	10000	-	Relu	1×1
00001	1×1	-	Mish	10001	-	Mish	1×1
00010	1×1	IN	-	10010	IN	-	1×1
00011	1×1	BN	-	10011	BN	-	1×1
00100	3×3	-	Relu	10100	-	Relu	3×3
00101	3×3	-	Mish	10101	-	Mish	3×3
00110	3×3	IN	ReLu	10110	IN	ReLu	3×3
00111	3×3	IN	Mish	10111	IN	Mish	3×3
01000	3×3	BN	ReLu	11000	BN	ReLu	3×3
01001	3×3	BN	Mish	11001	BN	Mish	3×3
01010	5×5	-	ReLu	11010	-	ReLu	5×5
01011	5×5	-	Mish	11011	-	Mish	5×5
01100	5×5	IN	ReLu	11100	IN	ReLu	5×5
01101	5×5	IN	Mish	11101	IN	Mish	5×5
01110	5×5	BN	ReLu	11110	BN	ReLu	5×5
01111	5×5	BN	Mish	11111	BN	Mish	5×5

Conv: Convolution kernel size, Norm: Normalization, Act: Activation function, BN: Batch Normalization, IN: Instance Normalization.

Encoding: The proposed method consider the internal block structure in a modified U-shaped network as a directed acyclic graph with nodes and edges based on Genetic CNN [135]. The connections between nodes are represented by edges, whereas each node indi-

cates a set of operations, as shown in Table 4.1. The output feature map of the pre-node can be transferred to the post-node through a directed edge between these nodes. The elements of the feature maps will be summed up if a node has multiple edge inputs, as the size of all feature maps inside a block is configured to be the same. If the maximum number of intermediate nodes is P, then $\frac{P(P+1)}{2}$ (i.e., $1+2+3+\cdots+(P+1) = \frac{P(P+1)}{2}$) bits are needed for encoding the inter-node connections. The proposed model considers 5 nodes; this requires 10 bits to build the network structure and 5 bits to choose the node ID. The two examples of binary string encoding between nodes in a block are shown in Figure 4.3. The first bit of binary string represents the link between (node1, node2), the next two bits represent the link between (node1, node3) and (node2, node3), and so on. The two nodes are connected if the corresponding bit is 1. In a block, the input node (red-coloured) receives input from the predecessor pooling layer and transfers it to the successor nodes (blue-coloured). The output node (violet-coloured) takes input from the predecessor nodes (blue-coloured) and forwards it to the next layer in a network.

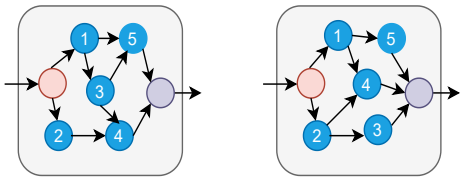


Figure 4.3: Encoding of the block inter-node connections.

4.1.3 BTU-Net

The BTU-Net used the Binary Teaching-Learning-Based Optimization (BTLBO) algorithm in the search process to search for the optimal node structure of each block in the designed U-shaped network. The standard TLBO algorithm, proposed by Rao et al. [22], takes inspiration from the interactions and communications between students and teachers in a classroom to design an effective optimization approach. The BTLBO algorithm is a binary version of TLBO that offers distinct advantages over other algorithms. Unlike many swarm intelligence and evolutionary-based algorithms that require tuning algorithm-specific parameters like mutation and crossover probabilities, BTLBO relies on general control pa-

rameters such as the number of generations and population size. This eliminates the need for extensive parameter tuning, reducing computational effort and avoiding suboptimal solutions trapped in local optima. The effectiveness of this algorithm has been demonstrated in various applications [136], [137], [138]. Therefore, BTLBO has been utilized in this work to find optimal medical image segmentation models during the evolution process. The BTLBO is an efficient metaheuristic algorithm that contains the teacher, learner, and binary conversion phases.

Teacher Phase: This phase imitates the learning process with the teacher's instruction. Let X be a population of the classroom, X_{mean} represent the mean of all learners' results, N_p represents the number of learners in the class, i.e., population size, decision variable K represents the number of subjects, and f indicates fitness function. In this phase, the optimal learner in the classroom is elected as a teacher $X_{teacher}$. The selected teacher tries to improve the class average result based on their ability. The $X^t_{teacher}$ represents the teacher at iteration t, and each learner X_i in the population X updates their values using Equation 4.1.

$$X_i^{t+1} = X_i^t + (rand(0,1) \times (X_{teacher} - (T_f \times X_{mean}))) \tag{4.1}$$

where T_f is a teaching factor, the value can be either 1 or 2 chosen randomly. Each learner X_i in a population X can be encoded as a U-shaped network (as given in section 4.1.2) and runs on a particular dataset then returns a Dice score (Equation 4.5) as a fitness score $f(X_i)$. The new learner X_i^{t+1} is updated to the population X if its fitness score $f(X_i^{t+1})$ is greater than the old learner X_i^t fitness value $f(X_i^t)$.

Learner phase: This phase simulates the learning process with the other learners. Through group discussions and presentations, learners improve their knowledge by interacting with others. A learner gains new information from others who are more knowledgeable than them. Each learner X_i^t interacts randomly with another learner X_j^t where $(i \neq j)$ iteration t to improve their knowledge. A unique partner X_j is chosen for every learner so that every learner can interact with any one of the learner in one generation. The learner X_i^{t+1} updates its value using Equation 4.2.

$$X_i^{t+1} = \begin{cases} X_i^t - r \times (X_i^t - X_j^t), & f(X_i) >= f(X_j) \\ X_i^t + r \times (X_i^t - X_j^t), & \text{otherwise} \end{cases} \tag{4.2}$$

Similar to the teacher phase, the old learner is replaced in the population only if the new learner provides a higher fitness value than the old learner.

Binary Conversion: In BTLBO, the position is represented by a binary vector, while the velocity is still a floating-point vector. When the position of particles is updated, velocity is used to compute the probability of a change from 0 to 1 or 1 to 0. The sigmoid function is a prominent function for normalizing velocities in the [0, 1]. The sigmoid function is used on the velocity component to normalize velocities into the range [0, 1] as follows:

$$X_i^t = \begin{cases} 1, & Sigmoid(X_i^t) \geq rand(0,1) \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

where,

$$Sigmoid(x) = \frac{1}{e^{-x}+1} \tag{4.4}$$

The proposed method utilizes the Dice score (Equation 4.5) as a fitness function in the BTLBO algorithm to optimize the network structures for medical image segmentation. The Dice score [139] is a widely used measure in medical image segmentation, which quantifies the similarity between the segmented region and the ground truth region. It calculates the overlap between the regions, providing a quantitative assessment of the segmentation quality. A higher Dice score indicates a greater degree of spatial correspondence, signifying a more accurate and precise segmentation. To optimize the network structures, the BTLBO algorithm aims to maximize the Dice score. By iteratively evolving the network parameters, the algorithm seeks to find the optimal configuration that maximizes the segmentation performance. The proposed method's pseudocode is outlined in Algorithm 4.1, which describes the steps involved in the BTLBO algorithm for optimizing the network structures. The fitness evaluation, defined in Algorithm 4.2, is responsible for calculating the Dice score as the fitness value for each network configuration during the optimization process.

Algorithm 4.1 The pseudocode of the proposed BTU-Net.

Input: Initial population X with Fitness scores f, Number of generations T_g, Pre-termination criteria Tc

Output: Evolved population X with Fitness scores f

global cache Gc = { }
for $t = 0$ to T_g **do**
  for each learner i in population X **do**
    // Teacher Phase
    $X_{teacher} = max(X, f)$
    $X_{new} = X_i + (rand(0, 1) \times (X_{teacher} - (T_f \times X_{mean})))$
    Convert learner X_{new} to binary learner X_{bnew} using Equations 4.3, 4.4
    Calculate fitness $f(X_{bnew})$ of X_{bnew} by Algorithm 4.2
    Greedy selection between X_{bnew} and X_i
    // Learner Phase
    Choose j randomly where $j \neq i$
    if $f(X_i) \geq f(X_j)$ **then**
      $X_{new} = X_i - r \times (X_j - X_i)$
    else
      $X_{new} = X_i + r \times (X_i - X_j)$
    end if
    Convert learner X_{new} to binary learner X_{bnew} using Equations 4.3, 4.4
    Calculate fitness $f(X_{bnew})$ of X_{bnew} by Algorithm 4.2
    Greedy selection between new X_{bnew} and X_i
  end for
  if Tc is **True then**
    break
  end if
end for
return X, f

In the search process, it is possible for metaheuristic algorithms to generate duplicate learners, resulting in the same network architecture. So, a cache mechanism is utilized in fitness evaluation to avoid retraining the same model (learner) that has already been evaluated or trained. The empty global cache Gc is initialized as described in Algorithm 4.1. The global cache Gc can contain each learner X_i with its fitness value $f(X_i)$. As given in Algorithm 4.2, the presence of learner X_i in Gc is checked before encoding the model [line 1]. If the learner is present, the fitness value $f(X_i)$ is obtained from Gc [line 2]. Otherwise, the model is encoded from learner X_i according to the process described in section 4.1.2, followed by training and evaluation of the model [from lines 3 to 17]. The encoded model is trained for a certain number of training epochs M on the training dataset (considered as 130epochs), and then the Dice score is calculated on the test dataset starting from the test epoch N, assumed to be the 80^{th} epoch. The fitness score for the learner is determined based on the epoch that achieves the highest score [from lines 8 to 13]. Finally, the learner X_i and its fitness score $f(X_i)$ are added to Gc to prevent the model from being retrained [line 16].

Algorithm 4.2 The pseudocode of the Fitness evaluation of a Learner

Input: Learner X_i, Global cache Gc, Train D_{train}, Test D_{test} datasets, Training epochs M, and Test epochs N

Output: Fitness score $f(X_i)$ of a Learner X_i

1: **if** X_i in Gc **then**
2: $f(X_i)$ = query the fitness score $f(X_i)$ of learner from Gc
3: **else**
4: Construct a U-shaped network model based on the information encoding learner as given in section 4.1.2
5: Set f_{best} = 0
6: **for** $epoch = 1$ to M **do**
7: Train the model on D_{train} dataset
8: **if** $epoch > N$ **then**
9: $Dice$ = Calculate the Dice score 4.5 (fitness score) on D_{test} dataset
10: **if** $f_{best} < Dice$ **then**

11: $f_{best} = Dice$
12: **end if**
13: **end if**
14: **end for**
15: Set $f(X_i) = f_{best}$
16: Add the learner X_i and fitness score $f(X_i)$ into the global Cache Gc
17: **end if**
18: **return** $f(X_i)$

4.2 Experimental Results

This section discussed the datasets and metrics used in the evaluation process. Later the experimental results of the proposed models with the state-of-the-art segmentation models were presented.

4.2.1 Datasets & Preprocessing

The proposed method is evaluated on five retinal vessel segmentation datasets and a spleen segmentation dataset. The retinal vessel datasets, including DRIVE [140], STARE [141], CHASEDB [142], IOSTAR [143], and HRF [144] are commonly available in the 2D format as retinal images are typically captured and represented as 2D images. The spleen segmentation dataset [145] contains 1051 2D slice images of CT scans with a size of 512×512. The DRIVE (Digital Retinal Pictures for Vessel Extraction) dataset comprises 40 coloured fundus images with dimensions of 565×584 pixels. The STARE (Structured Analysis of the Retina) dataset includes 20 fundus images with a resolution of 700×605. The IOSTAR dataset comprises 30 images of size 1024×1024. The HRF (High-Resolution Fundus) dataset contains three sets (glaucomatous, diabetic retinopathy and healthy patients), each with 15 fundus images at a resolution of 3504×2336. Lastly, the CHASEDB dataset consists of 28 images with a size of 999×960 from both the left and right eyes.

The datasets mentioned above were split into a training set D_{train} and a test set D_{test}. The training set is used to adjust the network weights, and the test set is used for per-

formance evaluation. The Spleen dataset is divided into 822 images for training and 229 images for testing. The standard partitioning strategy of 20 images for training and 20 images for testing is followed for the DRIVE dataset. For datasets without predefined splits, a random selection of images is performed for training and testing. In the case of the STARE dataset, 5 random images were chosen for testing and the remaining 15 for training. CHASEDB utilized 21 randomly selected images for training and the remaining 7 for testing. The IOSTAR dataset involved 24 randomly chosen images for training and 6 images for testing. As for HRF, five random images from each set were used for testing, while the remaining 30 images were used for training.

Considering the limited size of the datasets, data augmentation techniques, including vertical and horizontal flipping and random rotation [146], were applied to augment the training data. To ensure consistency, min-max normalization is performed to normalize the pixel values of all images to the range of 0 to 1.

4.2.2 Experimental Setup

In the of process of BTU-Net, the initial random binary population size N_p is set to 50. After determining the fitness scores of the population, the top 20 learners with the highest fitness scores were selected for the next generation. Therefore, starting from the 1st generation, the population size N_p is set to 20. The number of generations T_g is set to 30. Additionally, a pre-termination criterion Tc is used, which specifies that the algorithm would terminate if the top five learners remained unchanged for three consecutive generations. The proposed method utilized the Adam as optimizer [147] with a learning rate of 0.001 and a batch size of 2. The model is implemented using PyTorch v1.9.0 and executed on a 2.2GHz Intel Xeon®CPU and a Quadro P5000 graphics card with 16 GB of memory.

Evaluation metrics: The Dice score [139] (Equation 4.5) is used as a fitness function $f()$ to evaluate the networks, where A represents the ground truth image, B represents the predicted segmented image, $|A|$ and $|B|$ represents the cardinality of A and B. Furthermore, accuracy (Equation 4.6), recall (Equation 4.7), specificity (Equation 4.8), precision (Equation 4.9) and AUROC (Area Under the ROC Curve) are utilized as metrics during

evaluation. Where F_n, F_p, T_p, T_n indicates false negative, false positive, true positive, and true negative, respectively.

$$Dicescore = \frac{2 \times |A \cap B|}{|A| + |B|} \tag{4.5}$$

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_n + F_p} \tag{4.6}$$

$$Recall = \frac{T_p}{F_n + T_p} \tag{4.7}$$

$$Specificity = \frac{T_n}{F_p + T_n} \tag{4.8}$$

$$Precision = \frac{T_p}{F_p + T_p} \tag{4.9}$$

4.2.3 Comparison with Existing Methods

During the experiments, a denoising technique (developed in Chapter 3) is initially applied as a preprocessing step on the original spleen dataset. This denoising process effectively removed noise from the images, resulting in clearer views, as depicted in Figure 4.4. Subsequently, the proposed method is evaluated on both the original and denoised images, and its performance is compared to existing segmentation models, including U-Net [21], Segnet [148], DeepLabv3+ [149], and CPFNet [150]. The segmentation results for both the original and denoised image datasets are presented in Figure 4.5 to showcase the importance of denoising in the segmentation task. Furthermore, Table 4.2 provides the experimental results, highlighting the segmentation performance of the proposed model in comparison to state-of-the-art models. It is evident from both the quantitative and visual results that the proposed method outperforms existing models across all metrics. Notably, the consistent improvement observed in denoised images when compared to original images.

Furthermore, the evaluation is expanded to include five retinal vessel datasets in order to showcase the effectiveness of the proposed model. Based on the demonstrated improve-

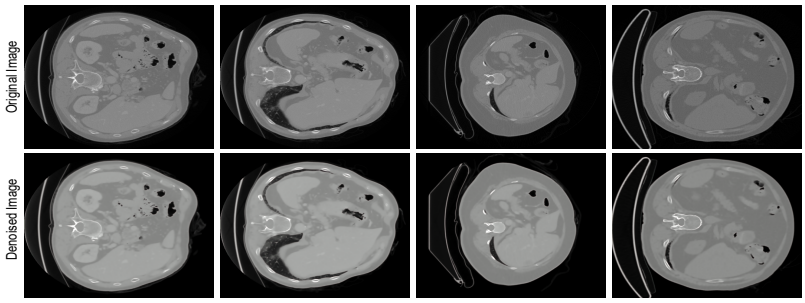


Figure 4.4: Original vs Denoised images.

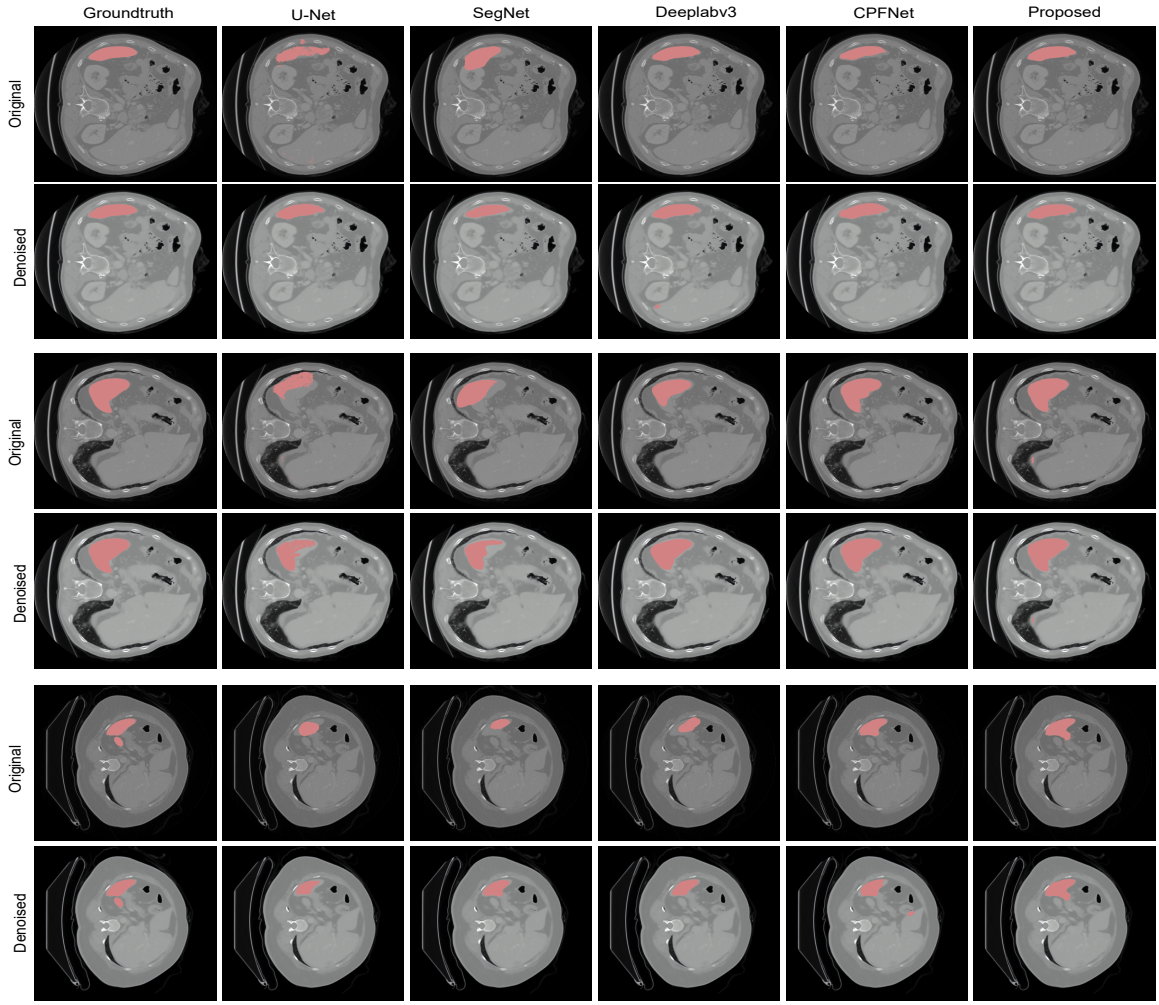


Figure 4.5: Visual results of Original and Denoised images of Spleen dataset.

ment in segmentation performance through denoising, denoising is performed on a dataset followed by segmentation. The segmented results of the proposed model were compared with the existing models, such as U-Net [21], CE-Net [151], FC-Densenet [152], CS2-Net

Table 4.2: Comparison with existing models on Spleen Dataset.

Model	Original					Denoised				
	Accuracy	Recall	Specificity	Precision	Dice score	Accuracy	Recall	Specificity	Precision	Dice score
U-Net	0.9689	0.8550	0.9712	0.8451	0.8325	0.9737	0.8601	0.9785	0.8555	0.8427
Segnet	0.9780	0.8685	0.9816	0.8559	0.8450	0.9860	0.8685	0.9894	0.8606	0.8514
DeepLabv3+	0.9805	0.8694	0.9827	0.8656	0.8608	0.9956	0.8743	0.9882	0.8662	0.8709
CPFNet	0.9845	0.8785	0.9826	0.8688	0.8612	0.9973	0.8843	0.9888	0.8702	0.8757
Proposed	**0.9886**	**0.8853**	**0.9835**	**0.8729**	**0.8699**	**0.9985**	**0.8855**	**0.9902**	**0.8734**	**0.8778**

Note: The highest values for original images and denoised images are shown in bold.

[153], FR-UNet [154], IterNet [52], and Genetic U-net [58]. All these models are trained in the same environment as the discovered architecture to ensure a fair comparison. From Table 4.3, it can be observed that the proposed model outperforms other existing models on all five datasets. The segmented results of the proposed model and existing models on five datasets are illustrated in Figures 4.6, 4.7, 4.8, 4.9, 4.10. Two small frames have been extracted from the resultant segmented images to gain a better insight into the preservation of details. These frames are depicted below, enclosed within blue and green boxes (Figures 4.6, 4.7, 4.8, 4.9, 4.10). These figures show the effectiveness of the proposed model in extracting complex structures from retinal images compared to existing models. These quantitative and qualitative results demonstrate that the proposed BTU-Net model outperforms other segmentation models. Additionally, the results illustrate the efficacy of NAS in designing neural network architectures for medical image segmentation, outperforming manually designed architectures across all metrics.

To compare the effectiveness of the BTLBO algorithm with other metaheuristic optimization algorithms, a few recent algorithms such as Equilibrium Optimizer Algorithm (EOA) [155], Grasshopper Optimization Algorithm (GOA) [156], Chameleon Swarm Algorithm (CSA) [157], and Marine Predators Algorithm (MPA) [158] were chosen. To evaluate the overall performance of algorithms, two metrics (mean and standard deviation) are used. The significance of findings is also assessed using the non-parametric statistical test known as the Wilcoxon rank-sum test [159]. Each algorithm is run for 30 generations, and each model for 30 epochs during training. The BTLBO algorithm produced the highest mean, standard deviation, and fitness score compared to other algorithms, as shown in Table 4.4. Additionally, Table 4.5 shows that the Wilcoxon rank-sum test's p-values are less than 0.05. This demonstrates the statistical significance of the results. The experimental re-

Table 4.3: Comparison with existing models on retinal vessel segmentation datasets.

