UNIVERSITY OF GUANAJUATO

CAMPUS IRAPUATO-SALAMANCA
ENGINEERING DIVISION

DEEP LEARNING-BASED ALGORITHMS FOR STENOSIS DETECTION IN X-RAY CORONARY ANGIOGRAPHY IMAGING

DISSERTATION

SUBMITTED FOR OBTAINING THE DEGREE OF

DOCTOR IN ELECTRICAL ENGINEERING

BY

M.SC. EMMANUEL OVALLE-MAGALLANES

ADVISORS:

DR. JUAN GABRIEL AVINA-CERVANTES
DR. IVAN CRUZ-ACEVES

Salamanca, Gto., México. August, 2023.

Universidad de Guanajuato

Campus Irapuato-Salamanca
División de Ingenierías

Algoritmos basados en Aprendizaje Profundo para la detección de estenosis en imágenes de Angiografía Coronaria de Rayos X

tesis

que para obtener el Grado de

Doctor en Ingeniería Eléctrica

presenta

M.C. Emmanuel Ovalle Magallanes

Directores:

Dr. Juan Gabriel Aviña Cervantes
Dr. Ivan Cruz Aceves

Salamanca, Gto., México. Agosto, 2023.

Salamanca, Gto., a 22 de junio del 2023.

M. en I. HERIBERTO GUTIÉRREZ MARTIN
COORDINADOR DE ASUNTOS ESCOLARES
PRESENTE.-

Por medio de la presente, se otorga autorización para proceder a los trámites de impresión, empastado de tesis y titulación al alumno(a) **Emmanuel Ovalle Magallanes** del ***Programa de Doctorado en Ingeniería Eléctrica*** y cuyo número de ***NUA*** es: 147347 del cual soy director. El título de la tesis es: **“Deep Learning-based algorithms for stenosis detection in X-Ray Coronary Angiography imaging”.**

Hago constar que he revisado dicho trabajo y he tenido comunicación con los sinodales asignados para la revisión de la tesis, por lo que no hay impedimento alguno para fijar la fecha de examen de titulación.

ATENTAMENTE

Dr. Juan Gabriel Aviña Cervantes

NOMBRE Y FIRMA
DIRECTOR DE TESIS
SECRETARIO

Dr. Ivan Cruz Aceves

NOMBRE Y FIRMA
DIRECTOR DE TESIS

Dr. José Ruiz Pinales

NOMBRE Y FIRMA
PRESIDENTE

Dr. Dora Luz Almanza Ojeda

NOMBRE Y FIRMA
VOCAL

A. Hdez. Aguirre.

Dr. Arturo Hernández Aguirre

NOMBRE Y FIRMA
VOCAL

González José
Barbosa Joel

Dr. José Joel González Barbosa

NOMBRE Y FIRMA
VOCAL

A Dora Elisa Alvarado Carrillo por sus conocimientos e ideas. Ella es una parte importante de este logro. Gracias por tu apoyo y ánimo que me brindas tanto profesional como personal.

A mi familia: a mis padres María del Refugio y Simón, a mis hermanas Myriam y Berenice, por apoyarme para concluir esta tesis y en mi vida en general. En motivarme a llegar cada vez más lejos y nuca dejar de aprender.

A mi sobrino Isaac, por compartir tardes de juego llenas de creatividad y curiosidad, acabando con la frase: "Es ciencia".

Agradecimientos

Me gustaría dar las gracias a todas las personas que han contribuido de alguna manera a realizar el trabajo de esta tesis. En primer lugar, doy gracias a mis asesores, el Dr. Juan Gabriel Aviña Cervantes y el Dr. Ivan Cruz Aceves por haberme dado la oportunidad de trabajar con ellos, por sus enseñanzas y apoyo brindado durante mis estudios de doctorado.