Dataset	Model	Accuracy	Recall	Specificity	Precision	Dice score	AUROC
IOSTAR	U-Net	0.9703	0.8324	0.9821	0.8387	0.8012	0.9013
	FC-Densenet	0.9688	0.8012	0.9662	0.8195	0.7853	0.8757
	CE-Net	0.9703	0.8013	0.9756	0.8284	0.7896	0.8834
	CS2-Net	0.9727	0.8101	0.9843	0.8371	0.7936	0.8987
	FR-UNet	0.9730	0.8249	0.9827	0.8320	0.8063	0.9086
	IterNet	0.9742	0.8322	0.9862	0.8417	0.8120	0.9104
	Genetic U-net	0.9731	0.8286	0.9863	0.8438	0.8054	0.9074
	BTU-Net	**0.9747**	**0.8388**	**0.9868**	**0.8473**	**0.8126**	**0.9118**
CHASEDB	U-Net	0.9746	0.8181	0.9850	0.7877	0.7936	0.9015
	FC-Densenet	0.9755	0.8257	0.9854	0.7929	0.8031	0.9055
	CE-Net	0.9738	0.8055	0.9850	0.7825	0.7892	0.8952
	CS2-Net	0.9744	0.8272	0.9842	0.7792	0.7940	0.9057
	FR-UNet	0.9750	0.8315	0.9835	0.7929	0.8024	0.9119
	IterNet	0.9757	0.8233	0.9858	0.7958	0.8053	0.9046
	Genetic U-net	0.9753	0.8374	0.9845	0.7825	0.8048	0.9110
	BTU-Net	**0.9787**	**0.8405**	**0.9864**	**0.7987**	**0.8109**	**0.9124**
HRF	U-Net	0.9716	0.8170	0.9841	0.8051	0.8019	0.9001
	FC-Densenet	0.9711	0.8245	0.9839	0.8019	0.7999	0.9042
	CE-Net	0.9704	0.8125	0.9833	0.7940	0.7901	0.8979
	CS2-Net	0.9725	0.8091	0.9858	0.8132	0.8002	0.8974
	FR-UNet	0.9726	0.8270	0.9842	0.8133	0.8081	0.9048
	IterNet	0.9718	0.8258	0.9828	0.8125	0.8032	0.9029
	Genetic U-net	0.9728	0.8269	0.9849	0.8116	0.8071	0.9059
	BTU-Net	**0.9782**	**0.8274**	**0.9864**	**0.8187**	**0.8174**	**0.9084**
DRIVE	U-Net	0.9667	0.8125	0.9807	0.8153	0.8075	0.8916
	FC-Densenet	0.9672	0.7947	0.9799	0.8091	0.8060	0.8894
	CE-Net	0.9631	0.7915	0.9798	0.8013	0.7994	0.8856
	CS2-Net	0.9656	0.8174	0.9813	0.8172	0.8056	0.8939
	FR-UNet	0.9659	0.8209	0.9803	0.8015	0.8097	0.8957
	IterNet	0.9661	0.8213	0.9785	0.8003	0.8116	0.8989
	Genetic U-net	0.9677	0.8221	0.9834	0.8236	0.8119	0.8950
	BTU-Net	**0.9689**	**0.8249**	**0.9844**	**0.8264**	**0.8178**	**0.9030**
STARE	U-Net	0.9621	0.7907	0.9786	0.7999	0.7906	0.8792
	FC-Densenet	0.9674	0.7936	0.9825	0.8053	0.7962	0.8881
	CE-Net	0.9623	0.8045	0.9751	0.8020	0.7919	0.8888
	CS2-Net	0.9641	0.7998	0.9785	0.8006	0.8019	0.8984
	FR-UNet	0.9682	0.7972	0.9832	0.8107	0.8005	0.9006
	IterNet	0.9699	0.7991	0.9849	0.8170	0.8036	0.8920
	Genetic U-net	0.9706	0.8194	0.9843	0.8195	0.8136	0.9018
	BTU-Net	**0.9793**	**0.8284**	**0.9862**	**0.8197**	**0.8189**	**0.9034**

Note: The best values for each dataset are highlighted in bold.

sults show that the proposed method achieves optimal results when the BTLBO algorithm is used for the segmentation task.

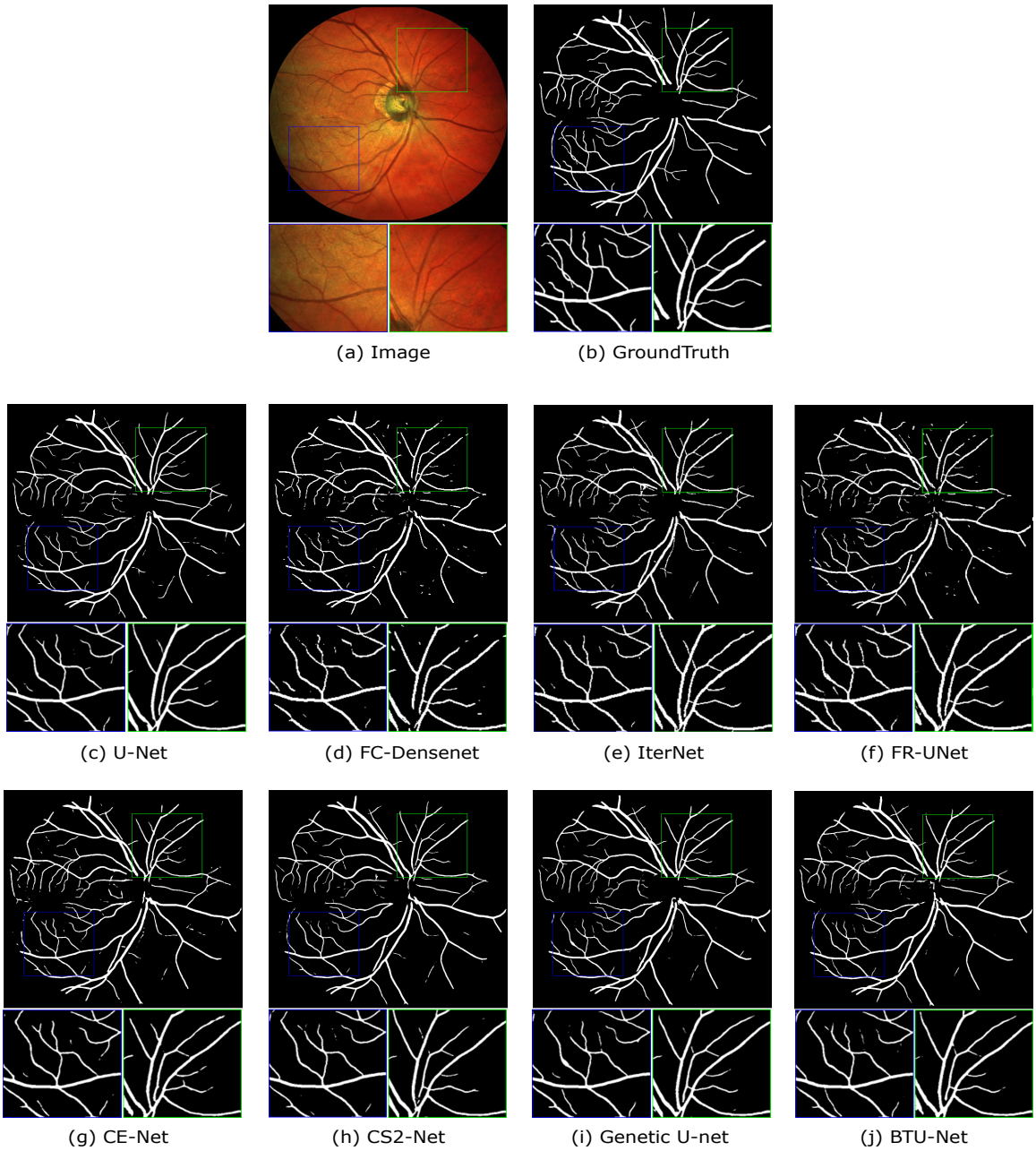


Figure 4.6: Visual segmentation results of IOSTAR dataset.

4.3 Discussion & Analysis

In this section, a comprehensive analysis and discussion of the proposed model components are presented, based on the obtained search results.

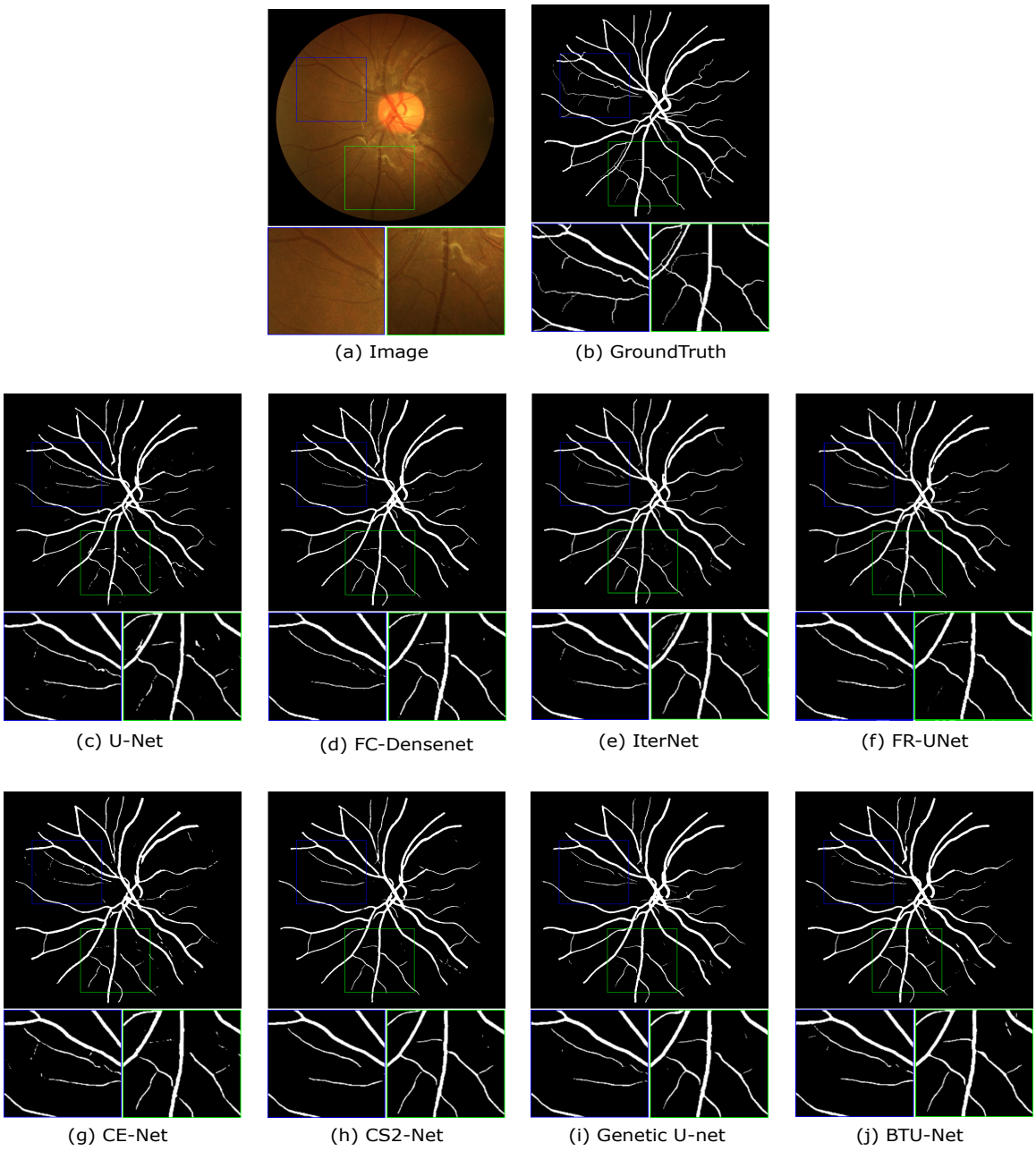


Figure 4.7: Visual segmentation results of CHASEDB dataset.

4.3.1 Performance analysis of Search Process

Figure 4.11 illustrates the maximum fitness scores achieved by the BTU-Net during its search process across generations. From the first generation, the scores of the learners gradually improve and converge at approximately subsequent generations. The pre-termination condition (Tc) states that the algorithm will terminate if the top five learners do not change

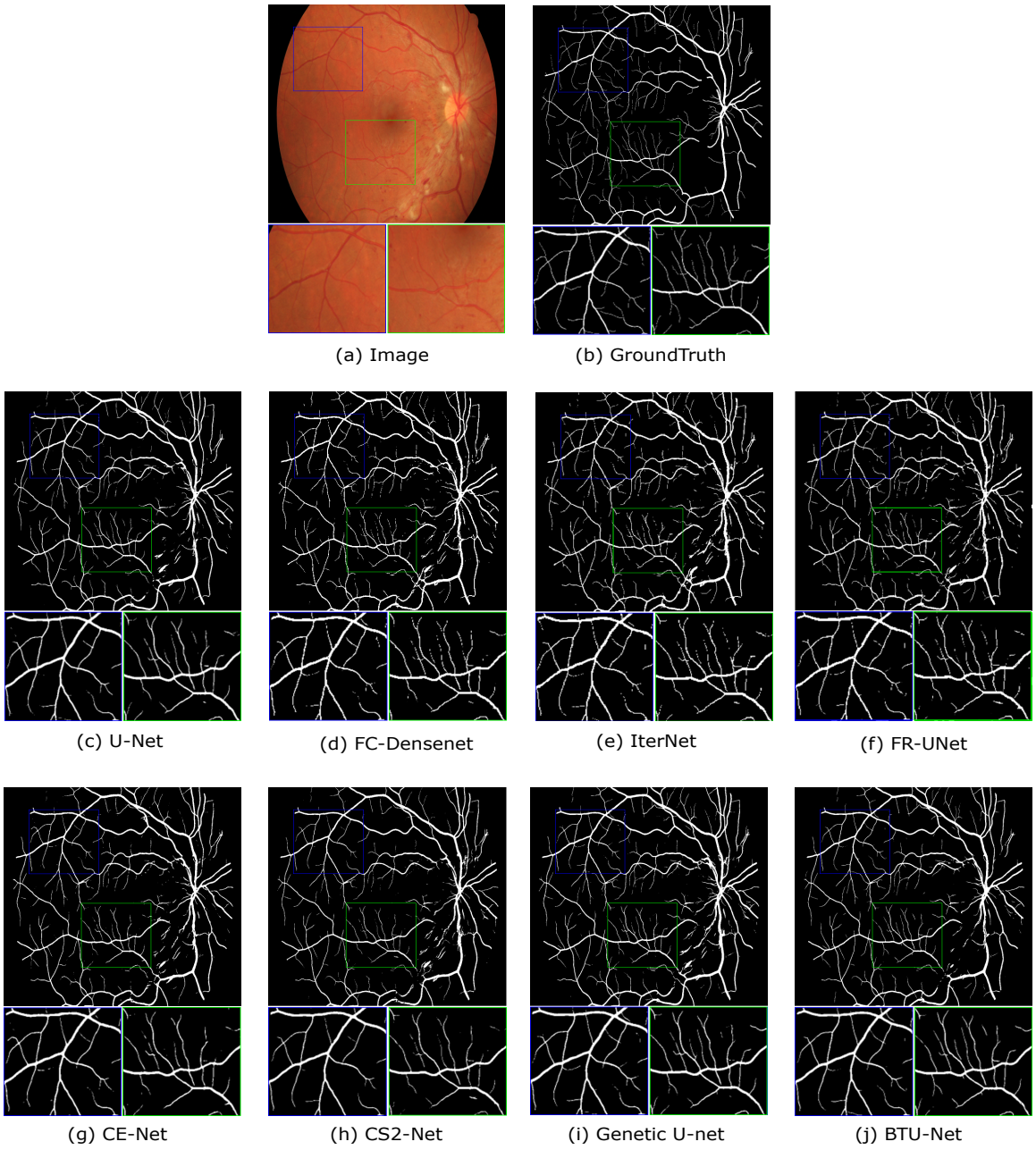


Figure 4.8: Visual segmentation results of HRF dataset.

for the following three generations. Furthermore, the last generation's top five network architectures from the population have been selected and trained on all datasets, and the optimal model outputs are taken. Figure 4.12 shows the performance of the proposed method on the training and test sets for each dataset.

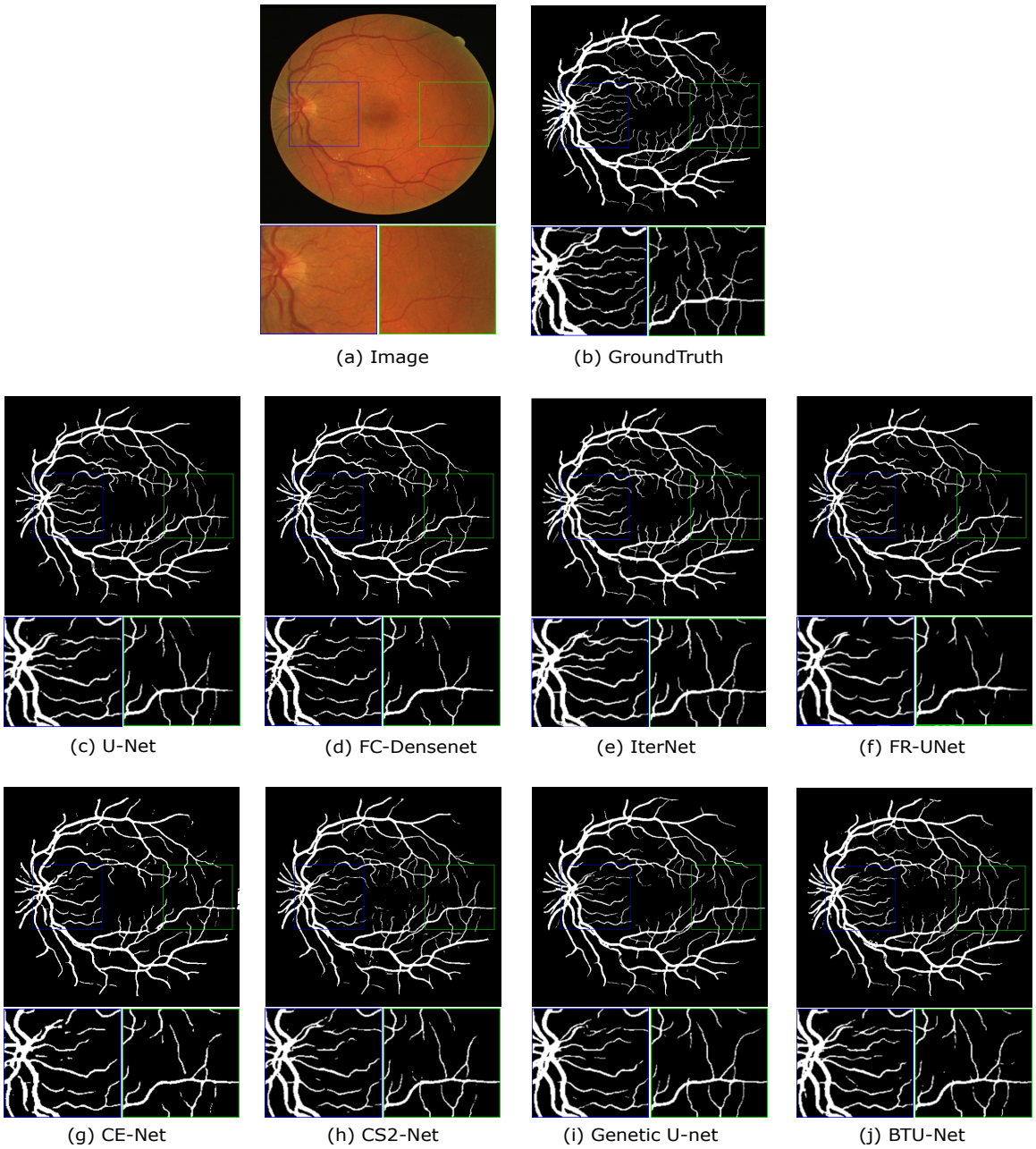


Figure 4.9: Visual segmentation results of DRIVE dataset.

4.3.2 Cache enabled BTLBO

In the search process, the cache component is used to speed up the fitness evaluation of a population based on the following aspects: 1) learners who survive into the following generation (if architecture remains unchanged) do not need to re-evaluate their fitness, and 2) the architecture that has been evaluated can be regenerated in future generations. The

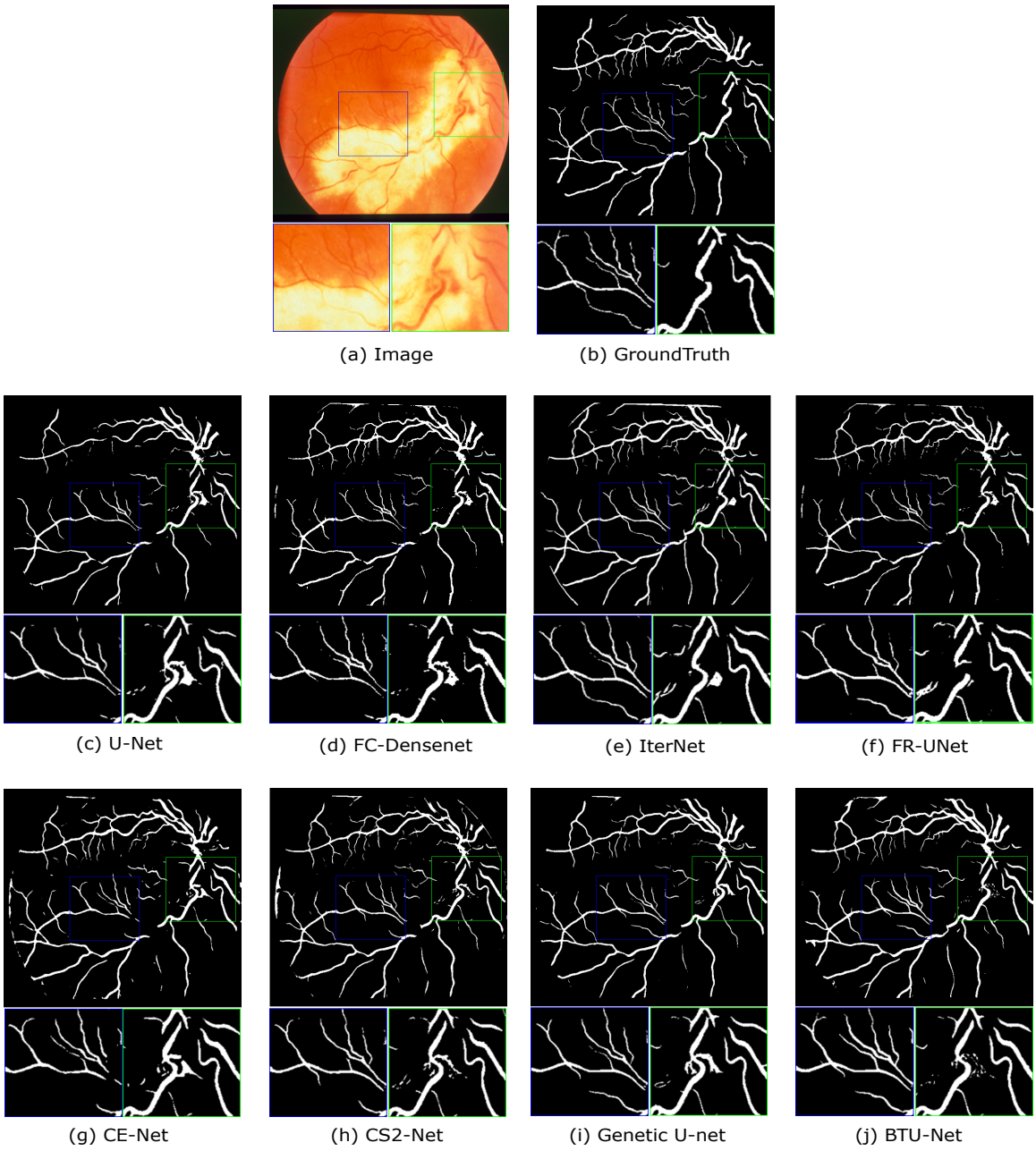


Figure 4.10: Visual segmentation results of STARE dataset.

BTLBO algorithm generates two learners for each learner from two phases in one generation, so there can be a chance that the same learner can reappear in any generation during the search process. In that case, the cache will retrieve the saved fitness score of the particular learner by avoiding re-execution. Generally, a cache system should take its size seriously and provide details to discuss the conflicting problem caused by duplicate keys.

Table 4.4: Mean, standard deviation and max fitness score values obtained from 30 runs.

Algorithm	Mean	Standard deviation	Max fitness score
CSA	0.7720551211029644	0.013247007656711045	0.7820
MPA	0.7686482127341582	0.012701060322495477	0.7779
EOA	0.7667829428999998	0.010721337729667133	0.7735
GOA	0.7797869557545665	0.013104164794438189	0.7867
BTLBO	0.7813792232272889	0.013744832946680097	0.7938

Table 4.5: Wilcoxon rank-sum test p-values.

Algorithm	P-value
BTLBO vs CSA	3.463437399492168e-06
BTLBO vs MPA	3.338018333641698e-05
BTLBO vs EOA	3.391034943845661e-05
BTLBO vs GOA	3.478324915956094e-06

The cache component of the proposed approach is similar to a map data structure, with a string comprising the learner and the corresponding model's fitness value. Even though there are thousands of records, the cache component will take very little disc space. Thus, there is no need to be concerned about the size of the cache component.