Así mismo agradezco a los profesores de la División de Ingenierías Campus Irapuato-Salamanca (DICIS) por la dedicación que muestran por sus estudiantes. Al comité del examen de grado: Dr. José Ruiz Pinales, Dra. Dora Luz Almanza Ojeda, Dr. Arturo Hernández Aguirre y Dr. José Joel González Barbosa. Por sus correcciones y comentarios que ayudaron a enriquecer este trabajo

A la Universidad de Guanajuato, en especial a la DICIS por proporcionarme lo necesario para realizar mi formación como Doctor en Ingeniería Eléctrica (NUA **147344**).

Al Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT), por la beca otorgada en la convocatoria Becas Nacionales **CVU 626154**, con el número de becario **755609**.

CONAHCYT
CONSEJO NACIONAL DE HUMANIDADES
CIENCIAS Y TECNOLOGÍAS

LA VERDAD OS HARÁ LIBRES
UNIVERSIDAD DE GUANAJUATO

Abstract

Coronary heart disease is the leading cause of death worldwide, with an estimated 17.9 million deaths yearly. This condition is characterized by plaque building up inside the coronary arteries, which supply blood to the heart muscle. Plaque buildup narrows the arteries, reducing blood flow to the heart muscle. Early detection of coronary artery disease is crucial because it allows for timely intervention and treatment, which can help prevent the disease from progressing and potentially causing severe complications such as heart attacks, heart failure, and even death. This work proposes various novel deep learning-based methodologies for detecting stenosis in X-ray Coronary Angiography (XCA) images.

The first approach involves using pre-trained Convolutional Neural Networks (CNN) on ImageNet, such as VGG16, ResNet50, and Inception-v3, with fine-tuning and cut strategies. The proposed method outperforms vanilla pre-trained networks and models trained from scratch, with an optimized ResNet50 achieving the best results while requiring fewer parameters than Inception-v3 and VGG16.

The second approach is a Hybrid Classical-Quantum Network, which combines a classical CNN and a Quantum Network (QN). This scheme improves stenosis detection concerning classical transfer learning approaches. The main contribution of this research was related to the QN architecture, where multiple (and smaller) Variational Quantum Circuits (VQCs) can replace a single VQC boosting the hybrid model.

The third proposal is the Hierarchical Bezier Generative Model, which generates a large-scale labeled dataset for stenosis detection in XCA images. The generative model is based on prior knowledge of the blood vessel structure. It demonstrates the value of transferring the weights pre-trained using a more alike (artificial) dataset instead of the ImageNet dataset for stenosis detection tasks with only limited data available.

Lastly, Lightweight Residual Attention Networks (LRA-Nets) for stenosis detection were introduced, which consist of Deep-Wise Separable Convolutions, a pruning convolution kernel ratio, and an attention module. LRA-Nets outperform Residual models with or without attention mechanisms and achieve better classification performance with a smaller dilation ratio for the attention blocks.

The Gradient-weighted Class Activation Map (GradCAM) technique visually explains each model's prediction, allowing a probability of stenosis and an explainable heat map of high-attention regions that can be used in medical praxis.

Resumen

La enfermedad coronaria es la principal causa de muerte en todo el mundo, con un estimado de 17,9 millones de muertes al año. Esta afección se caracteriza por la acumulación de placa dentro de las arterias coronarias, que suministran sangre al músculo cardíaco. La acumulación de placa estrecha las arterias, lo que reduce el flujo de sangre al músculo cardíaco. La detección temprana de la enfermedad de las arterias coronarias es crucial porque permite la intervención y el tratamiento oportunos, lo que puede ayudar a prevenir que la enfermedad progrese y cause complicaciones graves, como ataques cardíacos, insuficiencia cardíaca e incluso la muerte. Este trabajo propone varias metodologías novedosas basadas en el aprendizaje profundo para detectar estenosis en imágenes de angiografía coronaria de rayos X.

El primer enfoque implica el uso de redes neuronales convolucionales (CNN) previamente entrenadas en ImageNet, como VGG16, ResNet50 e Inception-v3, con estrategias de ajuste y corte. El método propuesto supera a las redes preentrenadas y a los modelos entrenados desde cero, con un ResNet50 optimizado que logra los mejores resultados y requiere menos parámetros que Inception-v3 y VGG16.

El segundo enfoque es una red híbrida clásica-cuántica, que combina una CNN clásica y una red cuántica (QN). Este esquema mejora la detección de estenosis con respecto a los enfoques clásicos de aprendizaje por transferencia. La principal contribución de esta investigación estuvo relacionada con la arquitectura QN, donde múltiples (y más pequeños) circuitos cuánticos variacionales (VQC) pueden reemplazar un solo VQC impulsando el modelo híbrido.

La tercera propuesta es el modelo generativo jerárquico Bezier, que genera un conjunto de datos etiquetados a gran escala para la detección de estenosis en imágenes XCA. El modelo generativo se basa en el conocimiento previo de la estructura de los vasos sanguíneos. Demuestra el valor de transferir los pesos previamente entrenados utilizando un conjunto de datos (artificiales) más parecido en lugar del conjunto de datos de ImageNet para tareas de detección de estenosis con solo datos limitados disponibles.

Por último, se introdujeron las Redes de Atención Residual Livianas (LRA-Nets) para la detección de estenosis, que consisten en Convoluciones Separables Profundas, factor de poda para las capas convolucionales y un módulo de atención. LRA-Nets supera a los modelos Residual con o sin mecanismos de atención y logra un mejor rendimiento de clasificación empleando un factor de reducción más pequeño para los bloques de atención.

La técnica Gradient-weighted Class Activation Map (GradCAM) proporciona una explicación visual de la predicción de cada modelo, lo que permite obtener no solo una probabilidad de estenosis, sino también un mapa de calor explicable de las regiones de alta atención que se puede utilizar en la praxis médica.

Contents

List of Figures

List of Tables

List of Abbreviations

AWGN	Additive White Gaussian Noise	54
BiLSTM	Bi-directional Long-Short-Term Memory	22
BN	Batch Normalization	14
CAM	Channel Attention Module	66
CBAM	Convolutional Block Attention Module	66
CHD	Coronary Heart Disease	1
CIFAR	Canadian Institute For Advanced Research	36
COCO	Common Objects in Context	21
CTA	Computed Tomography Angiography	1
CNN	Convolutional Neural Network	7
DSC	Depthwise Separable Convolution	69
D-VQC	Distributed Variational Quantum Circuit	37
ECA	Efficient Channel Attention	65
Faster-RCNN	Faster Region-Based Convolutional Neural Networks	21
FID	Fréchet Inception Distance	55
GAP	Global Average Pooling	14
GAN	Generative Adversarial Network	50
GradCAM	Gradient-weighted Class Activation Map	15
HBGM	Hierarchical Bezier-based Generative Model	51
H-CQN	Hybrid Classical-Quantum Network	37
ILSVRC	ImageNet Large Scale Visual Recognition Challenge	13
IS	Inception Score	55
LRA-Nets	Lightweight Residual Attention Networks	69

MLP	Multi-Layer Perceptron	66
MNIST	Modified National Institute of Standards and Technology	36
MRA	Magnetic Resonance Angiography	1
NN	Neural Network	7
QN	Quantum Network	37
QIS-Net	Quantum-inspired Neural Network	36
ResNet	Residual Network	14
R-FCN	Region-based Fully Convolutional Networks	21
SAM	Spatial Attention Module	66
SE	Squeeze-and-Excitation	63
SGDM	Stochastic Gradient Descent with Momentum	24
SSD	Single Shot multi-box Detector	21
TPE	Tree-structured Parzen Estimator	71
US	Ultrasound	1
VGG	Visual Geometry Group	13
VQC	Variational Quantum Circuit	37
WC	Wasserstein Critic	55
XCA	X-Ray Coronary Angiography	1

1

Introduction

> *"The stars are yours, if you have the head, the hands, and the heart for them."*
>
> — Ray Bradbury, *R Is for Rocket*