4.3.3 Operations and Operation Sequences

The primary goal of the proposed method is to find the optimal node connection structure for each block in a U-shaped network. For this, a total of 32 nodes were developed and divided into pre-activation and post-activation nodes, as outlined in Table 4.1, in the search space. Figure 4.13 shows the frequency of each node that occurred in top architectures during the evolution. However, the pre-activation nodes (such as 31, 29, 23, and 15) have appeared more times than post-activation nodes in the search process. These values show

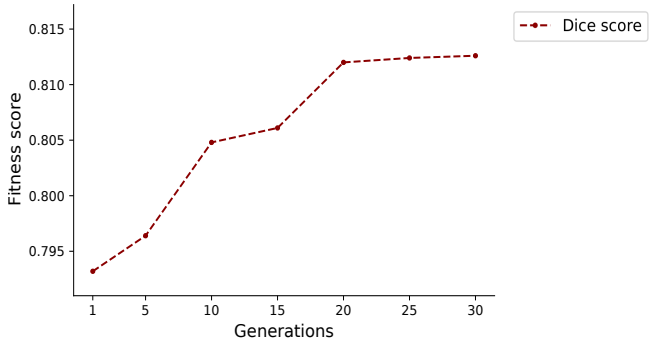
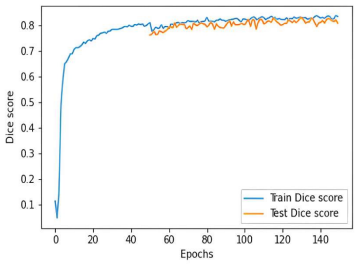
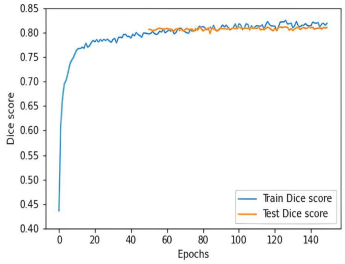


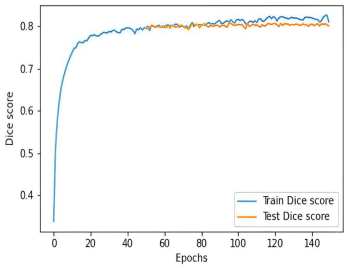
Figure 4.11: Generation wise fitness scores



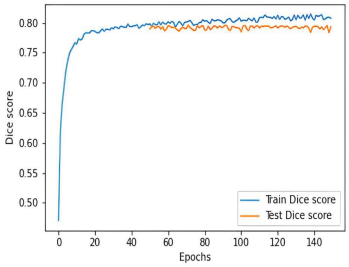
(a) CHASEDB



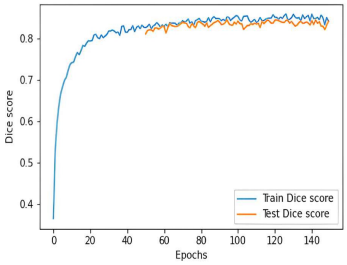
(b) DRIVE



(c) IOSTAR



(d) HRF



(e) STARE

Figure 4.12: Training and Test Dice scores

that applying the activation function before the convolution operation in encoder-decoder blocks can also improve the segmentation performance. The model with the highest Dice score during the evolution process is referred to as the optimal model, and its internal structure is the optimal structure. The optimal model discovered by the proposed model used nodes 23, 7, 12, 16, 15, 7, 29, 17, and 13 for 9 blocks in a modified U-Net. The experimental results show that replacing fixed block structures in U-shaped networks like U-Net with different block structures can enhance the segmentation accuracy.

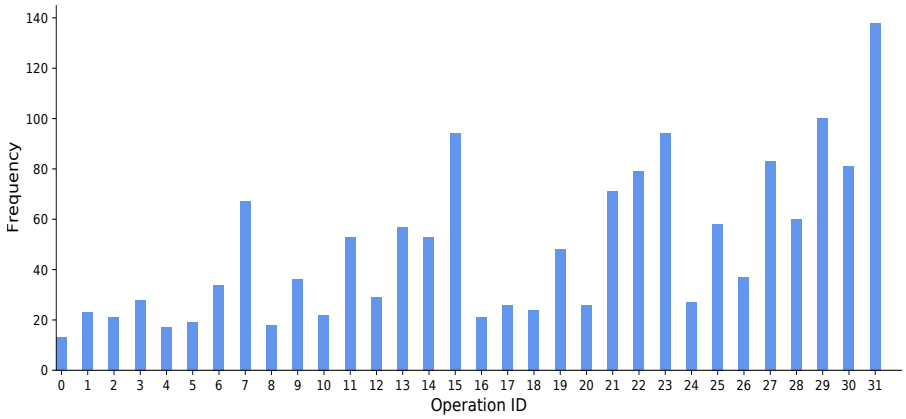


Figure 4.13: Frequency of node operations

4.3.4 Attention blocks

The attention mechanism is employed in the proposed model to capture the intricate vessels during segmentation. The results of the proposed method demonstrate the effectiveness and impact of utilizing an attention mechanism in a segmentation network. Moreover, the attention blocks Figure 4.2 that were used in the proposed work do not require computational overhead and a large number of model parameters as they use 1x1 convolutions. The frequency of each attention block in the search process is shown in Figure 4.14. Attention blocks 2 and 3, which have a skip connection with the downsampling layer, appeared many times during evolution. Furthermore, experiments were conducted using attention blocks, as well as utilizing both addition and concatenation operations. From Table 4.6, the proposed model performs much better using the attention blocks instead of addition or concatenation. This is because attention blocks can automatically learn more features.

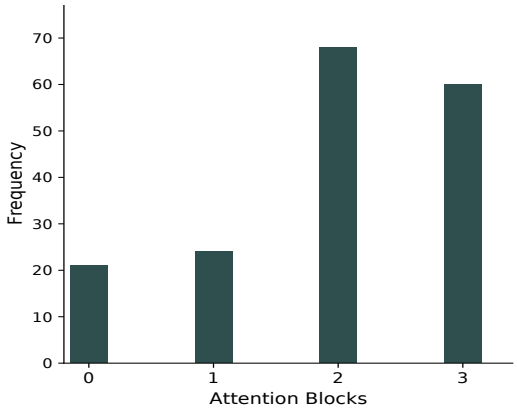


Figure 4.14: Frequency of attention blocks

Table 4.6: Performance with Attention blocks, Addition and Concatenation.

Operation	Dice score
Addition	0.7973
Concatenation	0.8027
Attention blocks	0.8198

4.3.5 Network Structure Parameters

In a U-shaped network, the downsampling and upsampling sections have a pooling layer after each block. However, choosing the appropriate pooling layer in a network can help to improve the model performance. Therefore, the proposed work has used the popular pooling layers such as average pooling and max pooling layers for downsampling, bilinear upsampling and transposed convolution for upsampling. The frequency distribution of each pooling layer during the search process is given in Figure 4.15. The average pooling layer in downsampling and transposed convolution layer in upsampling appeared more times during evolution. Furthermore, the Focal loss function has a higher frequency than the DiceBCE loss function in the search process, as shown in Figure 4.16.

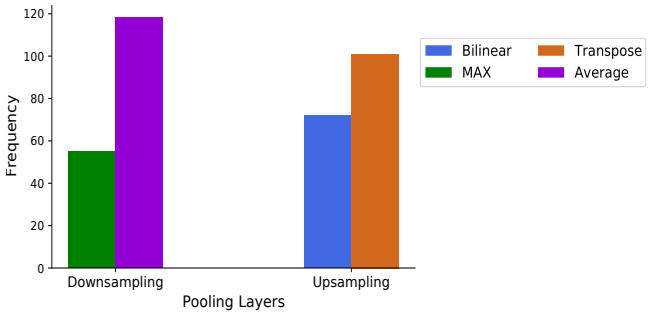


Figure 4.15: Frequency of Pooling layers

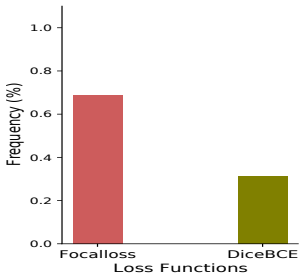


Figure 4.16: Frequency of Loss functions

4.3.6 Comparison of model size and parameters

The comparison between the size and parameters of the discovered architecture and other models is presented in Table 4.7. The discovered architecture has lesser parameters and execution time than U-Net but is comparatively greater than Genetic U-net due to convolution kernel sizes and attention blocks.

Table 4.7: Model size, number of parameters, and execution time comparison.

Model	Size	Params	Time
U-Net	120MB	31.03M	35.4ms
Genetic U-net	1.2MB	0.27M	27.5ms
BTU-Net	53MB	16.8M	31.1ms

4.4 Summary

In this chapter, a modified U-shaped (encoder-decoder) model based on the metaheuristic BTLBO algorithm, named BTU-Net, is introduced for 2D medical image segmentation

with denoising as a preprocessing step. The incorporation of the BTLBO algorithm optimizes each block within the U-shaped network, leading to enhanced segmentation performance. The attention mechanism is employed in BTU-Net to capture complex structures, allowing for more precise and accurate segmentation. To facilitate the search for an optimal model, a condensed but flexible search space is defined, enabling efficient exploration of various configurations. The utilization of a cache-enabled BTLBO algorithm significantly accelerates the fitness evaluation process by avoiding redundant training. By optimizing each block within the U-shaped network using the BTLBO algorithm, the proposed method achieves improved segmentation performance. Furthermore, the experimental results obtained from various medical image datasets highlight the significance of denoising as a preprocessing step in enhancing the performance of medical image segmentation. The results indicate the potential clinical applications of the proposed model in medical image segmentation tasks. In the next chapter, a metaheuristic based 3D medical image segmentation model will be discussed.

Chapter 5

To develop a metaheuristic based encoder-decoder model for 3D medical image segmentation

In this chapter, a metaheuristic based encoder-decoder model is presented, which has been specifically designed for 3D medical image segmentation using Chameleon Swarm Algorithm, leveraging insights gained from 2D segmentation in Chapter 4.

Chapter Organization: Section 5.1 provides the Preliminaries. The proposed methodology is presented in Section 5.2. The experimental results and analysis are provided in Section 5.3. A summary of the work is described in Section 5.5.

5.1 Preliminaries

In this section, the overview of the preliminaries, including the Chameleon Swarm Algorithm, which is relevant to the proposed method is discussed.

5.1.1 Chameleon Swarm Algorithm (CSA)

The CSA was recently introduced by Braik et al. [160] in 2021. This algorithm is inspired by chameleons' hunting and food-finding strategies. Chameleons are known for

their ability to change color and blend in with their surroundings, making them a highly specialized species. Although chameleons usually consume insects, they can survive in deserts, semi-desert, mountains, and lowland environments. Its food-hunting procedure includes many phases, including tracking the prey, chasing the prey with their sight, and attacking the prey. It has been validated on various benchmark functions, demonstrating competitive performance compared to other metaheuristic algorithms, including grey wolf optimization (GWO), PSO, and GA [23]. The effectiveness of CSA has also been evaluated in various optimization problems and engineering design applications, such as plant leaf disease identification [161], power engineering [162], feature selection [163] and image processing [164]. With its versatility and strong global search capabilities, CSA proves to be a valuable choice for optimizing functions and addressing practical problems. Based on its demonstrated versatility and effectiveness in various optimization problems, the CSA algorithm is chosen as a promising approach to enhance 3D medical image segmentation. The CSA algorithm comprises four steps: initialization, search for prey, eye rotation, and hunting, as given in Algorithm 5.3.

In this Algorithm 5.3, a random population X is initialized with a size of N_p chameleons in K-dimensional space, where each chameleon represents a candidate solution to a problem. Furthermore, Velocity V is initialized based on the chameleon's dropping tongue, lower bound Lb, and upper bound Ub are defined [lines 1 to 4]. Then,the fitness $f()$ is evaluated for each chameleon in the population X [line 5]. Then, an index of generation, t, which ranges from 0 to T_g (the number of generations), is assigned [lines 6 to 34]. In the search for prey step, prey is sought by the chameleons as they roam the desert and trees, causing their position to change accordingly [lines 9 to 19]. The perception of prey by the chameleon is determined by the probability Pr. The best position is denoted by $P_{i,j}$, while the global best position is denoted by Q_j, where j^{th} is the dimension of the i^{th} chameleon in the t^{th} generation. During the eye rotation step, the chameleons' eyes rotate independently, enabling them to explore the search space and locate prey [lines 20 to 24]. In the hunting step, chameleons primarily capture prey by extending their tongues [lines 25 to 31]. Subsequently, the fitness $f()$ is evaluated for each chameleon X_i in the population X [line 32]. This process is iterated for T_g iterations.

Algorithm 5.3 The pseudocode of the CSA algorithm [160].

Input: Population size N_p, Chameleon dimension K, Number of generations T_g

Output: Final Population X with Fitness scores $f(X)$

//Initialization

Ub and Lb are the upper and lower bounds

Initialize random population X with N_p chameleons and K dimension

Initialize the velocity V based on chameleon's dropping tongue

Evaluate the fitness $f()$ for each chameleon in the population X

Initialize $t = 0$

while $t < T_g$ **do**

  Define the iteration parameter function μ, inertia weight ω, and acceleration rate α, and position update probability Pr

//Search of prey

  for $i = 0, 1, \ldots N_p - 1$ **do**

    for $j = 0, 1, \ldots K - 1$ **do**

      Choose r, r_1, r_2, r_3 randomly in [0,1]

      if $r >= Pr$ **then**

        $X_{i,j} = X_{i,j} + m_1(P_{i,j} - Q_j)r_2 + m_2(Q_j - X_{i,j})r_1$

      else

        $X_{i,j} = X_{i,j} + \mu((Ub_j - Lb_j)r_3 + Lb_j)sgn(rand - 0.5)r_1$

      end if

    end for

  end for

//Eye rotation

  for $i = 0, 1, \ldots N_p - 1$ **do**

    Choose r_i randomly in [0,1]

    $X_i = Xr_i + X_i$

  end for

//Hunting
  for $i = 0, 1, \ldots N_p - 1$ **do**
    for $j = 0, 1, \ldots K - 1$ **do**
      $V_{i,j} = \omega V_{i,j} + c_1(Q_j - X_{i,j})r_1 + c_2(P_{i,j} - X_{i,j})r_2$
      $X_{i,j} = X_{i,j} + ((V_{i,j})^2 - (V_{i,j})^2)/(2\alpha)$
    end for
  end for
  Evaluate the fitness $f()$ for each chameleon (X_i) in the population (X)
  Update $t = t + 1$
end while

5.2 Methodology

The proposed method utilizes a 3D encoder-decoder CNN architecture consisting of three stages. Each stage includes an encoder block followed by a down-sampling layer and a decoder block followed by an up-sample layer with a residual skip connection [35], which connects the output of the encoder block to one of the inputs of the decoder block, facilitating the propagation of gradients and improving the flow of information. Additionally, a bridge block connects the blocks of the final stage blocks in the network. However, constructing these seven blocks is crucial as they play specific roles in the overall architecture of the effective analysis and processing of three-dimensional data. To find the optimal structure for each block in the network, the proposed NAS framework, named CS3DEA-Net, proposes a search space that encompasses various block structures and hyperparameters and employs a CSA algorithm by exploring different block structures and hyperparameters for 3D medical image segmentation.

5.2.1 Search space and Encoding

Search space: In Section 4.1.2, the focus is on searching for an optimal node structure within each block of the encoder-decoder architecture, which may improve the computational complexity for 3D images. In this work, the search space is composed of 8

different blocks, each comprising various activation and normalization layers. Table 5.1 describes the parameters and their corresponding ranges in the search space of the proposed method. The search space consists of 8 different blocks, 4 types of activation functions (MEMSWISH [165], ReLU [129], PReLU [166], and Leaky ReLU [167]), 4 types of normalization layers (Batch [112], Instance [131], Group [168], and None, i.e., no normalization layer), and 2 types of up-sample layers (Upsampling3D and ConvTranspose3D). Additionally, the search space includes 4 types of optimizers, including Adam [147], Adadelta [169], Adamax [147], and Stochastic Gradient Descent (SGD) [170], as well as 4 types of loss functions, including Dice loss [171], Dice focal loss [172], Focal loss [133], and Tversky loss [173]. The 8 distinct blocks were considered from various literature [69, 174, 175, 176, 177] to enhance the segmentation performance. Figure 5.1 displays these 8 distinct blocks. To enhance performance, more parameters can be included in the search space. However, this can lead to increased computational requirements. Therefore, an effective search space is chosen, allowing for the exploration of diverse architectures and training attributes.

In the proposed search space, the Block ID, activation function, normalization, and up-sample parameters are utilized to construct the backbone of the 3D encoder-decoder CNN network. Then the constructed 3D network is trained using the loss function and optimizer parameters. The CS3DEA-Net algorithm considers these parameters as decision variables and optimizes them for a specific dataset. This proposed algorithm aims to discover the optimal architecture that can yield the maximum fitness (Dice) score.

Table 5.1: Search space.

Parameter	Range
Block IDs	[Block1-Block8]
Activations	[MEMSWISH, ReLU, PReLU, Leaky ReLU]
Normalizations	[BATCH, INSTANCE, GROUP, None]
Up-sample	[ConvTranspose3D, Upsample3D]
Loss functions	[DiceLoss, DiceFocalLoss, FocalLoss, TverskyLoss]
Optimizers	[Adam, Adadelta, Adamax, SGD]

Encoding: The proposed 3D encoder-decoder CNN network features a distinct block structure, contrasting the fixed block structures employed in previous approaches. The network is composed of seven blocks, and each is assigned a unique block ID, activation function,

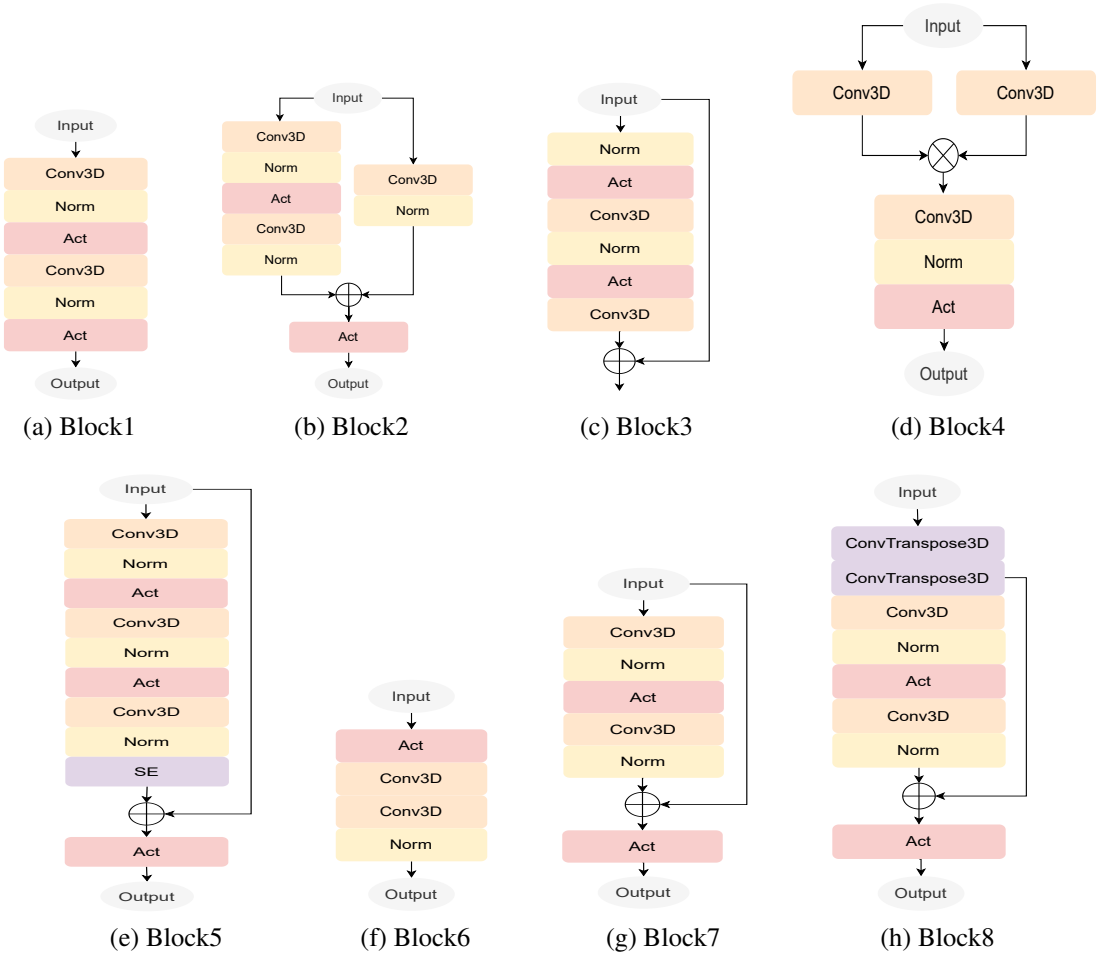


Figure 5.1: Structure of the 8 Blocks in the search space.

and normalization layer. To select a specific block, 3 bits are required for the block ID (with 8 blocks requiring 2^3 combinations), 2 bits for the activation function (with 4 functions requiring 2^2 combinations), and 2 bits for the normalization layer (with 4 layers requiring 2^2 combinations). Thus, for all seven blocks, 21 bits are needed for the block ID, 14 bits for the activation function, and 14 bits for the normalization layer. Additionally, 3 bits are utilized for selecting the up-sample layers for three stages, while 2 bits each are used for choosing the loss function and optimizer. The encoding scheme of a chameleon vector with dimension (K) 56 is depicted in Figure 5.2.

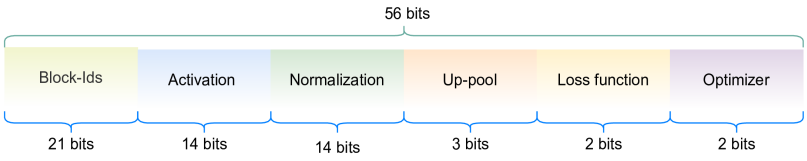


Figure 5.2: Encoding of a Chameleon vector.

5.2.2 CS3DEA-Net

The flow diagram of the proposed CS3DEA-Net for 3D medical image segmentation with the CSA algorithm, is shown in Figure 5.3. The CSA algorithm begins with the initialization of a random binary population consisting of N_p chameleons with a dimension of K. These chameleon undergo the steps outlined in Algorithm 5.3 within the CSA algorithm, generating a new population. Each chameleon vector generated by the CSA algorithm is then transformed into a binary vector using the sigmoid function, as described in Equation 4.3. This binary representation allows for encoding each chameleon vector into an encoder-decoder CNN network, as given in Section 5.2.1. The encoded network is then trained on the training (D_{train}) and validation (D_{val}) datasets. Following training, the trained model evaluates the Dice score (Equation 4.5) on the test (D_{test}) dataset. The Dice score serves as the fitness function within the CSA algorithm, and the CSA algorithm aims to maximize the fitness score through iterative generations of the population. The pseudocode for the fitness evaluation of each chameleon (X_i) and subsequent steps is provided in Algorithm 5.4.

To prevent the retraining of previously evaluated or trained models (chameleons), the proposed method incorporates a cache mechanism, as described in Section 4.1.3. The cache, denoted as Gc, is initialized as an empty global cache to store each chameleon (X_i) along with its corresponding fitness score $(f(X_i))$. The utilization of the cache is illustrated in Algorithm 5.4. Before encoding a model, the presence of the chameleon (X_i) in Gc is checked [lines 1-2]. If the chameleon is found in the cache, its fitness value $f(X_i)$ is retrieved from Gc. This step ensures that the model does not undergo unnecessary encoding, training, and evaluation, saving computational resources and time. However, if the chameleon is not present in the cache, it proceeds to be encoded into a encoder-

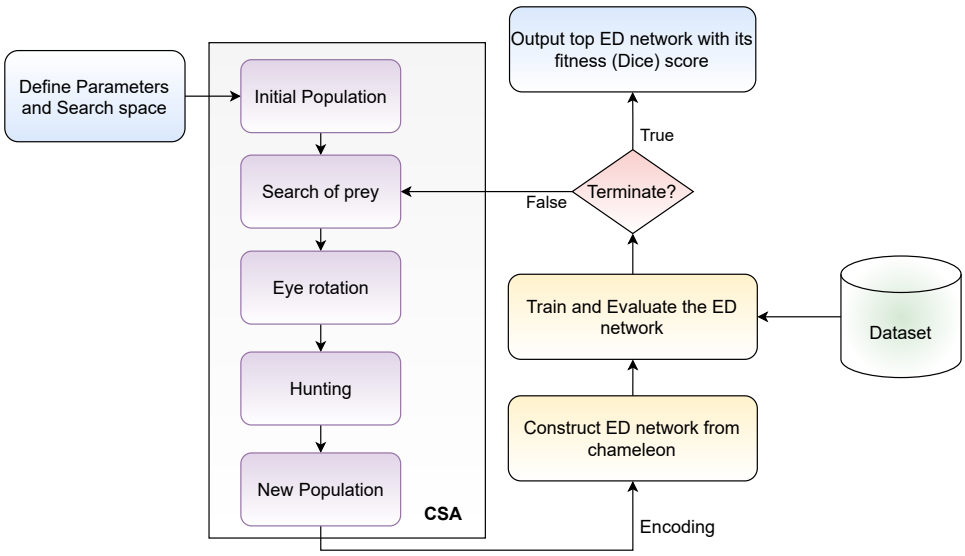


Figure 5.3: The flow diagram of CS3DEA-Net based on CSA.

decoder CNN model. The model is then trained and evaluated using the designated train and validation datasets [lines 3-17]. The training process involves training the encoded CNN model with the training dataset (D_{train}) for a specific number of epochs, typically denoted as M (considering 150 epochs). During training, after each even epoch, the Dice score is computed on the validation dataset (D_{val}). The model checkpoint with the highest Dice score is saved by storing the model weights [lines 10-16]. This allows for the retrieval of the best performing model during testing. During testing, the saved weights of the model are loaded, and the Dice score, which is fitness score ($f(X_i)$) for the chameleon X_i, is evaluated on the test dataset (D_{test}) [lines 19-20]. To ensure that the model does not undergo retraining, the chameleon (X_i) and its fitness score ($f(X_i)$) are added to the cache Gc [line 21].

Algorithm 5.4 The pseudocode of the fitness evaluation of a Chameleon.

Input: Chameleon (X_i), Global cache (Gc), Train (D_{train}), Val (D_{val}) and Test (D_{test}) datasets, Training epochs (M)

Output: Fitness score $f(X_i)$ of a chameleon X_i

1: **if** X_i in Gc **then**

2: $f(X_i)$ = query the fitness score $f(X_i)$ of chameleon from Gc

3: **else**

//Training
  Construct a CNN model from a Chameleon as given in section 5.2.1
  Set f_{best} = 0
  for $epoch = 1, 2, \ldots M$ **do**
    Train the model on training dataset D_{train}
//Validation
    if $epoch\%2 == 0$ **then**
      $Dice$ = Calculate the Dice score (Equation 6.7) on validation dataset D_{val}
      if $f_{best} < Dice$ **then**
        Update $f_{best} = Dice$
        Save the weights of current epoch
      end if
    end if
  end for
//Testing
  Load the saved weights to model
  $f(X_i)$ = Calculate the Dice score (Equation 6.7) as fitness score on test D_{test} dataset
  Add the chameleon X_i and fitness score $f(X_i)$ into the global Cache Gc
end if
return $f(X_i)$

5.3 Experiments

This section discusses the datasets and implementation details utilized in the evaluation process. Then, the proposed model's experimental results are provided and compared with existing state-of-the-art 3D segmentation models.

5.3.1 Datasets

The proposed CS3DEA-Net in this study is evaluated on publicly available two 3D medical image segmentation datasets, such as CT Spleen and MRI Heart segmentation datasets from

the Medical Segmentation Decathlon challenge [145]. The spleen segmentation dataset comprises 41 CT volumes with spleen body annotations, while the heart segmentation dataset includes 20 MRI volumes with corresponding annotations. Due to the limited number of images, various types of augmentation techniques, such as rotation and vertical and horizontal flips [178], were used to increase the number of training images for both datasets. Furthermore, the voxel intensities of the images were normalized to the range [0,1]. To focus on the valid body area of the images and labels, all zero borders are neglected and randomly sampled with volume sizes of [96,96,96]. The datasets were split into the train (D_{train}), validation (D_{val}), and test (D_{test}) sets, with the details of the dataset splitting described in Table 5.2.

Table 5.2: The number of volumes in train D_{train}, validation D_{val}, and test D_{test} datasets.

Dataset	Type	Volumes	D_{train}	D_{val}	D_{test}
Spleen	CT	41	29	6	6
Heart	MRI	20	13	3	4

5.3.2 Implementation Details

The CSA algorithm process is initiated with a population size N_p of 20 chameleons, which were represented as random binary vectors. The top 10 chameleons with the highest fitness scores were selected to update the population for the next generation, resulting in a population size N_p of 10 from the second generation onwards. The algorithm is designed to terminate termination criteria Tc, which is if the top three chameleons remain unchanged for three consecutive generations, or if the number of generations T_g reach to 30. The implementation of the proposed CS3DEA-Net framework utilized PyTorch v1.9.0 and Python 3.6. A learning rate of 0.0001 and a batch size of 2 were used for training. The training process is conducted on a system equipped with an Intel 2.2 GHz Xeon® Processor, a Quadro P5000 graphics card, and 16 GB RAM. For evaluating the segmentation performance of the proposed model, the Dice score (Equation 4.5) and Intersection Over Union (IoU) (Equation 5.1) metrics were employed. These metrics uses false negative, false positive, true negative, and true positive values, denoted as $F_{neg}, F_{pos}, T_{neg}, T_{pos}$, respectively.

$$IoU = T_{pos}/(F_{pos} + F_{neg}) \tag{5.1}$$

5.3.3 Experimental Results

The performance of the top model discovered by the proposed method is evaluated on two datasets: the CT Spleen dataset and the MRI Heart dataset. In order to assess its efficacy, the discovered model is compared against state-of-the-art 3D segmentation models, including 3D U-Net [59], V-net [60], SegResNet [69], Convnet [64], 3D Dense U-Net [68], NAS U-Net [15], 3D Evolve Net [83], and SwinUnetr [72]. To ensure a fair comparison, all models are trained and evaluated in the same environment. The results of the evaluation are presented in Table 5.3, showcasing the performance of each model on both the CT Spleen and MRI Heart datasets. The results demonstrate that the top model discovered by the proposed method outperforms all existing models on both datasets. This performance improvement highlights the effectiveness of the proposed CS3DEA-Net method in achieving accurate and robust segmentation results. To provide visual evidence of the segmentation quality, segmented images produced by the proposed model and the existing models on both datasets are displayed in Figures 5.4 and 5.5. These images visually illustrate the superior segmentation achieved by the proposed model compared to the other models, showcasing its ability to accurately delineate the desired structures. To further evaluate the preservation of details in the segmented images, small frames are cropped and extracted from the resultant segmented images of both the CT Spleen and MRI Heart datasets. These frames are depicted in Figures 5.6 and 5.7, respectively. The displayed frames clearly demonstrate the effectiveness of the proposed method in accurately extracting segmented images from the 3D data compared to the existing models. The level of detail and precision exhibited by the proposed model confirms its superior performance in accurately segmenting the desired structures. The combined qualitative and quantitative results firmly establish the superiority of the proposed CS3DEA-Net method over the existing segmentation models. Its ability to outperform state-of-the-art models in terms of accuracy and preservation of details highlights its potential for various applications in the field of medical image segmentation.

Table 5.3: Comparison with existing models on Spleen CT, Heart MRI datasets.

Models	Spleen		Heart	
	Dice score	IOU	Dice score	IOU
3D U-Net	0.841	0.750	0.722	0.692
V-net	0.865	0.792	0.766	0.709
SegResNet	0.933	0.876	0.844	0.792
Convnet	0.778	-	0.843	-
3D Dense U-Net	0.778	-	0.843	-
NAS U-Net	0.837	-	0.845	-
3D Evolve Net	0.941	-	0.887	-
SwinUNETR	0.892	0.814	0.805	0.780
CS3DEA-Net (Proposed)	**0.944**	**0.895**	**0.898**	**0.816**

Note: The highest values for each dataset is shown in bold.

5.4 Experimental Analysis

The maximum fitness scores achieved by CS3DEA-Net during its search process across generations are depicted in Figure 5.8. The graph illustrates that the chameleons' maximum fitness scores steadily improve with each subsequent generation and eventually converge at approximately subsequent generations. A pre-termination condition (Tc) is set, which halts the process if the top three chameleons remain the same for three consecutive generations.

To assess the effectiveness of the CSA algorithm in comparison to other metaheuristic optimization algorithms, recent algorithms such as GOA [156], BTLBO [179], and MPA [158] were selected. The significance of the findings is determined through the Wilcoxon rank-sum test [159], a non-parametric statistical test. Each algorithm is executed for 30 generations, and each model undergoes 30 epochs during the training process. As depicted in Table 5.4, the CSA algorithm demonstrates superior mean, standard deviation, and fitness scores compared to the other algorithms. Furthermore, Table 5.5 reveals that the p-values of the Wilcoxon rank-sum test are below 0.05, indicating the statistical significance of the results. These experimental outcomes strongly support the effectiveness of the proposed method in achieving optimal results for 3D segmentation tasks when employing the CSA algorithm.

The proposed method aims to determine the optimal parameters for constructing the encoder-decoder network and its hyperparameters to train a network. Figure 5.9 illustrates the frequency distribution of the blocks that appeared in the top architectures for the Spleen

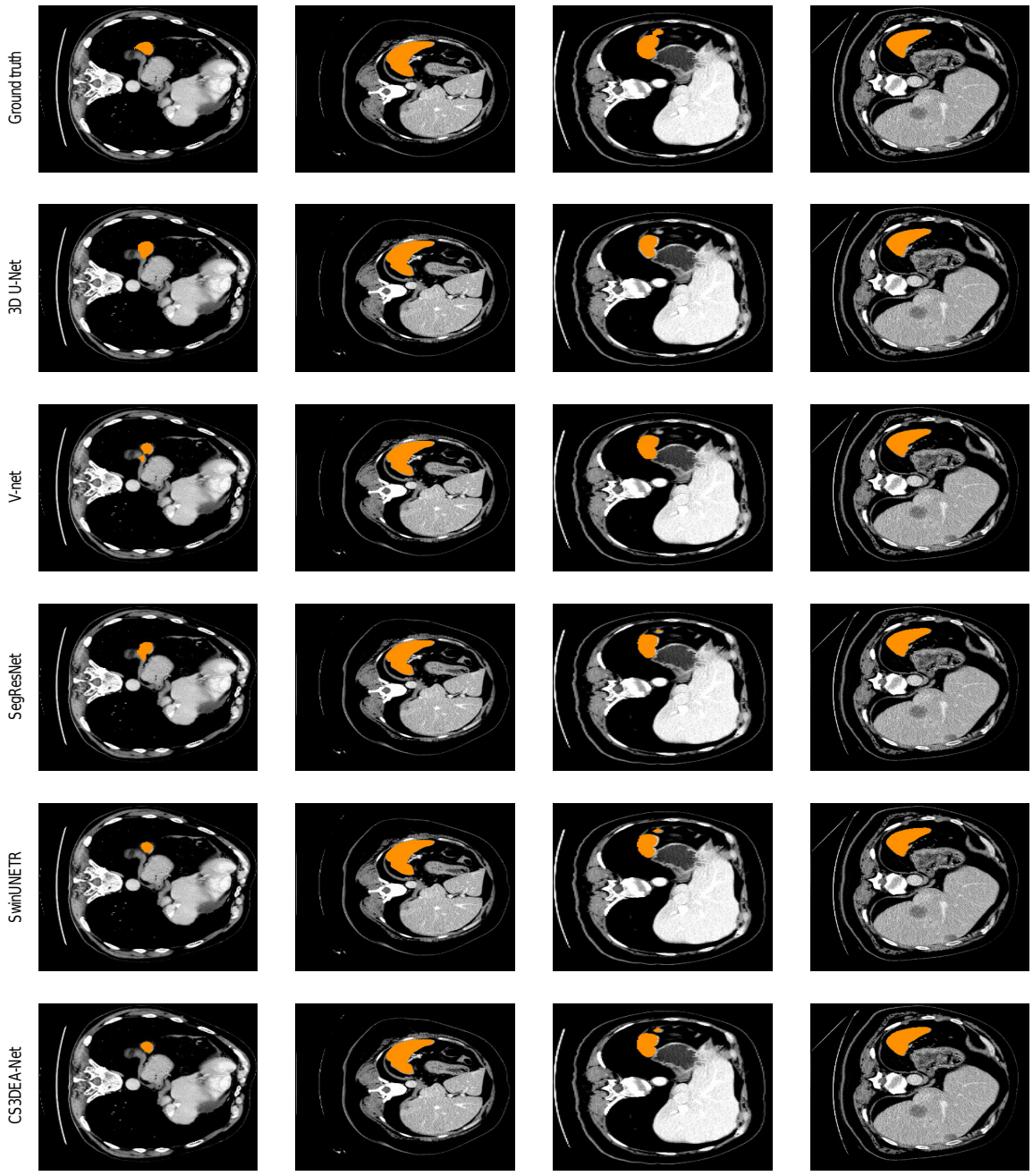


Figure 5.4: Visual segmentation results of Spleen dataset.

and Heart datasets in the search process. Notably, block IDs 8 and 7 appeared more frequently than the other blocks for both datasets during the evolution.

In addition to specifying the input-output relationship of each block, the choice of activation functions and normalization layers is crucial for ensuring neural network convergence and efficiency. However, it is also important to choose these components carefully to

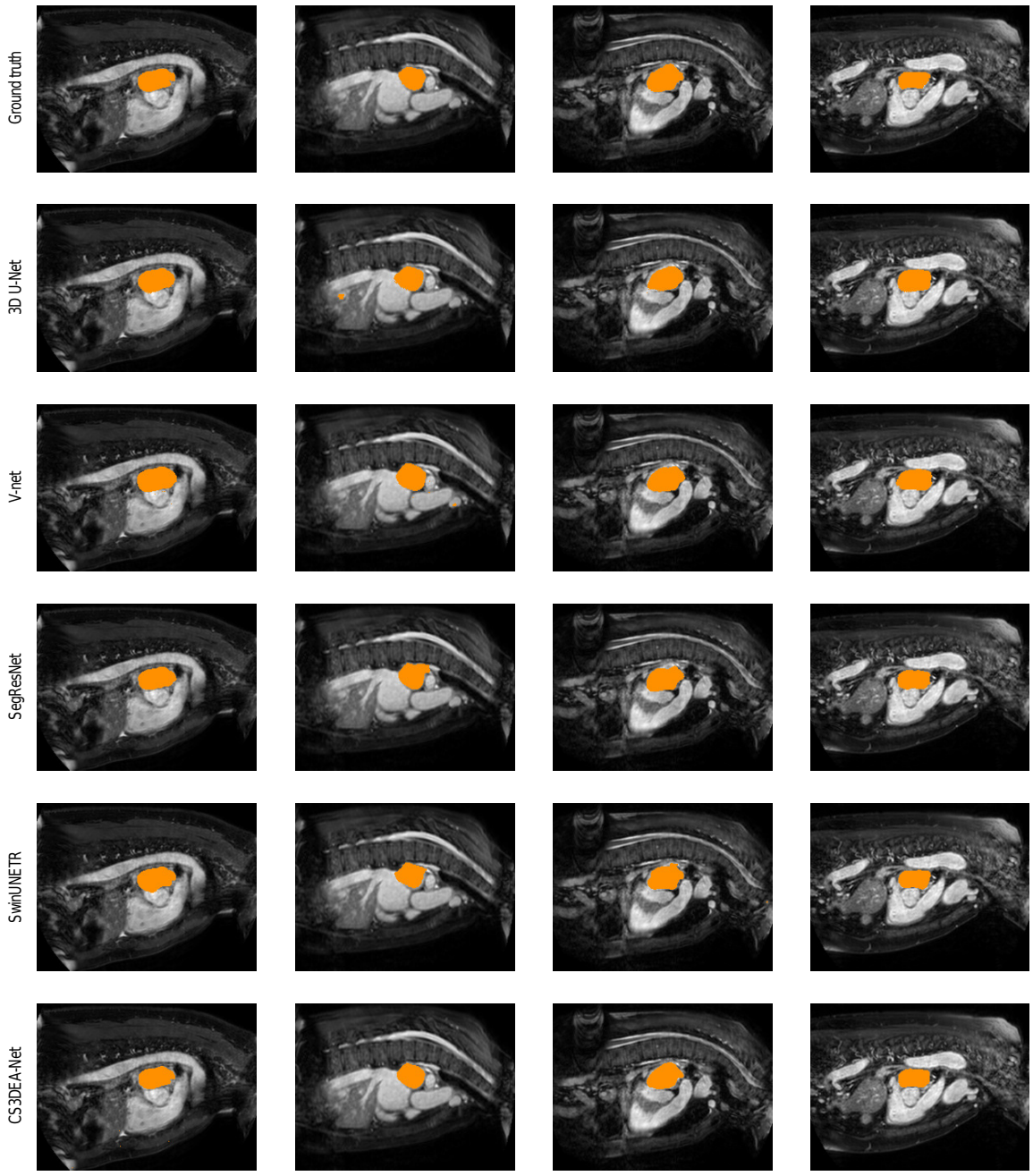


Figure 5.5: Visual segmentation results of Heart dataset.

prevent the network from getting stuck during training. During the search process, different datasets generate different types and numbers of activation functions and normalization layers, as illustrated in Figure 5.10 and Figure 5.11, respectively. Interestingly, the Leaky ReLU activation function is more frequently selected than any other activation function for both datasets.

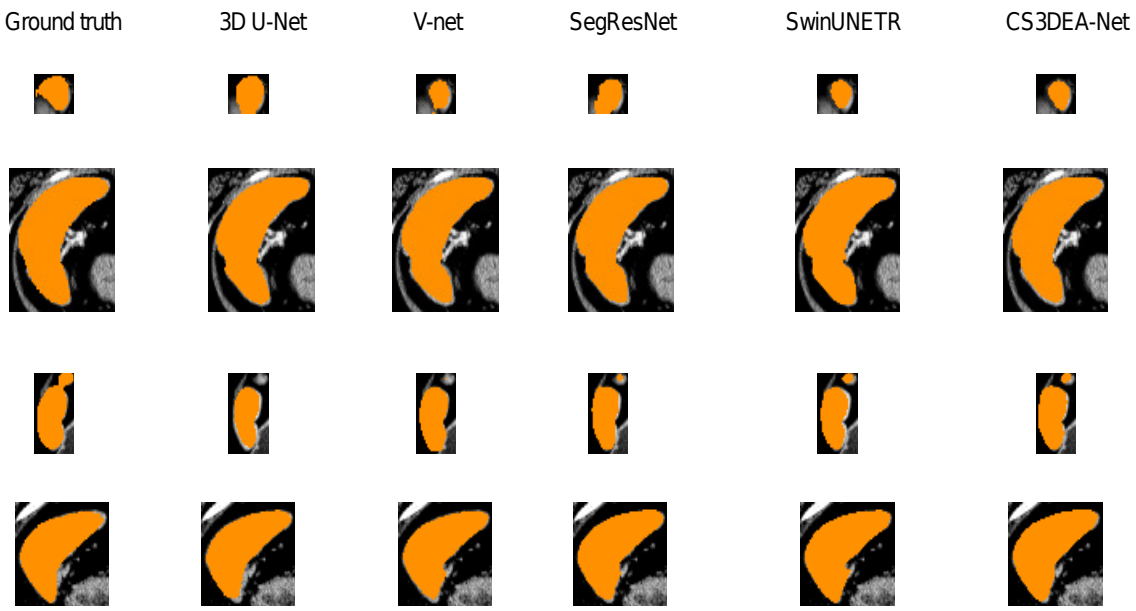


Figure 5.6: Cropped segmentation results of Spleen dataset.

Figure 5.7: Cropped segmentation results of Heart dataset.

The performance of an encoder-decoder network can be significantly affected by the pooling layers used in the up-sampling and down-sampling sections. Therefore, the proposed method has incorporated commonly used pooling layers, such as the upsampling3D and 3D transposed convolution layers, to achieve better segmentation results. Figure 5.12

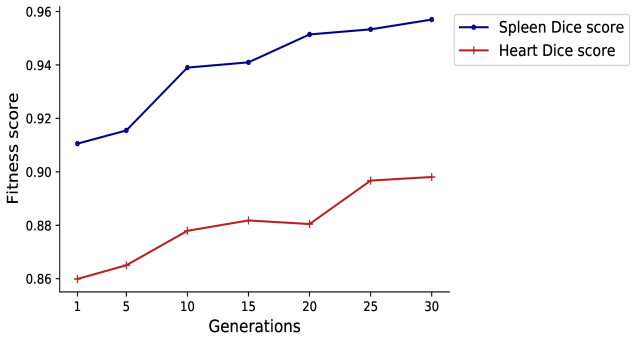


Figure 5.8: Generation wise fitness scores during evolution.

Table 5.4: Mean, standard deviation and max fitness score values obtained from 30 generations.

Algorithm	Mean	Standard deviation	Max fitness score
BTLBO	0.8720551211029644	0.02716676763359414	0.939
MPA	0.8547571383000001	0.14767009039123113	0.921
GOA	0.8470874825333333	0.09772537566730831	0.934
CSA	0.8966901123000002	0.01153792606106088	0.944

Table 5.5: Wilcoxon rank-sum test p-values.

Algorithm	p-value
CSA vs BTLBO	3.002414704136034e-04
CSA vs MPA	3.017518393579125e-04
CSA vs GOA	3.789619441580871e-06

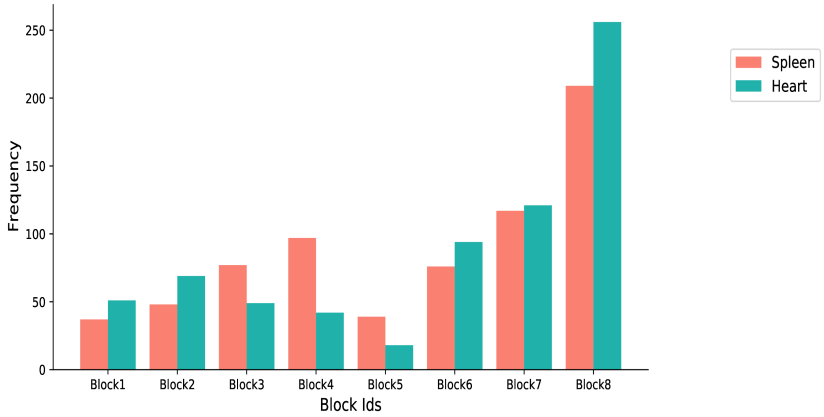


Figure 5.9: Frequency of blocks during evolution.

presents the frequency of each pooling layer used in the evolution. Notably, the upsampling3D layer is more frequently used than the 3D transposed convolution layer in the up-sampling section for both datasets during the evolution process.

The proposed method also selected the training parameters, including the optimizer

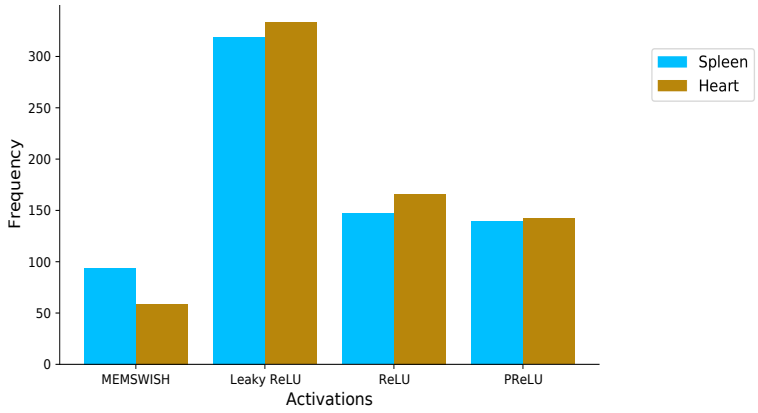


Figure 5.10: Frequency of activation functions during evolution.