1.1 | Motivation

Coronary Heart Disease (CHD) is the leading cause of death worldwide. According to the World Health Organization [2021], it is estimated that CHD takes the life of 17.9 million people (one-third of all global deaths) every year. Therefore, this condition seriously affects people's life quality and survival time. The primary pathological feature of CHD is the deficient supply of oxygen-rich blood to the heart muscle due to a partial narrowing or complete blocking of a coronary artery by adipose plaque formation [Britannica, The Editors of Encyclopaedia, 2021; National Heart, Lung, and Blood Institute, 2021]. This degenerative process, known as stenosis, can eventually cause life-threatening problems such as heart attacks and strokes [Athanasiou et al., 2017; National Heart, Lung, and Blood Institute, 2021]. Figure 1.1 illustrates coronary artery deterioration caused by stenosis.

In Mexico, as claimed by the report of the Instituto Nacional de Estadística y Geografía [2022], CHD has represented the most significant cause of death during the last decades. For instance, of the CHD casualties in 2021, a total of 226,703 cases were reported, that is, 7,999 more cases than in 2020. In particular, the ischemic losses (related to reduced blood flow for stenosis) represented 78.2%, with 177,263 cases.

Further tests may be carried out to diagnose CHD, including Computed Tomography Angiography (CTA) and other less invasive procedures such as Magnetic Resonance Angiography (MRA) and Ultrasound (US) imaging. However, X-Ray

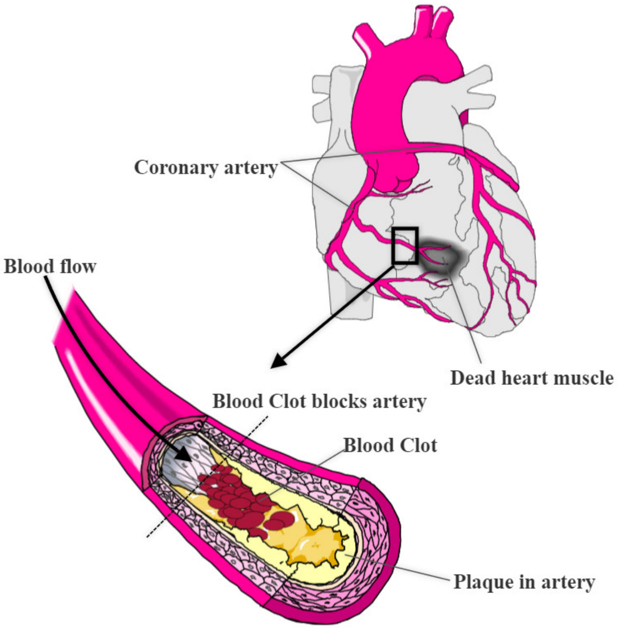


Figure 1.1: *Coronary stenosis characterization. The coronary artery blood flow is affected by the accumulation of plaque, generating a narrowing of the coronary artery. LifeART Collection Images Copyright ©1989-2001 by Lippincott Williams & Wilkins, Baltimore, MD.*

Coronary Angiography (XCA) remains the gold standard for CHD diagnosis [Nandalur et al., 2007] because of the accurate definition of coronary anatomy, obtaining high-resolution images of the main coronary arteries and their corresponding branches [Johal et al., 2021].

The XCA exams involve taking X-rays of the heart arteries, including dye injection and radiation exposure. They create detailed X-ray images allowing cardiologists to detect the blockage sections within coronary arteries, as shown in Figure 1.2.

In daily clinical practice, the physician finds the narrowed regions during an exhaustive visual examination of the X-ray images. As such, an in vivo assessment and treatment, such as angioplasty, where a short wire-mesh tube called a stent is inserted into the artery to restore the blood flow of the blocked or narrowed coronary arteries, can be performed [Athanasiou et al., 2017; Eckert et al., 2015; Johal et al., 2021].