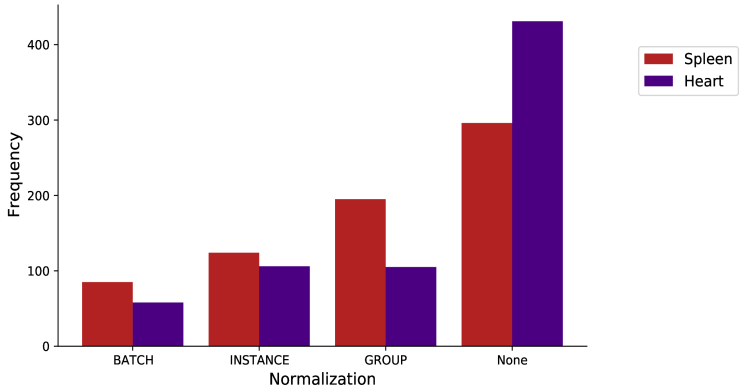


Figure 5.11: Frequency of normalization layers during evolution.

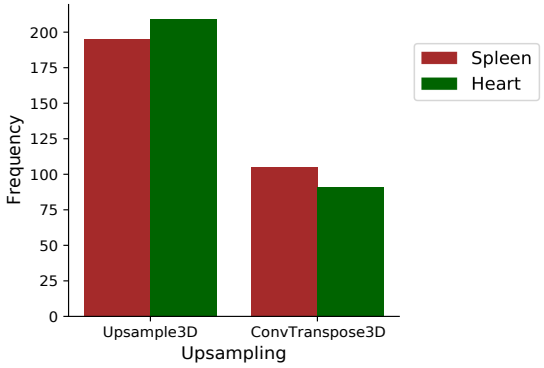


Figure 5.12: Frequency of up-sampling layers during evolution.

and loss function. The frequency distribution of optimizers and loss functions in the search process is presented in Figure 5.13 and Figure 5.14, respectively. During the evolution process, the Tversky loss function and SGD were more frequently selected than the other parameters on both datasets.

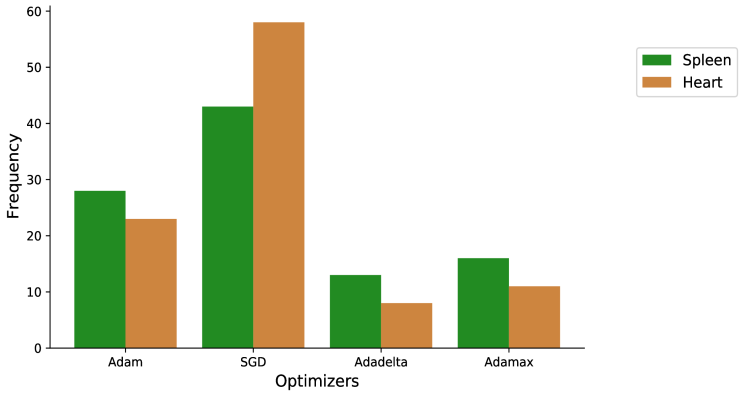


Figure 5.13: Frequency of optimizers during evolution.

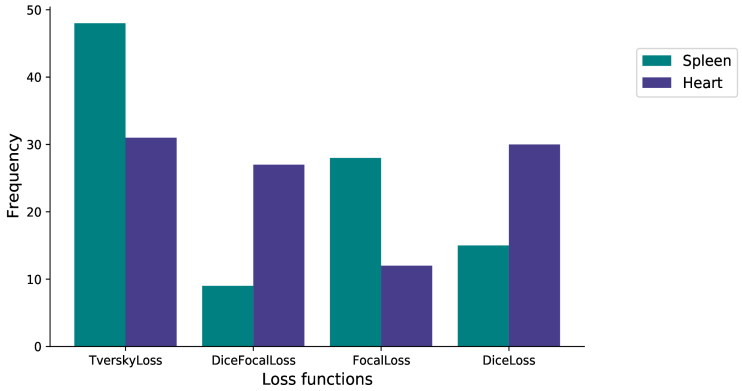


Figure 5.14: Frequency of loss functions during evolution.

Table 5.6 provides a comprehensive comparison of the discovered architecture and existing models in terms of size and the number of parameters. The results clearly demonstrate that the discovered architecture exhibits a significantly smaller size and requires fewer parameters compared to the other models. The discovered architecture has a size of 278.69MB. In contrast, the other existing models, such as 3D U-Net, V-net, SegResNet, and SwinUNETR, have sizes ranging from 314.71MB to 512.99MB. These findings highlight the compactness of the discovered architecture, making it more efficient in terms of storage requirements. Furthermore, the number of parameters for the discovered CS3DEA-Net architecture only requires 0.7 million parameters. In comparison, the existing models, such as 3D U-Net, V-net, SegResNet, and SwinUNETR, consist of parameters ranging from 3.8 million to 4.8 million. This substantial difference in the number of parameters further emphasizes the efficiency and effectiveness of the discovered architecture. The discovered

CS3DEA-Net architecture outperforms existing models not only in terms of segmentation accuracy, as discussed earlier, but also in terms of its compact size and reduced number of parameters.

Table 5.6: Model size and number of parameters.

Model	Size	Params
3D U-Net	512.99MB	4.8M
V-net	341.15MB	45.5M
SegResNet	314.71MB	4.7M
SwinUNETR	406.63MB	3.8M
CS3DEA-Net	278.69MB	0.7M

5.5 Summary

In this chapter, a metaheuristic-based NAS framework using an encoder-decoder architecture called CS3DEA-Net is introduced for 3D medical image segmentation. The proposed framework incorporates a search space that encompasses various types of block structures and training parameters required for constructing the network. The CSA algorithm is employed to enhance the performance of 3D medical image segmentation. It explores the designed search space to individually optimize the network blocks within an encoder-decoder architecture and identify the optimal training parameters for training the model. The performance of CS3DEA-Net is evaluated on both the Spleen and Heart 3D medical image segmentation datasets, revealing its superior performance compared to state-of-the-art models. These findings highlight the exceptional potential of CS3DEA-Net for clinical applications in the field of medical image segmentation. In the next chapter, the exploration of this approach will be extended by incorporating a multi-objective optimization algorithm for brain tumor detection.

Chapter 6

To design a multi-objective metaheuristic model for detecting brain tumors in 3D medical images

In this chapter, the work from Chapter 5 is extended by introducing EffiecientNet block, Transformer block, and Convolutional Neural Networks Meet Vision Transformer blocks into the search space for brain tumor detection in 3D medical images using a MO-ITLBO algorithm.

Chapter Organization: Section 6.1 provides the Preliminaries. The proposed methodology is presented in Section 6.2. The experimental results and analysis are provided in Section 6.3.4. A summary of the chapter is described in Section 6.4.

6.1 Preliminaries

This section offers an overview of the background knowledge of Multi-objective Neural Architecture Search, Improved Teacher Learner Based Optimization, and multi-objective optimization, which are utilized in the proposed method.

6.1.1 Multi-objective Neural Architecture Search (MO-NAS)

In recent years, the emergence of MO-NAS has attracted considerable attention for its ability to generate architectures that excel in diverse tasks. MO-NAS revolutionizes the optimization process by simultaneously considering multiple objectives, departing from the traditional single-objective approach. The primary objective of MO-NAS is to discover a collection of optimal architectures $\mathcal{A}_1, \mathcal{A}_2, \ldots \mathcal{A}_n$, that can optimize multiple objectives $f_1, f_2, \ldots, f_n$ simultaneously. These objectives often present conflicts, where enhancing one objective may have a detrimental impact on the others. For instance, reducing model size might result in decreased accuracy, while maximizing accuracy could lead to larger models. To address this inherent trade-off, MO optimization techniques are employed. These approaches excel in identifying a set of non-dominated solutions, referred to as the Pareto Front (PF). The PF represents the optimal balance between the conflicting objectives, offering a range of diverse and high-performing architectures. The common formulation of MO-NAS is as follows:

$$\mathcal{A}^* = argmin_{\mathcal{A} \in \mathbb{S}} \, f_i(\mathcal{A}, \omega^*(\mathcal{A})) \tag{6.1}$$

where $\mathcal{A}^* \in \mathbb{S}$ are non-dominated architectures from a search space $\mathbb{S}$, optimizes multiple objective functions f_i where $i = 2, 3 \ldots n$ and ω^* represents the learned parameters of $\mathcal{A}$.

6.1.2 Improved Teacher Learner Based Optimization (ITLBO)

The ITLBO algorithm [180] is an improved version of the standard TLBO algorithm, incorporating improvements such as the introduction of the adaptive teaching factor, number of teachers, and self-motivated learning concepts to improve the performance of the algorithm.

6.1.2.1 Initialization

In the Initialization step, a random population X is initialized with a size of N_p, consisting of learners ($i = 0, 1, \ldots N_p - 1$), as well as a specified number of subjects, i.e. decision variables (K), and a number of teachers (NT). The fitness $f(X)$ of the population X is then evaluated by calculating the fitness function.

6.1.2.2 Teacher Selection

The teacher selection phase ranks the evaluated population based on their fitness scores. For the maximization problem, the solutions are ranked in descending order, while for the minimization problem, they are ranked in ascending order. The best solution, denoted as $f(X^b)$, is selected as the chief teacher (T_l) of the class ($T_l = f(X^b)$) and remain teachers (T_s) are chosen by following Equation 6.2:

$$T_s = f(X^b) \pm f(X^b) * r, \tag{6.2}$$

where r is a random number chosen from range $[0, 1]$, $s = 2, 3, \ldots NT$. While the calculated value of T_s may not exactly match with the fitness value of existing solutions, in such instances, the existing solution with the fitness score closest to the calculated T_s value is selected.

6.1.2.3 Assignment of Learners to Teachers

In the assignment of learners to teachers, learners are grouped into various groups (s) based on their performance levels i.e., fitness scores. Each group within the population is assigned to a specific teacher selected from the pool of teachers (T_s). These teachers play a crucial role in guiding and enhancing the performance of their assigned groups. If a group's performance matches or exceeds their assigned teacher, the group is reassigned to another proficient teacher.

6.1.2.4 Teacher Phase

The teaching factor (T_f) determines the degree to which learners learn from the teacher, ranging from learning nothing to learning everything. In the standard TLBO algorithm, increasing the value of T_f accelerates the search process but diminishes the exploration capability. Conversely, decreasing the value of T_f enables a more precise search in smaller increments but may result in slower convergence. In the teaching-learning phenomenon, learners have the flexibility to learn from the teacher in varying proportions, indicating that the value of T_f can vary within its extremes. To capture this variability, the teaching factor is modified in ITLBO according to the Equation 6.3:

$$(T_f)_s^t = \begin{cases} 1 & \text{If } T_s = 0, \\ \left(\frac{f(X_i)}{T_s}\right)^t, & \text{Otherwise.} \end{cases}, \tag{6.3}$$

Here, $f(X_i)$ represents the fitness value of a learner i of a group s, and the result of a teacher in the same group s is denoted by T_s at iteration t. As a result, the value of T_f dynamically adjusts in the search process based on the relative performance of the learners and teachers.

In the teacher phase, learners can learn by interacting with the teacher or with their peers, similar to discussions and assignments during tutorial hours. This search mechanism is incorporated to allow students to increase their knowledge by collaborating with others as Equation 6.4:

$$X_{j,i}^t = \begin{cases} \left(X_{j,i}^t + r_1(X_{j,i}^s - (T_f)_s^t * M_{s,j})\right) + \left(r_2(X_{j,h}^t - X_{j,i}^t)\right), & \text{If } f(X_i) > f(X_h), i \neq h, \\ \left(X_{j,i}^t + r_1(X_{j,i}^s - (T_f)_s^t * M_{s,j})\right) + \left(r_2(X_{j,i}^t - X_{j,h}^t)\right), & \text{Otherwise.} \end{cases}, \tag{6.4}$$

where $h \in \{0, 1, \ldots N_p - 1\}$, $M_{s,j}$ represents the mean grade of the learner in group s for subject j, and $X_{j,i}^s$ represents the grade of the teacher associated with group s for subject j in a population X. The equation consists of two terms on the right side: the first term represents learning within the classroom, and the second term represents learning through

tutorials. The fitness of the population generated during the teacher phase is evaluated; then the individuals with the best fitness scores are retained in the population using the greedy approach.

6.1.2.5 Learner Phase

The learner phase incorporates self-learning, allowing learners to acquire knowledge independently. This phase enables exploration and knowledge enhancement through self-motivation. Learners update their positions by learning from other learners and through self-motivated learning as given in Equation 6.5:

$$X_{j,i}^{t+1} = \begin{cases} \left(X_{j,i}^{t} + r_1(X_{j,p}^{t} - X_{j,q}^{t}) + r_2(X_{j,i}^{s} - E_f * X_{j,p}^{t})\right) & \text{If } f(X_q) > f(X_p), q \neq p, \\ \left(X_{j,i}^{t} + r_1(X_{j,q}^{t} - X_{j,p}^{t}) + r_2(X_{j,i}^{s} - E_f * X_{j,p}^{t})\right), & \text{Otherwise.} \end{cases} \tag{6.5}$$

where $p, q \in \{0, 1, \ldots N_p - 1\}$, $M_{s,j}$, E_f represents the exploration factor, which is calculated as $E_f = round(1 + r_i)$, r_1 and r_2 are two random numbers in the range [0,1]. The first term on the right side of Equation 6.5 indicates learning through interactions with other learners, while the second term represents self-motivated learning. The fitness of the population is evaluated and the individuals with the highest fitness values are retained in the updated population using greedy approach.

The algorithm iterates until a termination condition is satisfied, often defined as a predetermined number of iterations. Figure 6.1 shows the ITLBO algorithm steps.

6.1.3 Multi-Objective Optimization

In Single-Objective Optimizations (SOOs), selecting the optimal individual for the population is a straightforward task, whereas, in Multi-Objective Optimizations (MOOs), the decision-making process is complex. In MOOs, two key concepts, namely Non-dominated sorting and Crowding distance sorting, are useful in the decision-making process [181].

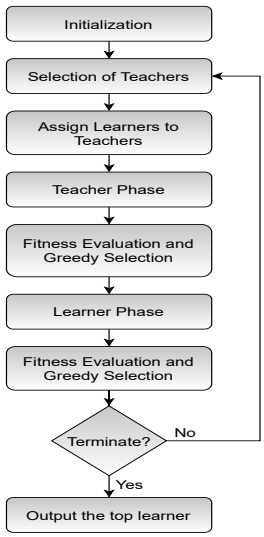


Figure 6.1: Flow chart of the ITLBO algorithm.

6.1.3.1 Non-dominated sorting

This approach sorts the solutions in a population by comparing the non-domination levels. In a minimization problem, two solutions are considered non-dominated if neither solution has a lower value than the other in all objectives. Each solution in a population is evaluated against every other solution to determine if it is dominant. For each solution i, two parameters are calculated: the first parameter, denoted as S_i, represents a set of solutions dominated by solution i, and the second parameter n_i denotes the set of solutions that dominate solution i. The solutions in the first level of non-dominated front $F1$ have $n_i = 0$, while solutions in subsequent fronts are determined by reducing the domination count of their corresponding sets [181]. This iterative process runs until all fronts such as $F2, F3, \ldots, Fn$ are identified [182].

6.1.3.2 Crowding distance sorting

The density of solutions surrounding a specific solution i in the population is estimated using this approach. The computation of crowding distance occurs in two scenarios. Firstly, when two solutions do not dominate each other, their crowding degrees are evaluated with respect to the non-dominated solutions in the population. The less crowded solution is selected as the new solution for the next generation. Secondly, if the number of solutions

exceeds the population size, the solutions located in the most crowded region are identified and eliminated from the population.

To compute the crowding distance, the population is initially sorted in ascending order of each objective function. The boundary solutions, which have the smallest and largest values, are assigned an infinite distance. For intermediate solutions, the distance is determined by the absolute normalized difference between the function values of adjacent solutions. This process is repeated for each objective function, and the crowding distance is computed by summing the individual distance values for each objective. Prior to calculating the crowding distance, each objective function is normalized [181]. This enables an accurate assessment of solution density and aids in efficient decision-making within the algorithm [182].

6.2 Methodology

This section introduces the proposed search space, encoding process and presents the proposed method, called 3DMOEA-Net, which uses a multi-objective ITBLO algorithm to build the optimal network for brain tumor detection.

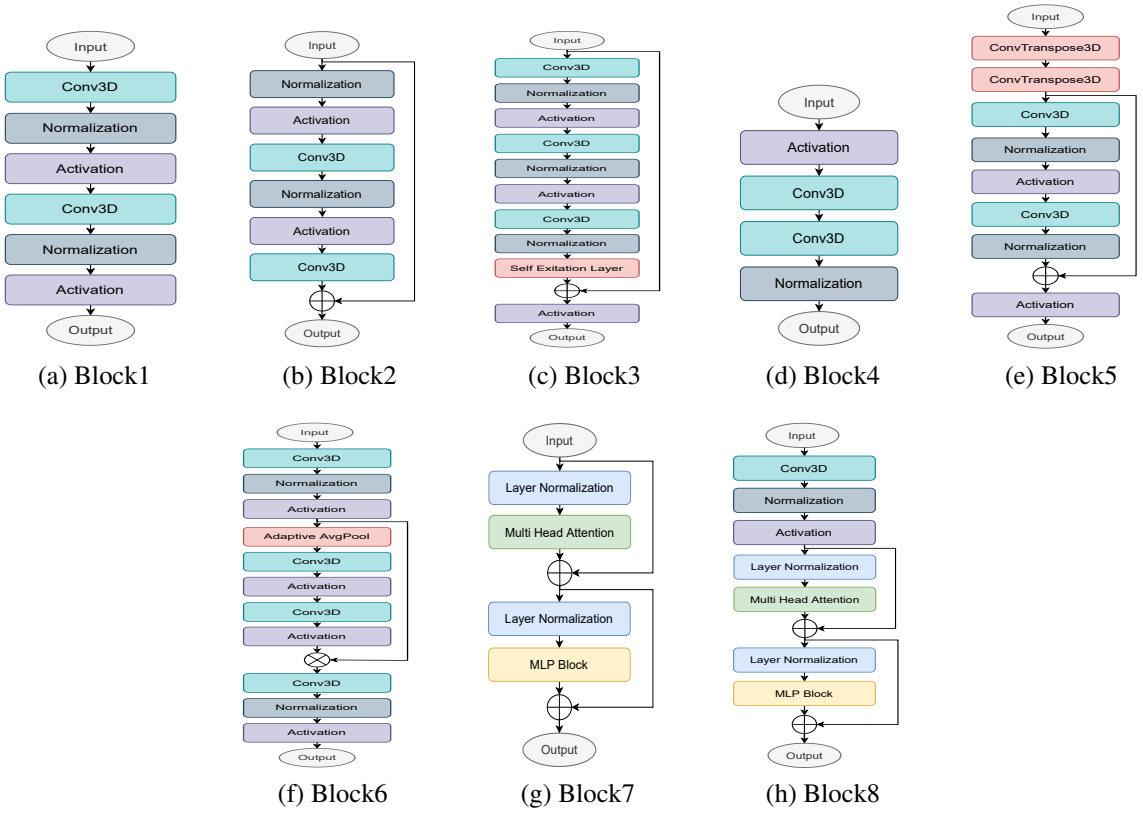
6.2.1 Search space and Encoding

Search space: Segmentation involves the precise delineation and identification of specific structures or regions of interest within an image, such as tumor boundaries. The utilization of this segmentation knowledge in the detection process enables the identification and localization of abnormalities, such as tumors, within the segmented regions. The performance of the detection task can be enhanced by accurately segmenting the relevant structures, allowing detection algorithms to focus on the areas of interest. To extend the segmentation work, the top 5 well-performed blocks (i.e., block1, block3, block5, block6, block8) from Chapter 5 were selected, and an additional three blocks, including EffiecientNet block, Transformer block, and Convolutional Neural Networks Meet Vision Transformer blocks, were added to the search space. Furthermore, the less effective Adadelta optimizer is replaced with AdamW [183], and the DiceFocalLoss is replaced with the DiceCE loss [173]

in the search space. These modifications were implemented to improve the overall performance and effectiveness of the detection process in the evolution. The parameters and their corresponding range in the search space of the proposed method are described in Table 6.1.

Table 6.1: Proposed search space.

Parameter	Range
Block IDs	[Block1-Block8]
Activations	[MEMSWISH, ReLU, PReLU, Leaky ReLU]
Normalization	[BATCH, INSTANCE, Group, None]
Upsampling	[ConvTranspose3D, Upsample3D]
Loss functions	[DiceLoss, DiceCELoss, FocalLoss, TverskyLoss]
Optimizers	[Adam, AdamW, Adamax, SGD]



(a) Block1 (b) Block2 (c) Block3 (d) Block4 (e) Block5

(f) Block6 (g) Block7 (h) Block8

Figure 6.2: Structure of the 8 Blocks in the search space.

Encoding: The proposed approach incorporates a 3D encoder-decoder CNN network with a unique block structure, setting it apart from previous methods that utilize fixed block structure. The encoder-decoder network used in the proposed method consists of a total of seven blocks, each distinguished by a specific block ID, activation function, and nor-

malization layer. The encoding process for each block requires 7 bits, with 3 bits for the block ID, 2 bits for the activation function, and 2 bits for the normalization layer. Consequently, a total of 49 bits are allocated to represent all seven blocks within the network. In addition to the block encoding, 3 bits are used to select the up-sample layers for three stages of the network. Furthermore, 2 bits each are used for specifying the loss function and optimizer in the learner. The encoding of a learner, depicted in Figure 6.3, consists of a length of 56 bits. Within the defined search space, the block ID, activation function, normalization layer, and up-sample parameters are utilized for constructing the backbone of the 3D encoder-decoder CNN network. Once the network structure is formed, it is trained using the specified optimizer and loss function parameters to optimize its performance and achieve accurate results. By encoding and optimizing these network components, the proposed approach aims to find the most effective configuration for the 3D encoder-decoder CNN network within the search space.

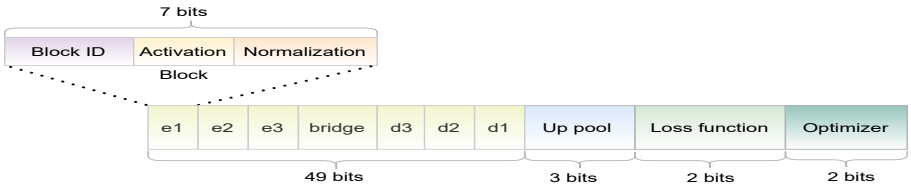


Figure 6.3: Encoding of a Learner.

6.2.2 3DMOEA-Net

Figure 6.4 illustrates the flow diagram of the proposed 3DMOEA-Net method. The proposed method's goal is to identify the optimal blocks and hyperparameters for constructing efficient 3D detection networks. The proposed method employs the MO-ITLBO algorithm as a search strategy to optimize the multi-objective NAS for brain tumor detection. The MO-ITLBO algorithm incorporates non-dominated sorting and crowding distance computation in the selection process of the ITLBO algorithm. These techniques are used to select superior solutions from the population while maintaining diversity among the chosen solutions. By employing these strategies, the method ensures optimal detection results while effectively reducing loss and computational complexity. By considering the search space

parameters as decision variables, the MO-ITLBO algorithm aims to minimize two metrics: the Dice loss (Equation 6.8) and the model parameters, which serve as the fitness functions. Algorithm 6.5 describes the MO-ITLBO algorithm.

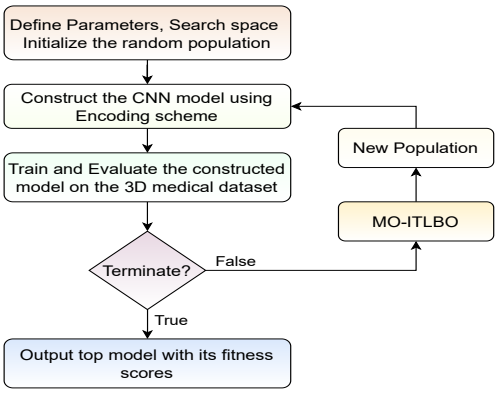


Figure 6.4: The flow diagram of 3DMOEA-Net based on MO-ITLBO.

In each iteration t, the population X^t undergoes the teacher phase. During this phase, a new population $\bar{X}^t$ is derived according to the procedure outlined in Section 6.1.2.4. The fitness scores $f(\bar{X}^t)$ of the new population $\bar{X}^t$ are then evaluated using Algorithm 6.6. Subsequently, the population X^t is combined with the teacher phase population $\bar{X}^t$, resulting in $X^t \cup \bar{X}^t$. From the combined population $X^t \cup \bar{X}^t$, the Pareto front population is obtained, which is denoted by (Y^t) for iteration t, with a size equal to the population size N_p, are selected using non-dominated sorting and crowding distance computation [lines 2 to 5].

The resulting population Y^t is subsequently transferred to the learner phase, where a new population $\bar{Y}^t$ is obtained using the procedure outlined in Section 6.1.2.5. Next, the population Y^t is combined with the learner phase population $\bar{Y}^t$, leading to $\bar{Y}^t \cup Y^t$. From the combined population $\bar{Y}^t \cup Y^t$, the Pareto front solutions of the population X for the next iteration $t+1$, with a size equal to the population size N_p, are chosen using non-dominated sorting and crowding distance computation [lines 6 to 9]. The resulting population X^{t+1} is utilized as the population for the subsequent generation. The algorithm is runs for T_g generations and designed for a termination criteria Tc, which terminate if the top three learners remain unchanged for three consecutive generations [lines 10, 11].

Algorithm 6.5 The pseudocode of MO-ITLBO.

Input: Population X, Number of generations T_g, Population size N_p

Output: Population X, Fitness scores $f(X)$

1: **for** $t = 0, 1, \ldots, T_g - 1$ **do**
2: **//Teacher Phase**
3: $\bar{X}$ = Generate a population from population X using teacher phase (from Section 6.1.2.4)
4: $f(\bar{X})$ = Calculate the fitness scores of population $\bar{X}$ using Algorithm 6.6
5: Y = Select the top N_p pareto front learners from $(X \cup \bar{X})$ (from Section 6.1.3)
6: **//Learner Phase**
7: $\bar{Y}$ = Generate a population from population Y using learner phase (from Section 6.1.2.5)
8: $f(\bar{Y})$ = Calculate the fitness scores of population $\bar{Y}$ using Algorithm 6.6
9: X = Select the top N_p pareto front learners from $(Y \cup \bar{Y})$ (from Section 6.1.3)
10: **if** T_g is **True** or T_c is **True then**
11: **break**
12: **end if**
13: **end for**
14: **return** Fitness scores $f(X)$

To prevent duplicate training of models, the proposed method incorporates a cache mechanism [179] during fitness evaluation. Upon evaluating a model, each learner X_i along with its fitness values $f(X_i)$ are stored in a cache denoted as Gc. Prior to initiating the training process, a check is performed to determine if the learner X_i exists within the cache Gc. If it is found, the fitness values $f(X_i)$ are retrieved from Gc, thereby avoiding redundant training [lines 2, 3]. In case the learner is not found in the cache, the model undergoes training for a specified number of epochs (M), with a value of 40 epochs [lines 5 to 25]. Additionally, an Early Stop Criterion denoted as ESC is adopted, which terminates the training process if there is no improvement in the Dice score (Equation 6.7) within 15 epochs, thereby saving training time [lines 12, 13]. The fitness evaluation of a population is described in Algorithm 6.6.

Algorithm 6.6 The pseudocode for fitness evaluation of a population.

Input: Population X, Cache Gc, Train D_{train} and Test D_{test} datasets

Output: Fitness scores $f(X)$ of a Population X

for each learner X_i **in** X **do**
  if X_i **in** Gc **then**
    $f(X_i)$ = query the Dice loss, parameters of X_i from Gc
  else
    Construct a 3D model by encoding a learner as described in Section 6.2.1
    //Training
    Set f_{best} = 0
    Set $epoch = 1$
    while $epoch \leq M$ **do**
      Train the model on train (D_{train}) dataset
      $Dice$ = Calculate the Dice score (Equation 6.7) on test D_{test} dataset
      if ESC is **True then**
        break
      end if
      if $f_{best} < Dice$ **then**
        $f_{best} = Dice$
        Save the weights of current $epoch$
      end if
    end while
    //Testing
    Load the saved weights to model
    $diceLoss$ = Calculate the Dice loss (Equation 6.8) on test D_{test} dataset
    $params$ = Calculate number of parameters of a model
    Set $f(X_i) = diceLoss, params$
    Add the learner X_i with fitness scores $f(X_i)$ into the Cache Gc
  end if
end for

28: **return** $f(X)$

6.3 Experiments and Analysis

In this section, the datasets used for evaluation, implementation details, and the experimental results of the proposed method are presented. Additionally, a comparison with state-of-the-art models is included, followed by an analysis to provide further insights.

6.3.1 Datasets

The evaluation of the proposed 3DMOEA-Net is conducted on the BraTS2021 and Brain-Tumor datasets. In order to focus on the valid body area, any blank labels present in the images and labels are removed. Random sampling is performed on images with [96, 96, 96] volume size, and voxel intensities are normalized to the [0, 1] range. To mitigate the limited size of the dataset, augmentation techniques like rotation and vertical/horizontal flips [178] are applied to the training data.

6.3.2 Implementation details

The metaheuristic algorithm is initiated with an initial population size N_p of 20 random learners, number of generations T_g is 20. The top 10 learners with the highest fitness scores were selected for the next generation, resulting in a population size N_p of 10. The proposed 3DMOEA-Net method is implemented using MONAI v1.1, PyTorch v1.9.0, and Python 3.8. The training process utilized a batch size of 4 and a learning rate of 0.0001.The experiments were carried out on a system equipped with an Intel 2.2 GHz Xeon® Processor, 16 GB of memory, and a Quadro P5000 graphics card.

Evaluation Metrics: During the experiments, the detection performance is evaluated using the 95% Hausdorff Distance (HD) (Equation 6.6) and the Dice score (Equation 6.7). The 95% HD calculates the distance between the point sets of the prediction and ground truth surfaces, specifically using the 95th percentile. The prediction and ground truth volumes are denoted as P and G, respectively, while the point sets of the prediction and ground truth surfaces are represented as $\bar{P}$ and $\bar{G}$, respectively.

$$95\%HD = \max\left(\max_{\rho\in\bar{P}}\min_{\delta\in\bar{G}}||\rho-\delta||, \max_{\delta\in\bar{G}}\min_{\rho\in\bar{P}}||\delta-\rho||\right) \tag{6.6}$$

$$Dicescore = \frac{2\sum_{i=1}^{I} G_i P_i}{\sum_{i=1}^{I} P_i + \sum_{i=1}^{I} G_i} \tag{6.7}$$

$$DiceLoss = 1 - \frac{2\sum_{i=1}^{I} G_i P_i}{\sum_{i=1}^{I} P_i + \sum_{i=1}^{I} G_i} \tag{6.8}$$

6.3.3 Experiments

To evaluate the performance, a comparison is conducted between the top model discovered by the proposed framework and several state-of-the-art models, such as 3D U-Net [59], V-net [60], UNETR [177], SegResNet [69], and SwinUnetr [72]. Furthermore, to ensure a fair comparison, all models are trained in the same environment.

6.3.3.1 Experimental results on BraTS2021 dataset

The BraTS2021 dataset from the BraTS challenge [184] consists of four types of MRI scans, namely T1-weighted (T1), T2-weighted (T2), T1-enhanced contrast (T1-ce), and T2 fluid-attenuated inversion recovery (Flair). The dataset includes labels for background, GB-enhancing tumors, peritumoral edema, and necrotic and non-enhancing tumors. The labels are used for segmenting the enhanced tumor (ET), tumor core (TC), and whole tumor (WT) regions. The dataset is divided into two sets: the training set (D_{train}) comprising 1001 volumes, and the test set (D_{test}) comprising 250 volumes.

The experimental results are illustrated in Table 6.2, which compares the proposed 3DMOEA-Net model to other existing models. Figure 6.5 displays the visual results of the proposed 3DMOEA-Net model, along with existing models, on the BraTS2021 dataset. Notably, the proposed model outperforms the existing models in both Dice score and HD metrics for TC, WT, and ET on the BraTS2021 dataset.

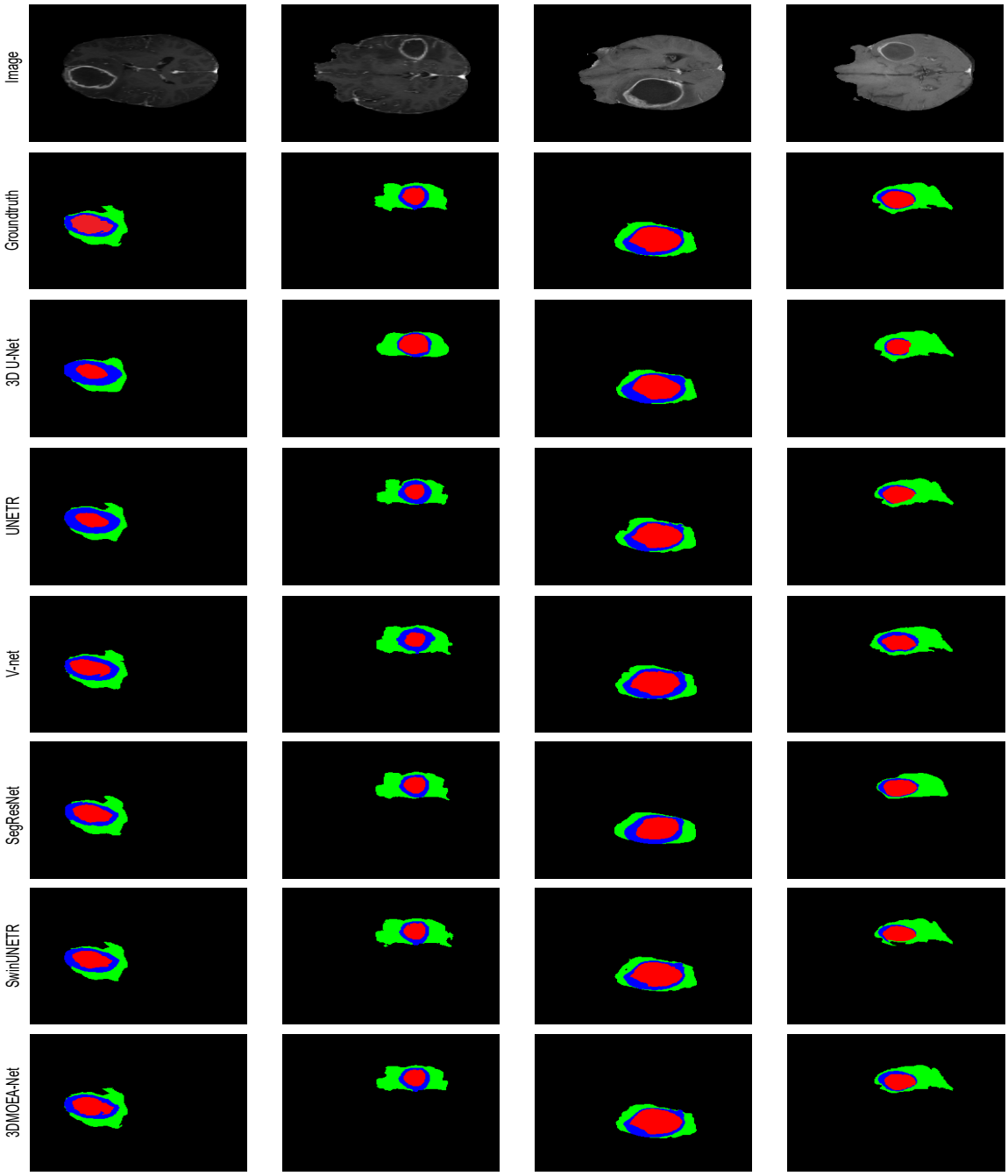


Figure 6.5: Visual results of BraTS2021 dataset (Whole tumor (Green), Enhanced Tumor (Blue), Tumor core (Red)).

6.3.3.2 Experimental results on BrainTumor dataset

The BrainTumor dataset, provided by the Decathlon challenge [145], consists of 484 MRI volumes along with corresponding label volumes. The dataset is partitioned into two sets:

Table 6.2: Comparison with existing models on BraTS2021 dataset.

Models	Dice score (%)				95% HD (mm)			
	TC	WT	ET	Avg	TC	WT	ET	Avg
3D U-Net	0.6761	0.7501	0.6969	0.7077	15.00	20.81	14.18	16.66
UNETR	0.7054	0.7811	0.7674	0.7513	14.05	18.28	13.17	15.16
V-net	0.7504	0.7986	0.7857	0.7782	13.61	17.06	13.02	14.56
SegResNet	0.7618	0.7971	0.7750	0.7779	12.60	14.66	11.95	13.06
SwinUNETR	0.8024	0.8270	0.8100	0.8131	11.67	13.40	10.27	11.78
3DMOEA-Net	**0.8048**	**0.8319**	**0.8136**	**0.8167**	**10.41**	**11.92**	**9.07**	**10.46**

Note: The best values are shown in bold.

the training set (D_{train}) consisting of 363 volumes, and the test set (D_{test}) consisting of 121 volumes. Table 6.3 shows that the proposed 3DMOEA-Net model outperformed existing models on the BrainTumor dataset. The superior performance of the proposed model in accurately extracting tumor from 3D images is evident in Figure 6.6, where the proposed model's segmented images exhibit higher quality compared to other models. These compelling qualitative and quantitative outcomes show the superiority of the proposed 3DMOEA-Net model over existing detection models.

Table 6.3: Comparison with existing models on BrainTumor dataset.

Models	Dice score (%)				95% HD (mm)			
	TC	WT	ET	Avg	TC	WT	ET	Avg
3D U-Net	0.6331	0.7643	0.6263	0.6745	20.78	22.38	21.19	21.45
UNETR	0.6512	0.7970	0.6655	0.7045	19.63	20.61	21.64	20.62
V-net	0.7655	0.8002	0.7712	0.7789	15.23	17.91	18.45	17.19
SegResNet	0.7897	0.8081	0.7756	0.7911	13.87	16.23	16.56	15.55
SwinUNETR	0.8085	0.8197	0.7847	0.8043	12.48	14.49	13.47	13.48
3DMOEA-Net	**0.8267**	**0.8351**	**0.7949**	**0.8189**	**10.86**	**11.61**	**13.20**	**11.89**

Note: The best values are shown in bold.

6.3.4 Experimental analysis

The evolutionary trajectory of the average Dice scores for both brain tumor datasets is illustrated in Figure 6.7. The plot demonstrates the progression of Dice scores over successive generations during the search process. As the generations advance, there is a gradual improvement in the Dice scores, indicating an enhancement in the detection performance. The trajectory eventually converges, suggesting that the algorithm has reached a stable state where further improvement in the Dice scores becomes minimal. This convergence indi-

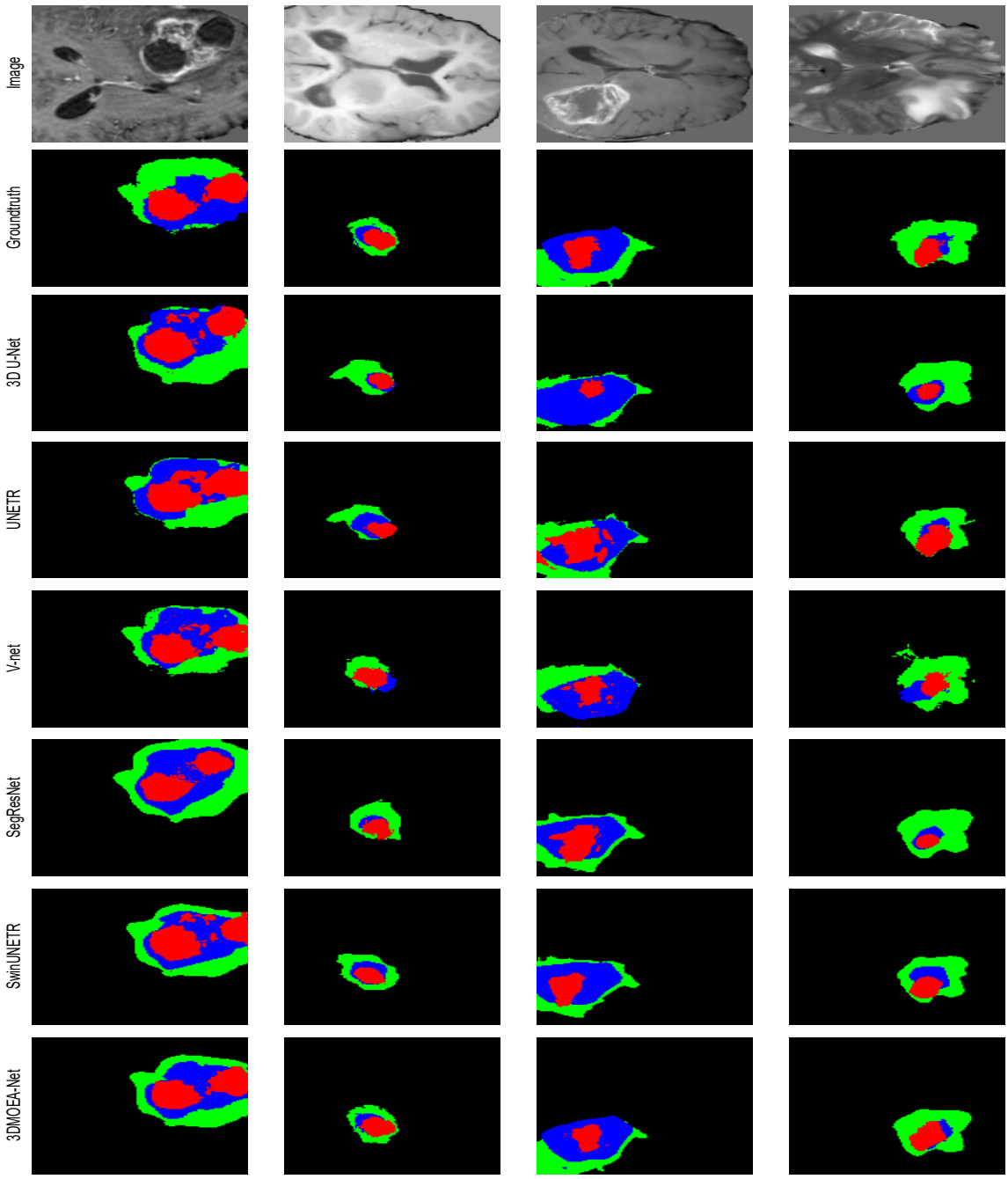


Figure 6.6: Visual results of BrainTumor dataset (Whole tumor (Green), Enhanced Tumor (Blue), Tumor core (Red)).

cates that the proposed method effectively optimizes the detection performance through the search process.

The effectiveness of the MO-ITLBO algorithm in comparison to other metaheuristic

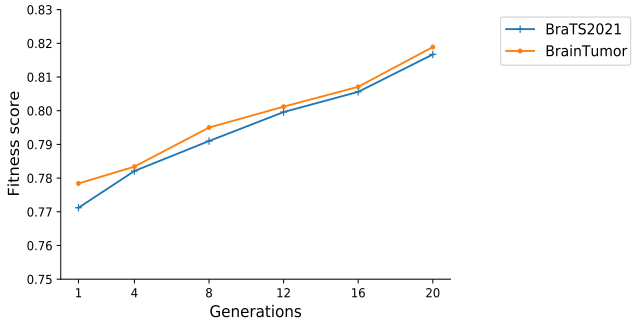


Figure 6.7: Generation wise Dice scores.

optimization algorithms, such as GOA [156], CSA [157], BTLBO [179], and MPA [158], is assessed. The significance of the findings is determined through the Wilcoxon rank-sum test [159], a non-parametric statistical test. Each algorithm is executed for 30 generations, and each model underwent 30 epochs during the training process. As shown in Table 6.4, the MO-ITLBO algorithm exhibited superior mean, standard deviation, and Dice scores compared to the other algorithms. Additionally, Table 6.5 demonstrated that the p-values of the Wilcoxon rank-sum test were below 0.05, indicating the statistical significance of the results. These experimental outcomes strongly support the effectiveness of the proposed method in achieving optimal results for brain tumor detection tasks when employing the MO-ITLBO algorithm. The improved teaching and learning phases in the MO-ITLBO algorithm contribute to accelerated convergence rates. Furthermore, the incorporation of non-dominated sorting enables the algorithm to handle problems with multiple objectives, while the calculation of the crowding distance ensures solution diversity is maintained within a single run, resulting in more efficient search processes. These features establish the MO-ITLBO algorithm as a superior choice when compared to other algorithms in 3D brain tumor detection.

Table 6.4: Mean, standard deviation and max dice score values obtained from 30 generations.

Algorithm	Mean	Standard deviation	Max Dice score
MO-BTLBO	0.7702645483333334	0.05831463716580559	0.8123
MO-MPA	0.7507568571428557	0.11534369098136387	0.8026
MO-CSA	0.7647825066666666	0.04775903374379986	0.8081
MO-GOA	0.7595219220000001	0.10829973361561454	0.8109
MO-ITLBO	0.7847825066666666	0.02262811794524536	0.8167

Table 6.5: Wilcoxon rank-sum test p-values.

Algorithm	p-value
MO-ITLBO vs MO-BTLBO	1.907348632e-06
MO-ITLBO vs MO-CSA	1.715625643e-06
MO-ITLBO vs MO-MPA	1.001953125e-05
MO-ITLBO vs MO-GOA	1.103515625e-05

The proposed method aims to optimize the construction and training of the encoder-decoder network by identifying the optimal blocks and hyperparameters. Figure 6.8 provides valuable insights into the frequency distribution of the blocks present in the top architectures obtained through the search process for both datasets. It is noteworthy that block IDs 1, 5, and 7 demonstrate significantly higher occurrence rates compared to other blocks. This observation suggests that these particular blocks play a crucial role in achieving improved detection performance, as they are repeatedly selected and incorporated into the top-performing architectures. This information can guide future studies in focusing on the design and optimization of these specific blocks to further enhance the performance of the encoder-decoder network.

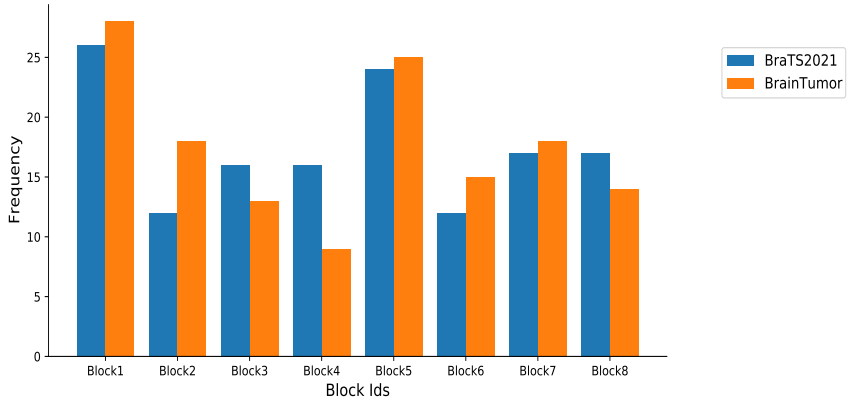


Figure 6.8: Frequency of blocks.

The selection of activation functions and normalization layers is pivotal in ensuring network convergence and efficiency. Figures 6.9, 6.10 provide detailed insights into the distribution of activation functions and normalization layers obtained on top architectures during the search process. Notably, the PReLU and MEMSWISH activation functions, along with Batch and Instance normalizations, were selected more frequently compared to other options. This preference suggests that these specific activation functions and normal-

ization layers demonstrate superior performance in achieving desirable network behaviour, as they were repeatedly favoured by the optimization algorithm.