Therefore, early stenosis detection is essential in cardiology to diagnose and provide suitable medical treatment before permanent heart damage. Nevertheless, the automation of this task is considered a challenging problem up to date because of background noise [Manson et al., 2019], non-coronary vascular structures, and multiple superposed branching points [Chang et al., 2019], among others. In the last decade, Deep Learning based methods have achieved outstanding performance

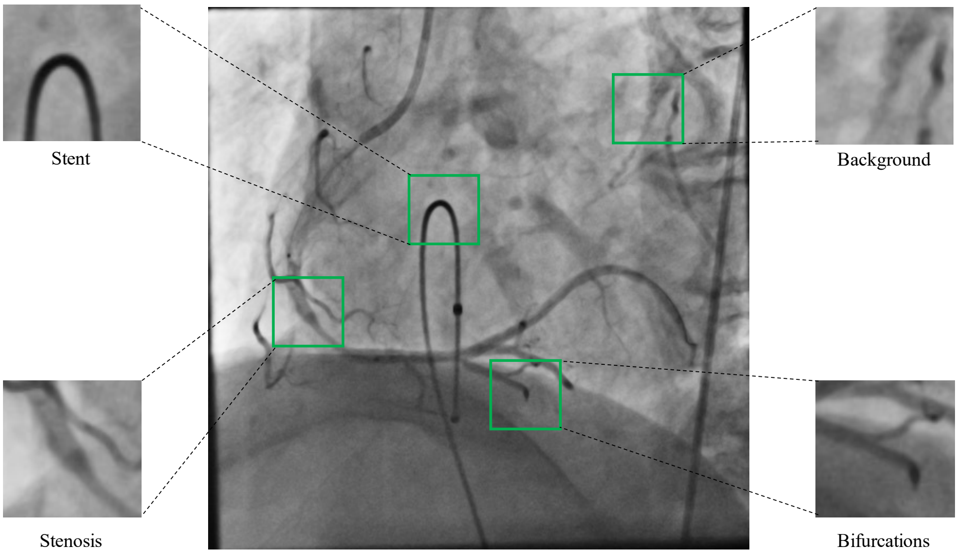


Figure 1.2: *XCA sample image. The created X-ray image can include specific characteristic regions, such as stents, background artifacts, coronary blood vessels with bifurcations, and stenosis cases.*

gains for several real-world applications. In particular, for computer vision problems, Convolutional Neural Networks (CNN) have been applied successfully in the medical imaging domain for different tasks, such as segmentation, identification, and classification [Mohapatra et al., 2021; Sarvamangala and Kulkarni, 2021]. The core of CNN is its capability to extract, select and classify features during the optimization step, while in traditional Machine Learning methods, each of these steps is conducted independently. For this reason, CNN has become a de-facto standard for computer vision problems.

This doctoral thesis addresses and introduces a state-of-the-art automatic stenosis detection in XCA images based on Deep Learning and Quantum Machine Learning algorithms that could support the physician's decision-making process. Herein, four main contributions are presented. The first method introduced was a network-cut and fine-tuning approach in that an optimal cut and fine-tuned layers were selected by minimizing the loss function. The second method presented a Hybrid Classical-Quantum Network, which involved connecting a Quantum Network to the head of a classical network to enhance the feature representation. Next, a Hierarchical Bezier Generative Model addresses the problem of a small XCA image dataset. And finally, a Lightweight Residual Attention Network, including separable convolutions, a pruning strategy, and an attention module, achieve high classification rates with lower computational requirements regarding the required parameters.

1.2 | X-Ray Coronary Angiography Stenosis Datasets

The public Deep Stenosis Detection Dataset (DSDD) [Antczak and Liberadzki, 2022] was employed in this work, and it consists of small grayscale XCA image patches of 32 × 32 pixels from different image positions and sources. It contains 1,519 images, where only 125 are positive cases of stenosis and 1,394 negative cases, which generate an unbalanced ratio of 1:11, *i.e.*, one positive case for eleven negative ones. This database does not specify a partition for training and testing sets.