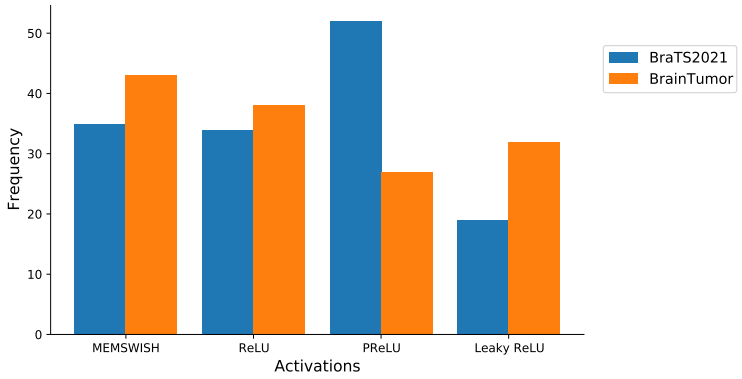


Figure 6.9: Frequency of activation functions.

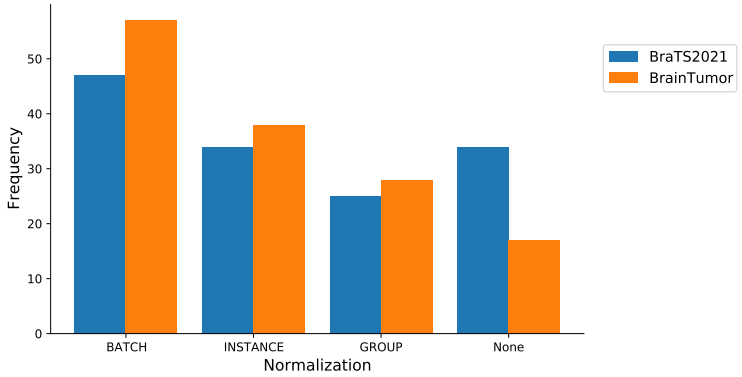


Figure 6.10: Frequency of normalization layers.

To enhance the performance of encoder-decoder networks, the proposed method integrates popularly used pooling layers, specifically the upsampling3D and 3D transposed layers, in the up-sampling part of encoder-decoder architecture. Figure 6.11 provides valuable insights into the distribution and occurrence of these pooling layers within the top architectures generated during the search process. Notably, the 3D transposed convolution layer is found to be more frequently utilized compared to the upsampling3D layer in the up-sampling section. This observation highlights the preference for the 3D transposed convolution layer due to its effectiveness and suitability for the task of enhancing the resolution and preserving spatial information during the up-sampling process.

In addition to identifying optimal blocks and architecture components, the proposed

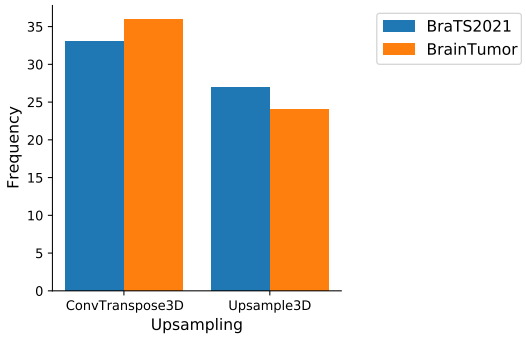


Figure 6.11: Frequency of up-sampling layers.

method also focuses on determining the most suitable training parameters, including the optimizer and loss function. The analysis of Figures 6.12, 6.13 offers valuable insights into the distribution and occurrence of different optimizers and loss functions in the top architectures generated through the search process. It is noteworthy that the SGD optimizer and Tversky loss function were consistently chosen more frequently than other parameters, indicating their effectiveness in brain tumor detection. The Tversky loss function excels in handling class imbalance and capturing intricate details in detection tasks, while the SGD optimizer is widely acknowledged for its simplicity and robustness. The prominence of these choices among the top architectures demonstrates their potential for improving detection performance and highlights their importance in the task of brain tumor detection.

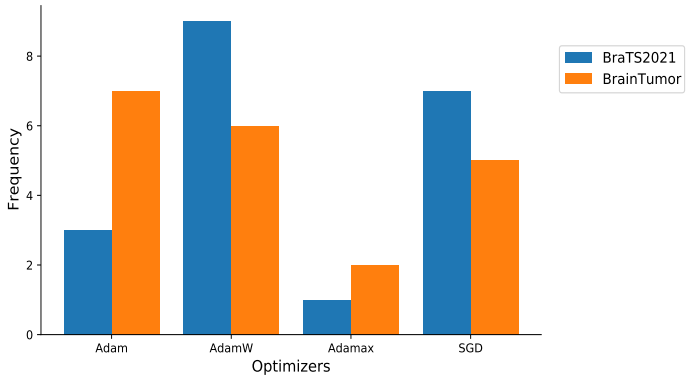


Figure 6.12: Frequency of optimizers.

Table 6.6 displays a comparison of the size and parameters of the discovered architecture with other models, indicating that it has fewer parameters and a smaller size compared to existing models. Furthermore, the proposed method achieves significant improvement in

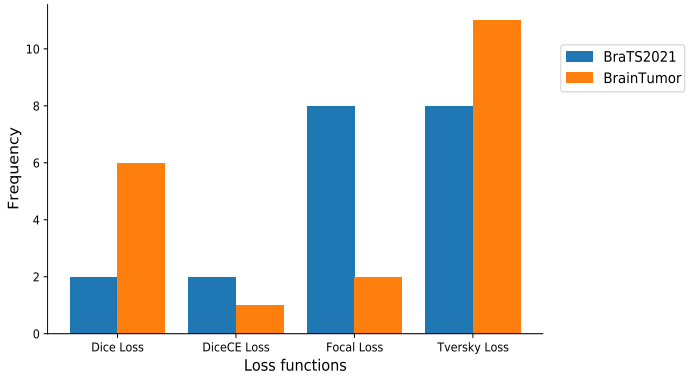


Figure 6.13: Frequency of loss functions.

both Hausdorff distances and Dice scores over traditional CNN models like V-Net, 3D U-Net, and SegResNet, as well as transformer-based models like UNETR and SwinUNETR.

Table 6.6: Comparison of existing models size and parameters.

Model	Size	Parameters
V-net	3410.15MB	45.60M
SegResNet	3145.71MB	24.70M
SwinUNETR	6061.63MB	62.19M
3DMOEA-Net	2785.69MB	17.39M

6.4 Summary

In this chapter, a novel multi-objective NAS based framework named 3DMOEA-Net is proposed for discovering optimal 3D models to detect brain tumors in 3D MRI volumes. The proposed method introduces a diverse search space comprising different blocks such as EffiecientNet block, Transformer block, and Convolutional Neural Networks Meet Vision Transformer blocks, as well as training parameters. To explore this search space effectively, a multi-objective ITLBO algorithm is employed, which identifies non-dominant solutions on the Pareto front by optimizing both the number of parameters and Dice loss. Experimental outcomes demonstrate the superiority of the proposed method in brain tumor detection, demonstrating enhanced performance while minimizing parameters compared to state-of-the-art methods. Notably, the utilization of different block structures in encoder-decoder networks, as opposed to fixed block structures like U-Net, significantly enhances detection

performance and shows the significance of the diverse network architectures in constructing automated CNN models. In the next chapter, the conclusion and future scope of the thesis are presented.

Chapter 7

Conclusion and Future Scope

This chapter presents the summary of the contributions of this thesis, the conclusion of each objective and the future scope of research for further direction of this thesis is presented.

7.1 Conclusions

This thesis demonstrates the efficacy of metaheuristic optimization algorithms in developing various deep-learning models for medical image analysis. The utilization of metaheuristic approaches allows for comprehensive exploration, optimization, and customization of architectures, resulting in improved performance and potential applications across various medical imaging tasks such as denoising, segmentation, and detection.

In chapter 3, a metaheuristic block-based deep neural network named DEvoNET is designed for medical image denoising. By leveraging the power of the differential evolution algorithm, DEvoNET autonomously identifies the optimal network structure and training parameters. The incorporation of DE algorithm allows for efficient exploration of the search space and the discovery of high-performing architectural components. Additionally, a transfer learning approach is employed to accelerate the evolutionary process by leveraging knowledge from previously evolved architectures. The experimental results showcased the superior performance of DEvoNET compared to existing denoising models, highlighting its potential in automatically constructing networks for medical image denoising tasks.

In chapter 4, a modified U-shaped (encoder-decoder) model based on the metaheuristic

algorithm called BTU-Net is proposed for 2D medical image segmentation with denoising (from chapter 3) as a preprocessing step. The proposed method utilized the metaheuristic BTLBO algorithm to optimize each block within the network, resulting in enhanced segmentation performance. The attention mechanism incorporated in BTU-Net enabled precise and accurate segmentation of complex structures. By defining a condensed and flexible search space, the method efficiently explored different architectural components. Additionally, the use of a cache-enabled BTLBO algorithm accelerated the fitness evaluation process, avoiding redundant training. The proposed method achieved improved segmentation performance by optimizing each block within the U-shaped network. The experimental results obtained from various medical image datasets emphasized the importance of denoising as a preprocessing step in enhancing medical image segmentation performance. These findings indicate the potential clinical applications of the BTU-Net model in medical image segmentation tasks.

In chapter 5, a metaheuristic NAS based framework named CS3DEA-Net is developed for 3D medical image segmentation. The proposed framework employed an encoder-decoder architecture and incorporated a comprehensive search space containing various block structures and training parameters. By utilizing the CSA algorithm, CS3DEA-Net achieved enhanced performance in 3D medical image segmentation. The algorithm efficiently explored the search space, optimizing the network blocks within the encoder-decoder architecture and identifying the optimal training parameters for model training. Extensive evaluations on diverse 3D medical image segmentation datasets demonstrated the superior performance of CS3DEA-Net compared to state-of-the-art models. These findings showcase the exceptional potential of CS3DEA-Net for clinical applications in the field of medical image segmentation, indicating its significance in advancing the accuracy and reliability of segmentation tasks in healthcare settings.

In chapter 6, a multi-objective NAS based framework named 3DMOEA-Net is proposed for discovering optimal 3D models to detect brain tumors in 3D MRI volumes. The proposed method developed a diverse search space, combining top-performing blocks from chapter 5 segmentation work with additional blocks such as EffiecientNet block, Transformer block, and Convolutional Neural Networks Meet Vision Transformer blocks, along

with training parameters. To effectively explore this search space, a multi-objective ITLBO algorithm is employed, optimizing both the number of parameters and dice loss to identify non-dominant solutions on the pareto front. Notably, the utilization of different block structures in encoder-decoder networks instead of fixed block structures like U-Net significantly improved the detection performance and highlighted the importance of architectural diversity in constructing automated CNN models. Experimental results demonstrated the superiority of the proposed 3DMOEA-Net method in brain tumor detection, achieving enhanced performance while minimizing parameters compared to state-of-the-art methods. These findings emphasize the potential of multi-objective NAS frameworks in advancing automated brain tumor detection and highlight the benefits of incorporating diverse architectural components in the design of CNN models.

7.2 Future Scope

Future scope for the proposed metaheuristic-based deep learning models in medical image analysis can focus on several directions:

- As medical imaging datasets continue to grow in size and complexity, future research can explore strategies to improve the scalability and efficiency of the proposed models. This can involve investigating parallel computing techniques, distributed training, and model compression methods to reduce memory and computational requirements without compromising performance.

- There is a scope to expand the search space in medical image analysis by incorporating novel architectural components or the exploration of different combinations of existing blocks and considering other parameters. This would enable the discovery of more diverse and effective network architectures that can further enhance the performance of medical image analysis tasks.

- Collaborating with medical experts and conducting large-scale studies on diverse datasets from different healthcare institutions can provide valuable insights into the models' performance and their impact on clinical decision-making.

- Real-time or dynamic medical imaging, such as video sequences or continuous monitoring data, pose unique challenges due to their temporal nature. Future work can extend the proposed models to handle such data by incorporating temporal information and developing specialized architectures that can effectively process and analyze dynamic medical imaging data.

- Deep learning models are often considered black boxes, making it challenging to interpret and explain their decisions. Future work can focus on developing Explainable AI techniques to build trust and promote the adoption of these models by enhancing their interpretability and explainability. This can involve integrating visual explanations or generating saliency maps to highlight the regions of interest that contribute to the model's predictions.

Bibliography

[1] Marden E Alder, S Thomas Deahl, and Stephen R Matteson. Clinical usefulness of two-dimensional reformatted and three-dimensionally rendered computerized tomographic images: literature review and a survey of surgeons' opinions. *Journal of oral and maxillofacial surgery*, 53(4):375–386, 1995.

[2] Georgios Sakas. Trends in medical imaging: from 2d to 3d. *Computers & Graphics*, 26(4):577–587, 2002.

[3] Zeynettin Akkus, Alfiia Galimzianova, Assaf Hoogi, Daniel L Rubin, and Bradley J Erickson. Deep learning for brain mri segmentation: state of the art and future directions. *Journal of digital imaging*, 30:449–459, 2017.

[4] El-Sayed A El-Dahshan, Heba M Mohsen, Kenneth Revett, and Abdel-Badeeh M Salem. Computer-aided diagnosis of human brain tumor through mri: A survey and a new algorithm. *Expert systems with Applications*, 41(11):5526–5545, 2014.

[5] TS Deepthi Murthy and G Sadashivappa. Brain tumor segmentation using thresholding, morphological operations and extraction of features of tumor. In *2014 international conference on advances in electronics computers and communications*, pages 1–6. IEEE, 2014.

[6] Muhammad Kaleem, M Sanaullah, M Ayyaz Hussain, M Arfan Jaffar, and Tae-Sun Choi. Segmentation of brain tumor tissue using marker controlled watershed transform method. In *Emerging Trends and Applications in Information Communication Technologies: Second International Multi Topic Conference, IMTIC 2012, Jamshoro, Pakistan, March 28-30, 2012. Proceedings 2*, pages 222–227. Springer, 2012.

[7] KV Ahammed Muneer and K Paul Joseph. Performance analysis of combined k-mean and fuzzy-c-mean segmentation of mr brain images. In *Computational Vision and Bio Inspired Computing*, pages 830–836. Springer, 2018.

[8] Meghavi Rana and Megha Bhushan. Machine learning and deep learning approach for medical image analysis: diagnosis to detection. *Multimedia Tools and Applications*, 82(17):26731–26769, 2023.

[9] Pin Wang, En Fan, and Peng Wang. Comparative analysis of image classification algorithms based on traditional machine learning and deep learning. *Pattern Recognition Letters*, 141:61–67, 2021.

[10] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[11] Aliasghar Mortazi and Ulas Bagci. Automatically designing cnn architectures for medical image segmentation. In *International Workshop on Machine Learning in Medical Imaging*, pages 98–106. Springer, 2018.

[12] Dario Floreano, Peter Dürr, and Claudio Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary intelligence*, 1:47–62, 2008.

[13] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Kay Chen Tan. A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, 2021.

[14] Dong Yang, Holger Roth, Ziyue Xu, Fausto Milletari, Ling Zhang, and Daguang Xu. Searching learning strategy with reinforcement learning for 3d medical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 3–11. Springer, 2019.

[15] Yu Weng, Tianbao Zhou, Yujie Li, and Xiaoyu Qiu. Nas-unet: Neural architecture search for medical image segmentation. *IEEE Access*, 7:44247–44257, 2019.

[16] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[17] José V Manjón and Pierrick Coupe. Mri denoising using deep learning. In *Patch-Based Techniques in Medical Imaging: 4th International Workshop, Patch-MI 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*, pages 12–19. Springer, 2018.

[18] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

[19] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.

[20] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[22] R Venkata Rao, Vimal J Savsani, and DP Vakharia. Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. *Information sciences*, 183(1):1–15, 2012.

[23] Ahmet Uluslu. Chameleon swarm algorithm assisted optimization of u-slot patch antenna for quad-band applications. *IEEE Access*, 10:74152–74163, 2022.

[24] Vivek K Patel and Vimal J Savsani. A multi-objective improved teaching–learning based optimization algorithm (mo-itlbo). *Information Sciences*, 357:182–200, 2016.

[25] Thanasis Loupas, WN McDicken, and Paul L Allan. An adaptive weighted median filter for speckle suppression in medical ultrasonic images. *IEEE transactions on Circuits and Systems*, 36(1):129–135, 1989.

[26] Amit Garg and Vineet Khandelwal. Despeckling of medical ultrasound images using fast bilateral filter and neighshrinksure filter in wavelet domain. In *Advances in Signal Processing and Communication: Select Proceedings of ICSC 2018*, pages 271–280. Springer, 2019.

[27] Hajer Jomaa, Rostom Mabrouk, Nawres Khlifa, and Frédéric Morain-Nicolier. Denoising of dynamic pet images using a multi-scale transform and non-local means filter. *Biomedical Signal Processing and Control*, 41:69–80, 2018.

[28] Simarjot Kaur Randhawa, Ramesh Kumar Sunkaria, and Emjee Puthooran. Despeckling of ultrasound images using novel adaptive wavelet thresholding function. *Multidimensional Systems and Signal Processing*, 30:1545–1561, 2019.

[29] Pierrick Coupé, Pierre Hellier, Sylvain Prima, Charles Kervrann, and Christian Barillot. 3d wavelet subbands mixing for image denoising. *International Journal of Biomedical Imaging*, 2008, 2008.

[30] Minh N Do and Martin Vetterli. The contourlet transform: an efficient directional multiresolution image representation. *IEEE Transactions on image processing*, 14(12):2091–2106, 2005.

[31] Ami George, S Logesh Kumar, K Manikandan, C Vijayalakshmi, and V Renuga. Medical image denoising using wavelet transform. *image*, 5:6, 2016.

[32] Forest Agostinelli, Michael R Anderson, and Honglak Lee. Adaptive multi-column deep neural networks with application to robust image denoising. *Advances in neural information processing systems*, 26, 2013.

[33] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

[34] Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. In *2016 IEEE 16th international conference on data mining workshops (ICDMW)*, pages 241–246. IEEE, 2016.

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[36] Qiangqiang Yuan, Qiang Zhang, Jie Li, Huanfeng Shen, and Liangpei Zhang. Hyperspectral image denoising employing a spatial–spectral deep residual convolutional neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 57(2):1205–1218, 2018.

[37] Maryam Gholizadeh-Ansari, Javad Alirezaie, and Paul Babyn. Low-dose ct denoising with dilated residual network. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 5117–5120. IEEE, 2018.

[38] Leonardo Rundo, Andrea Tangherloni, Marco S Nobile, Carmelo Militello, Daniela Besozzi, Giancarlo Mauri, and Paolo Cazzaniga. Medga: a novel evolutionary method for image enhancement in medical imaging systems. *Expert Systems with Applications*, 119:387–399, 2019.

[39] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[40] Mohamed Elhoseny and K Shankar. Optimal bilateral filter and convolutional neural network based denoising method of medical image measurements. *Measurement*, 143:125–135, 2019.

[41] Peng Liu, Mohammad D El Basha, Yangjunyi Li, Yao Xiao, Pina C Sanelli, and Ruogu Fang. Deep evolutionary networks with expedited genetic algorithms for medical image denoising. *Medical image analysis*, 54:306–315, 2019.

[42] Md Sujan, Nashid Alam, Syed Abdullah Noman, and M Jahirul Islam. A segmentation based automated system for brain tumor detection. *International Journal of Computer Applications*, 153(10):41–49, 2016.

[43] N Hema Rajini, T Narmatha, and R Bhavani. Automatic classification of mr brain tumor images using decision tree. In *Proceedings of international conference on electronics*, volume 31, 2012.

[44] Wenyin Zhang, Yong Wu, Bo Yang, Shunbo Hu, Liang Wu, and Sahraoui Dhelim. Overview of multi-modal brain tumor mr image segmentation. In *Healthcare*, volume 9, page 1051. MDPI, 2021.

[45] Asieh Khosravanian, Mohammad Rahmanimanesh, Parviz Keshavarzi, and Saeed Mozaffari. Fast level set method for glioma brain tumor segmentation based on

superpixel fuzzy clustering and lattice boltzmann method. *Computer Methods and Programs in Biomedicine*, 198:105809, 2021.

[46] S Reza and Khan M Iftekharuddin. Multi-fractal texture features for brain tumor and edema segmentation. In *Medical Imaging 2014: Computer-Aided Diagnosis*, volume 9035, pages 11–20. SPIE, 2014.

[47] Frank G Zöllner, Kyrre E Emblem, and Lothar R Schad. Support vector machines in dsc-based glioma imaging: suggestions for optimal characterization. *Magnetic resonance in medicine*, 64(4):1230–1236, 2010.

[48] Xinyu Zhou, Xuanya Li, Kai Hu, Yuan Zhang, Zhineng Chen, and Xieping Gao. Ervnet: An efficient 3d residual neural network for brain tumor segmentation. *Expert Systems with Applications*, 170:114566, 2021.

[49] Bo Wang, Shuang Qiu, and Huiguang He. Dual encoding u-net for retinal vessel segmentation. In *International conference on medical image computing and computer-assisted intervention*, pages 84–92. Springer, 2019.

[50] Gendry Alfonso Francia, Carlos Pedraza, Marco Aceves, and Saul Tovar-Arriaga. Chaining a u-net with a residual u-net for retinal blood vessels segmentation. *IEEE Access*, 8:38493–38500, 2020.

[51] Changlu Guo, Márton Szemenyei, Yugen Yi, Ying Xue, Wei Zhou, and Yangyuan Li. Dense residual network for retinal vessel segmentation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1374–1378. IEEE, 2020.

[52] Liangzhi Li, Manisha Verma, Yuta Nakashima, Hajime Nagahara, and Ryo Kawasaki. Iternet: Retinal image segmentation utilizing structural redundancy in vessel networks. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3656–3665, 2020.

[53] Tahereh Hassanzadeh, Daryl Essam, and Ruhul Sarker. Evou-net: An evolutionary deep fully convolutional neural network for medical image segmentation. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 181–189, 2020.

[54] Yanan Sun, Bing Xue, Mengjie Zhang, and Gary G Yen. Evolving deep convolutional neural networks for image classification. *IEEE Transactions on Evolutionary Computation*, 24(2):394–407, 2019.

[55] Zhun Fan, Jiahong Wei, Guijie Zhu, Jiajie Mo, and Wenji Li. Evolutionary neural architecture search for retinal vessel segmentation. *arXiv preprint arXiv:2001.06678*, 2020.

[56] Bin Wang, Bing Xue, and Mengjie Zhang. Particle swarm optimisation for evolving deep neural networks for image classification by evolving and stacking transferable

blocks. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

[57] Tahereh Hassanzadeh, Daryl Essam, and Ruhul Sarker. An evolutionary denseres deep convolutional neural network for medical image segmentation. *IEEE Access*, 8:212298–212314, 2020.

[58] Jiahong Wei, Guijie Zhu, Zhun Fan, Jinchao Liu, Yibiao Rong, Jiajie Mo, Wenji Li, and Xinjian Chen. Genetic u-net: Automatically designed deep networks for retinal vessel segmentation using a genetic algorithm. *IEEE Transactions on Medical Imaging*, 2021.

[59] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pages 424–432. Springer, 2016.

[60] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. Ieee, 2016.

[61] Patrick Ferdinand Christ, Mohamed Ezzeldin A Elshaer, Florian Ettlinger, Sunil Tatavarty, Marc Bickel, Patrick Bilic, Markus Rempfler, Marco Armbruster, Felix Hofmann, Melvin D'Anastasi, et al. Automatic liver and lesion segmentation in ct using cascaded fully convolutional neural networks and 3d conditional random fields. In *International conference on medical image computing and computer-assisted intervention*, pages 415–423. Springer, 2016.

[62] Hao Chen, Qi Dou, Lequan Yu, and Pheng-Ann Heng. Voxresnet: Deep voxelwise residual networks for volumetric brain segmentation. *arXiv preprint arXiv:1608.05895*, 2016.