Figure 1.3 shows examples of positive and negative stenosis cases in the dataset. A stenosis region is characterized by a rapid reduction of the blood vessel diameter, and a non-stenosis region by an uniform tubular shape of the blood vessel.

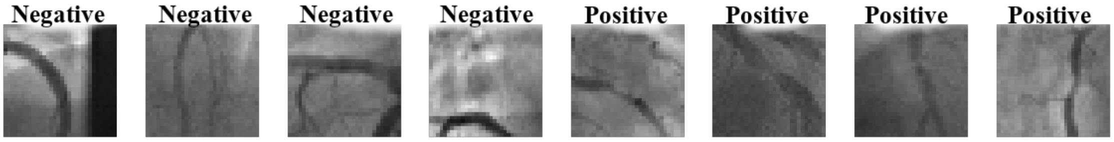


***Figure 1.3:** Dataset sample from X-ray Coronary Angiography images. Four negative and four positive stenosis cases are shown.*

1.3 | Objectives

1.3.1 | General Objective

The main objective of this research is to develop automatic methods for stenosis detection in XCA images, achieving high-performance and accuracy rates such that the proposed system can support medical practice decisions, thus, advancing the state-of-the-art in this problem. The following specific objectives have been established to accomplish the general objective.

1.3.2 | Specific Objectives

- Compare state-of-the-art convolutional neural network architectures for natural image classification and select the more suitable for stenosis detection in XCA images.

- Develop a new training strategy based on Transfer Learning to optimize the trainable parameters and develop more efficient models.

- Propose novel convolutional neural network architectures for stenosis detection in XCA images that achieve state-of-the-art results.
- Explore and develop hybrid models, combining quantum machine learning and deep learning to improve classical convolutional models.
- Compare different strategies to deal with the limited and unbalanced annotated data, *i.e.*, labeled XCA images, to train a convolutional network from scratch and achieve adequate training.
- Propose a new generative model that generates new stenosis samples to pre-train a convolutional model with synthetic data and improve classification performance.

1.4 | Document structure

The rest of this dissertation is organized as follows:

- Chapter 2 introduces the mathematical background of Convolutional Neural Networks, including their core components such as convolutional layers, pooling layers, and activation functions. Also, the Cross-Entropy Loss function employed for optimization is described. Next, modern convolution architectures are depicted. Finally, are given visual explanations of how Convolutional Neural Networks model predictions can be performed.
- Chapter 3 presents a novel Transfer Learning and Network cut for stenosis detection employing different modern convolutional architectures. Numerical results show outstanding performance for this task. Moreover, an L2-Constrained SoftMax loss function was used to improve the model accuracy further.
- Chapter 4 solved the stenosis classification problem employing a novel hybrid classical-quantum convolutional neural network. First, a background in quantum computing is introduced. Secondly, a novel quantum layer reduces the computation time and processes a more significant number of classical features. Finally, extensive experiments demonstrate the potential of the hybrid model.
- Chapter 5 addresses the lack of annotated data for accurate stenosis detection in XCA images by generating synthetic images that model real coronary vascular structure regions. A robust Bezier-based generative model faithfully generates image patches with blood vessel structures, including bifurcations and stenosis.

- Chapter 6 brought the mathematical foundations of attention mechanisms and a sequential model-based optimization approach, the Tree-structured Parzen Estimator. These two modules are the key components of an Attention-based Convolutional Neural Network for stenosis detection. Furthermore, this architecture was developed to be Lightweight regarding the number of parameters and operations.
- Chapter 7 includes the contributions and publications achieved during the doctoral studies. Furthermore, final remarks on this work and future research directions are presented.

2

Convolutional Neural Networks Review

"Begin at the beginning, the King said gravely, and go on till you come to the end: then stop."

— Lewis Carroll, *Alice in Wonderland*

This chapter introduces several fundamental concepts in Convolutional Neural Networks (CNN) that will be used in this chapter and subsequent chapters, including their essential components and hyperparameters, loss functions, and optimizers. Moreover, a review of modern convolutional architectures employed as backbone models for this doctoral research is included. Finally, the classification performance metrics that will be used are described.