[63] Mahsa Shakeri, Stavros Tsogkas, Enzo Ferrante, Sarah Lippe, Samuel Kadoury, Nikos Paragios, and Iasonas Kokkinos. Sub-cortical brain structure segmentation using f-cnn's. In *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pages 269–272. IEEE, 2016.

[64] Lequan Yu, Xin Yang, Hao Chen, Jing Qin, and Pheng Ann Heng. Volumetric convnets with mixed residual connections for automated prostate segmentation from 3d mr images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[65] Kisuk Lee, Jonathan Zung, Peter Li, Viren Jain, and H Sebastian Seung. Superhuman accuracy on the snemi3d connectomics challenge. *arXiv preprint arXiv:1706.00120*, 2017.

[66] Xiaomeng Li, Hao Chen, Xiaojuan Qi, Qi Dou, Chi-Wing Fu, and Pheng-Ann Heng. H-denseunet: hybrid densely connected unet for liver and tumor segmentation from ct volumes. *IEEE transactions on medical imaging*, 37(12):2663–2674, 2018.

[67] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999*, 2018.

[68] Martin Kolařík, Radim Burget, Václav Uher, Kamil Říha, and Malay Kishore Dutta. Optimized high resolution 3d dense-u-net network for brain and spine segmentation. *Applied Sciences*, 9(3):404, 2019.

[69] Andriy Myronenko. 3d mri brain tumor segmentation using autoencoder regularization. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II 4*, pages 311–320. Springer, 2019.

[70] Guanhua Li, Yuting Chen, Wenjie Chen, and Yen-Wei Chen. Jointly learning from mri and ct images for automatic liver segmentation with 3d deep learning. *IEEE Transactions on Medical Imaging*, 40(3):893–903, 2021.

[71] Haoyu Liu, Haoyu Lu, Lequan Yu, and Qi Dou. Ensemble of 3d deep neural networks for automatic liver segmentation. *IEEE Journal of Biomedical and Health Informatics*, 25(3):1073–1083, 2021.

[72] Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, Dong Yang, Holger R Roth, and Daguang Xu. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 7th International Workshop, BrainLes 2021, Held in Conjunction with MICCAI 2021, Virtual Event, September 27, 2021, Revised Selected Papers, Part I*, pages 272–284. Springer, 2022.

[73] Marc Demoustier, Ines Khemir, Quoc Duong Nguyen, Lucien Martin-Gaffé, and Nicolas Boutry. Residual 3d u-net with localization for brain tumor segmentation. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 7th International Workshop, BrainLes 2021, Held in Conjunction with MICCAI 2021, Virtual Event, September 27, 2021, Revised Selected Papers, Part I*, pages 389–399. Springer, 2022.

[74] Tahereh Hassanzadeh, Daryl Essam, and Ruhul Sarker. Evolutionary deep attention convolutional neural networks for 2d and 3d medical image segmentation. *Journal of Digital Imaging*, 34:1387–1404, 2021.

[75] Amrita Kaur, Lakhwinder Kaur, and Ashima Singh. Ga-unet: Unet-based framework for segmentation of 2d and 3d medical images applicable on heterogeneous datasets. *Neural Computing and Applications*, 33(21):14991–15025, 2021.

[76] Zhimin Xu, Si Zuo, Edmund Y Lam, Byoungho Lee, and Ni Chen. Autosegnet: An automated neural network for image segmentation. *IEEE Access*, 8:92452–92461, 2020.

[77] Sungwoong Kim, Ildoo Kim, Sungbin Lim, Woonhyuk Baek, Chiheon Kim, Hyungjoo Cho, Boogeon Yoon, and Taesup Kim. Scalable neural architecture search for 3d medical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 220–228. Springer, 2019.

[78] Ken CL Wong and Mehdi Moradi. Segnas3d: Network architecture search with derivative-free global optimization for 3d image segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part III 22*, pages 393–401. Springer, 2019.

[79] Zhuotun Zhu, Chenxi Liu, Dong Yang, Alan Yuille, and Daguang Xu. V-nas: Neural architecture search for volumetric medical image segmentation. In *2019 International conference on 3d vision (3DV)*, pages 240–248. IEEE, 2019.

[80] Woong Bae, Seungho Lee, Yeha Lee, Beomhee Park, Minki Chung, and Kyu-Hwan Jung. Resource optimized neural architecture search for 3d medical image segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part II 22*, pages 228–236. Springer, 2019.

[81] Fabian Isensee, Paul F Jaeger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2):203–211, 2021.

[82] Maria Baldeon Calisto and Susana K Lai-Yuen. Adaen-net: An ensemble of adaptive 2d–3d fully convolutional networks for medical image segmentation. *Neural Networks*, 126:76–94, 2020.

[83] Tahereh Hassanzadeh, Daryl Essam, and Ruhul Sarker. 2d to 3d evolutionary deep convolutional neural networks for medical image segmentation. *IEEE Transactions on Medical Imaging*, 40(2):712–721, 2020.

[84] Qihang Yu, Dong Yang, Holger Roth, Yutong Bai, Yixiao Zhang, Alan L Yuille, and Daguang Xu. C2fnas: Coarse-to-fine neural architecture search for 3d medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4126–4135, 2020.

[85] Jikuan Qian, Rui Li, Xin Yang, Yuhao Huang, Mingyuan Luo, Zehui Lin, Wenhui Hong, Ruobing Huang, Haining Fan, Dong Ni, et al. Hasa: hybrid architecture search with aggregation strategy for echinococcosis classification and ovary segmentation in ultrasound images. *Expert Systems with Applications*, 202:117242, 2022.

[86] Xiaowen Chu and Xin He. Medpipe: End-to-end joint search of data augmentation policy and neural architecture for 3d medical image classification. 2022.

[87] Maria Baldeon-Calisto and Susana K Lai-Yuen. Adaresu-net: Multiobjective adaptive convolutional neural network for medical image segmentation. *Neurocomputing*, 392:325–340, 2020.

[88] Maria G Baldeon Calisto and Susana K Lai-Yuen. Self-adaptive 2d-3d ensemble of fully convolutional networks for medical image segmentation. In *Medical Imaging 2020: Image Processing*, volume 11313, pages 459–469. SPIE, 2020.

[89] Maria Baldeon Calisto and Susana K Lai-Yuen. Emonas-net: Efficient multiobjective neural architecture search using surrogate-assisted evolutionary algorithm for 3d medical image segmentation. *Artificial Intelligence in Medicine*, 119:102154, 2021.

[90] Yan Zhang, Hanqing Li, and Jiahui Liu. Multi-objective neural architecture search for 3d medical image segmentation. *IEEE Transactions on Medical Imaging*, 40(5):1425–1435, 2021.

[91] Qianli Hu, Zhengxing Ma, Yichong Wang, Fei Peng, Chunming Zheng, Xin Gao, Zhihua Wang, and Xiaodong Yang. Automated design of deep learning methods for biomedical image segmentation using multi-objective evolutionary optimization. *IEEE Transactions on Medical Imaging*, 2021.

[92] Emad A Zanaty, Shaker H El-Sappagh, and Mohammed Elmogy. Multi-objective optimization for the segmentation of tumors in brain mri scans using deep convolutional neural networks. *IEEE Access*, 9:61926–61942, 2021.

[93] Chuanpeng Guo, Yuan Yuan, Xiaohua Wang, and Lei Zhang. Multi-objective evolutionary algorithm-based 3d convolutional neural network design for medical image segmentation. *Computerized Medical Imaging and Graphics*, 89:101882, 2021.

[94] Haofu Zhao, Xiangyu Wang, Jian Yang, Liang Chen, Lei Zhang, and Dong Ni. Multi-objective neural architecture search for 3d medical image segmentation. *IEEE Transactions on Medical Imaging*, 40(9):2628–2638, 2021.

[95] Derviş Karaboğa and Selçuk Ökdem. A simple and global optimization algorithm for engineering problems: differential evolution algorithm. *Turkish Journal of Electrical Engineering and Computer Sciences*, 12(1):53–60, 2004.

[96] Seng Poh Lim and Habibollah Haron. Performance comparison of genetic algorithm, differential evolution and particle swarm optimization towards benchmark functions. In *2013 IEEE Conference on Open Systems (ICOS)*, pages 41–46. IEEE, 2013.

[97] Brian Hegerty, Chih-Cheng Hung, and Kristen Kasprak. A comparative study on differential evolution and genetic algorithms for some combinatorial problems. In *Proceedings of 8th Mexican international conference on artificial intelligence*, volume 9, page 13, 2009.

[98] Xing Xu and Yuanxiang Li. Comparison between particle swarm optimization, differential evolution and multi-parents crossover. In *2007 International Conference on Computational Intelligence and Security (CIS 2007)*, pages 124–127. IEEE, 2007.

[99] Omid Tarkhaneh and Haifeng Shen. An adaptive differential evolution algorithm to optimal multi-level thresholding for mri brain image segmentation. *Expert Systems with Applications*, 138:112820, 2019.

[100] Bin Wang, Yanan Sun, Bing Xue, and Mengjie Zhang. A hybrid differential evolution approach to designing deep convolutional neural networks for image classification. In *Australasian Joint Conference on Artificial Intelligence*, pages 237–250. Springer, 2018.

[101] Manjit Kaur and Dilbag Singh. Multi-modality medical image fusion technique using multi-objective differential evolution based deep neural networks. *Journal of Ambient Intelligence and Humanized Computing*, 12(2):2483–2493, 2021.

[102] Vikas Gupta, Chi Chiu Chan, and Pui Tze Sian. A differential evolution approach to pet image de-noising. In *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4173–4176. IEEE, 2007.

[103] Xiaoyan Xu and Robert D Dony. Differential evolution with powell's direction set method in medical image registration. In *2004 2nd IEEE International Symposium on Biomedical Imaging: Nano to Macro (IEEE Cat No. 04EX821)*, pages 732–735. IEEE, 2004.

[104] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[105] SMA Sharif, Rizwan Ali Naqvi, and Mithun Biswas. Learning medical image denoising with deep dynamic residual attention network. *Mathematics*, 8(12):2192, 2020.

[106] Worku Jifara, Feng Jiang, Seungmin Rho, Maowei Cheng, and Shaohui Liu. Medical image denoising using convolutional neural network: a residual learning approach. *The Journal of Supercomputing*, 75(2):704–718, 2019.

[107] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[108] Fan Jia, Wing Hong Wong, and Tieyong Zeng. Ddunet: Dense dense u-net with applications in image denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 354–364, 2021.

[109] Bumjun Park, Songhyun Yu, and Jechang Jeong. Densely connected hierarchical network for image denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[110] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[111] Tahereh Hassanzadeh, Leonard GC Hamey, and Kevin Ho-Shon. Convolutional neural networks for prostate magnetic resonance image segmentation. *IEEE Access*, 7:36748–36760, 2019.

[112] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[113] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.

[114] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010.

[115] Kang Zhang, Xiaohong Liu, Jun Shen, Zhihuan Li, Ye Sang, Xingwang Wu, Yunfei Zha, Wenhua Liang, Chengdi Wang, Ke Wang, et al. Clinically applicable ai system for accurate diagnosis, quantitative measurements, and prognosis of covid-19 pneumonia using computed tomography. *Cell*, 181(6):1423–1433, 2020.

[116] Joseph Paul Cohen, Paul Morrison, Lan Dao, Karsten Roth, Tim Q Duong, and Marzyeh Ghassemi. Covid-19 image data collection: Prospective predictions are the future. *arXiv preprint arXiv:2006.11988*, 2020.

[117] Msoud Nickparvar. Brain tumor mri dataset, 2021.

[118] Shuo Wang, Chengliang Dai, Yuanhan Mo, Elsa Angelini, Yike Guo, and Wenjia Bai. Automatic brain tumour segmentation and biophysics-guided survival prediction. In *International MICCAI Brainlesion Workshop*, pages 61–72. Springer, 2019.

[119] Awais Mansoor, Ulas Bagci, and Daniel J Mollura. Optimally stabilized pet image denoising using trilateral filtering. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 130–137. Springer, 2014.

[120] Pascal Hannequin and Jacky Mas. Statistical and heuristic image noise extraction (shine): a new method for processing poisson noise in scintigraphic images. *Physics in Medicine & Biology*, 47(24):4329, 2002.

[121] Xiao-Jiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using convolutional auto-encoders with symmetric skip connections. *arXiv preprint arXiv:1606.08921*, 2016.

[122] Karthik Ramasubramanian and Abhishek Singh. Deep learning using keras and tensorflow. In *Machine Learning Using R*, pages 667–688. Springer, 2019.

[123] Richard Dosselmann and Xue Dong Yang. A comprehensive assessment of the structural similarity index. *Signal, Image and Video Processing*, 5(1):81–91, 2011.

[124] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.

[125] Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Toward convolutional blind denoising of real photographs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1712–1722, 2019.

[126] Shengkai Zhuo, Zhi Jin, Wenbin Zou, and Xia Li. Ridnet: Recursive information distillation network for color image denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[127] Lianzheng Cheng, Yun Wang, Chao Wang, Ali Wagdy Mohamed, and Tiaojie Xiao. Adaptive differential evolution based on successful experience information. *IEEE Access*, 8:164611–164636, 2020.

[128] Chilukamari Rajesh and Sushil Kumar. An evolutionary block based network for medical image denoising using differential evolution. *Applied Soft Computing*, page 108776, 2022.

[129] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[130] Diganta Misra. Mish: A self regularized non-monotonic activation function. *arXiv preprint arXiv:1908.08681*, 2019.

[131] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[132] Yuting Su, Wanning Sun, Jing Liu, Guangtao Zhai, and Peiguang Jing. Photo-realistic image bit-depth enhancement via residual transposed convolutional neural network. *Neurocomputing*, 347:200–211, 2019.

[133] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[134] Shruti Jadon. A survey of loss functions for semantic segmentation. In *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–7. IEEE, 2020.

[135] Lingxi Xie and Alan Yuille. Genetic cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1379–1388, 2017.

[136] R Venkata Rao. Applications of tlbo algorithm and its modifications to different engineering and science disciplines. In *Teaching learning based optimization algorithm*, pages 223–267. Springer, 2016.

[137] R Rao. Review of applications of tlbo algorithm and a tutorial for beginners to solve the unconstrained and constrained optimization problems. *Decision science letters*, 5(1):1–30, 2016.

[138] Mohan Allam and M Nandhini. Optimal feature selection using binary teaching learning based optimization algorithm. *Journal of King Saud University-Computer and Information Sciences*, 2018.

[139] Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[140] Joes Staal, Michael D Abràmoff, Meindert Niemeijer, Max A Viergever, and Bram Van Ginneken. Ridge-based vessel segmentation in color images of the retina. *IEEE transactions on medical imaging*, 23(4):501–509, 2004.

[141] AD Hoover, Valentina Kouznetsova, and Michael Goldbaum. Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. *IEEE Transactions on Medical imaging*, 19(3):203–210, 2000.

[142] Christopher G Owen, Alicja R Rudnicka, Robert Mullen, Sarah A Barman, Dorothy Monekosso, Peter H Whincup, Jeffrey Ng, and Carl Paterson. Measuring retinal vessel tortuosity in 10-year-old children: validation of the computer-assisted image analysis of the retina (caiar) program. *Investigative ophthalmology & visual science*, 50(5):2004–2010, 2009.

[143] Jiong Zhang, Behdad Dashtbozorg, Erik Bekkers, Josien PW Pluim, Remco Duits, and Bart M ter Haar Romeny. Robust retinal vessel segmentation via locally adaptive derivative frames in orientation scores. *IEEE transactions on medical imaging*, 35(12):2631–2644, 2016.

[144] Thomas Köhler, Attila Budai, Martin F Kraus, Jan Odstrčilik, Georg Michelson, and Joachim Hornegger. Automatic no-reference quality assessment for retinal fundus images using vessel segmentation. In *Proceedings of the 26th IEEE international symposium on computer-based medical systems*, pages 95–100. IEEE, 2013.

[145] Amber L Simpson, Michela Antonelli, Spyridon Bakas, Michel Bilello, Keyvan Farahani, Bram Van Ginneken, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, et al. A large annotated medical image dataset for the development and evaluation of segmentation algorithms. *arXiv preprint arXiv:1902.09063*, 2019.

[146] Zifan Zhu, Qing An, Zhicheng Wang, Qian Li, Hao Fang, and Zhenghua Huang. Ilu-net: Inception-like u-net for retinal vessel segmentation. *Optik*, page 169012, 2022.

[147] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[148] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[149] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[150] Shuanglang Feng, Heming Zhao, Fei Shi, Xuena Cheng, Meng Wang, Yuhui Ma, Dehui Xiang, Weifang Zhu, and Xinjian Chen. Cpfnet: Context pyramid fusion network for medical image segmentation. *IEEE transactions on medical imaging*, 39(10):3008–3018, 2020.

[151] Zaiwang Gu, Jun Cheng, Huazhu Fu, Kang Zhou, Huaying Hao, Yitian Zhao, Tianyang Zhang, Shenghua Gao, and Jiang Liu. Ce-net: Context encoder network for 2d medical image segmentation. *IEEE transactions on medical imaging*, 38(10):2281–2292, 2019.

[152] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 11–19, 2017.

[153] Lei Mou, Yitian Zhao, Huazhu Fu, Yonghuai Liu, Jun Cheng, Yalin Zheng, Pan Su, Jianlong Yang, Li Chen, Alejandro F Frangi, et al. Cs2-net: Deep learning segmentation of curvilinear structures in medical imaging. *Medical image analysis*, 67:101874, 2021.

[154] Wentao Liu, Huihua Yang, Tong Tian, Zhiwei Cao, Xipeng Pan, Weijin Xu, Yang Jin, and Feng Gao. Full-resolution network and dual-threshold iteration for retinal vessel and coronary angiograph segmentation. *IEEE Journal of Biomedical and Health Informatics*, 26(9):4623–4634, 2022.

[155] Phu-Hung Dinh. Multi-modal medical image fusion based on equilibrium optimizer algorithm and local energy functions. *Applied Intelligence*, 51(11):8416–8431, 2021.

[156] Phu-Hung Dinh. A novel approach based on grasshopper optimization algorithm for medical image fusion. *Expert Systems with Applications*, 171:114576, 2021.

[157] Phu-Hung Dinh. Combining spectral total variation with dynamic threshold neural p systems for medical image fusion. *Biomedical Signal Processing and Control*, 80:104343, 2023.

[158] Phu-Hung Dinh and Nguyen Long Giang. A new medical image enhancement algorithm using adaptive parameters. *International Journal of Imaging Systems and Technology*, 32(6):2198–2218, 2022.

[159] Frank Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer, 1992.

[160] Malik Shehadeh Braik. Chameleon swarm algorithm: A bio-inspired optimizer for solving engineering design problems. *Expert Systems with Applications*, 174:114685, 2021.

[161] A Umamageswari, N Bharathiraja, and D Shiny Irene. A novel fuzzy c-means based chameleon swarm algorithm for segmentation and progressive neural architecture search for plant disease classification. *ICT Express*, 2021.

[162] Rizk M Rizk-Allah, Aboul Ella Hassanien, and Václav Snášel. A hybrid chameleon swarm algorithm with superiority of feasible solutions for optimal combined heat and power economic dispatch problem. *Energy*, 254:124340, 2022.

[163] Reham R Mostafa, Ahmed A Ewees, Rania M Ghoniem, Laith Abualigah, and Fatma A Hashim. Boosting chameleon swarm algorithm with consumption aeo operator for global optimization and feature selection. *Knowledge-Based Systems*, 246:108743, 2022.

[164] Phu-Hung Dinh. Medical image fusion based on enhanced three-layer image decomposition and chameleon swarm algorithm. *Biomedical Signal Processing and Control*, 84:104740, 2023.

[165] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[166] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[167] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, Georgia, USA, 2013.

[168] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[169] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[170] John Tsitsiklis, Dimitri Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9):803–812, 1986.

[171] Rongjian Zhao, Buyue Qian, Xianli Zhang, Yang Li, Rong Wei, Yang Liu, and Yinggang Pan. Rethinking dice loss for medical image segmentation. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 851–860. IEEE, 2020.

[172] Michael Yeung, Evis Sala, Carola-Bibiane Schönlieb, and Leonardo Rundo. Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation. *Computerized Medical Imaging and Graphics*, 95:102026, 2022.

[173] Jun Ma, Jianan Chen, Matthew Ng, Rui Huang, Yu Li, Chen Li, Xiaoping Yang, and Anne L Martel. Loss odyssey in medical image segmentation. *Medical Image Analysis*, 71:102035, 2021.

[174] Fabian Isensee, Paul F Jäger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. Automated design of deep learning methods for biomedical image segmentation. *arXiv preprint arXiv:1904.08128*, 2019.

[175] Guotai Wang, Xinglong Liu, Chaoping Li, Zhiyong Xu, Jiugen Ruan, Haifeng Zhu, Tao Meng, Kang Li, Ning Huang, and Shaoting Zhang. A noise-robust framework for automatic segmentation of covid-19 pneumonia lesions from ct images. *IEEE Transactions on Medical Imaging*, 39(8):2653–2663, 2020.

[176] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[177] Ali Hatamizadeh, Yucheng Tang, Vishwesh Nath, Dong Yang, Andriy Myronenko, Bennett Landman, Holger R Roth, and Daguang Xu. Unetr: Transformers for 3d medical image segmentation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 574–584, 2022.

[178] Patrice Y Simard, David Steinkraus, John C Platt, et al. Best practices for convolutional neural networks applied to visual document analysis. In *Icdar*, volume 3. Edinburgh, 2003.

[179] Chilukamari Rajesh, Ravichandra Sadam, and Sushil Kumar. An evolutionary u-shaped network for retinal vessel segmentation using binary teaching–learning-based optimization. *Biomedical Signal Processing and Control*, 83:104669, 2023.

[180] R Venkata Rao and Vivek Patel. An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *Scientia Iranica*, 20(3):710–720, 2013.

[181] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[182] Feng Zou, Lei Wang, Xinhong Hei, Debao Chen, and Bin Wang. Multi-objective optimization using teaching-learning-based optimization algorithm. *Engineering Applications of Artificial Intelligence*, 26(4):1291–1300, 2013.

[183] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[184] Spyridon Bakas, Hamed Akbari, Aristeidis Sotiras, Michel Bilello, Martin Rozycki, Justin S Kirby, John B Freymann, Keyvan Farahani, and Christos Davatzikos. Advancing the cancer genome atlas glioma mri collections with expert segmentation labels and radiomic features. *Scientific data*, 4(1):1–13, 2017.

List of Publications

Publications from the thesis

Journal papers:

1. **Chilukamari Rajesh**, and Sushil Kumar. "An evolutionary block-based network for medical image denoising using Differential Evolution." *Applied Soft Computing*, 121, p.108776, Elsevier, 2022. (**Published**, SCIE, *IF: 8.7*).

2. **Chilukamari Rajesh**, Ravichandra Sadam, and Sushil Kumar. "An evolutionary U-shaped network for Retinal Vessel Segmentation using Binary Teaching–Learning-Based Optimization." *Biomedical Signal Processing and Control*, 83, p.104669, Elsevier, 2023. (**Published**, SCIE, *IF: 5.1*).

3. **Chilukamari Rajesh**, Ravichandra Sadam, and Sushil Kumar. "An evolutionary Chameleon Swarm Algorithm based network for 3D Medical Image Segmentation." *Expert Systems With Applications*, 239, p.122509, Elsevier, 2023. (**Published**, SCIE, *IF: 8.5*).

4. **Chilukamari Rajesh**, Ravichandra Sadam, and Sushil Kumar. "A multi-objective neural architecture search based deep learning model for brain tumor detection in 3D medical images" *IEEE Transactions on Medical Imaging*, IEEE, 2023. (**Under review**, SCIE, *IF: 10.6*).

Conference papers:

1. **Chilukamari Rajesh**, and Sushil Kumar. "Automatic Retinal Vessel Segmentation using BTLBO." *In Soft Computing for Problem Solving: Proceedings of the SocProS* (pp. 189-200), Springer, 2022. (**Published**)

2. **Chilukamari Rajesh**, Ravichandra Sadam, and Sushil Kumar. "An evolutionary network for medical image segmentation with denoising." *In Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO)*, ACM, 2024. (**Submitted**)