2.1 | Convolutional Neural Networks

A Convolutional Neural Network (CNN) is an end-to-end supervised Deep Learning algorithm which is a specific type of Neural Network (NN) that is designed for image analysis exploiting the semantic representation of image pixels (*i.e.*, 2D grayscale or 3D-colored images). CNNs consist of alternating convolutional and pooling layers attempting to extract discriminative features (*e.g.*, edges, interest points) across a set of input images upon going deeper and deeper into the network. At the network's top, the features feed a set of fully connected layers to estimate the correct class for each input. The last layers are also known as the network head.

The *convolutional layer* uses K filters that perform convolution operations over the input data. All filters are convolved across the complete input image to produce a 2D activation map for each filter during optimization. Formally, let $f_{conv}(\cdot, \mathbf{W})$:

$\mathbb{R}^{h^{in} \times w^{in} \times c^{in}} \rightarrow \mathbb{R}^{h^{out} \times w^{out} \times 1}$ be a single standard convolution operation that takes as input $\mathbf{X}^{in}$ and produces $\mathbf{X}^{out}$ parameterized by the kernel $\mathbf{W} \in \mathbb{R}^{k \times k}$ computed as:

$$\mathbf{X}^{out}(i,j) = f_{conv}(\mathbf{X}^{in}, \mathbf{W}) = \sum_{u=1}^{k} \sum_{v=1}^{k} \sum_{m=1}^{c^{in}} \mathbf{W}(i,j) * \mathbf{X}^{in}_{m}(i + r \times u, j + r \times v), \tag{2.1}$$

where $*$ represents the convolution operation, k is the filter size, and r is the dilation ratio. Here h^{in}, h^{out}, and w^{in}, w^{out} denote the height and width of the input and output feature maps, respectively, and c^{in} indicates the number of input channels.

Figure 2.1 exhibits the general mechanism of the convolution operation. Therefore,

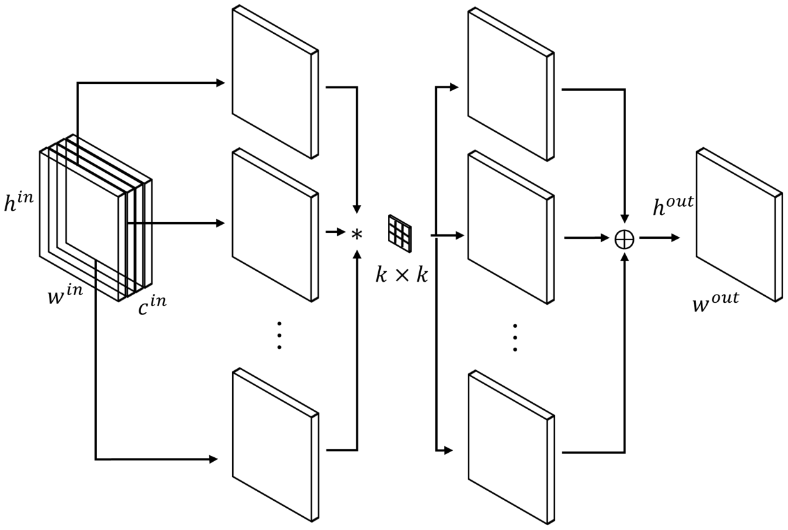


Figure 2.1: Single standard convolution operation. During the convolution, a kernel of size k × k is convolved with the input feature maps across each channel.

a convolutional layer learns dynamic filters that allow the network to detect specific or relevant visual features. Hitherto, the features extracted from low-level layers are more generic (*e.g.*, luminance, edges, contrasting colors, and curves) than those extracted by the top layers. The resulting output $\mathbf{X}^{out}$ is called a *features map* or *activation map*. Thus, a convolutional layer with K filters produces K-features maps such that $\mathcal{L}_{conv}(\cdot, \mathbf{W}) : \mathbb{R}^{h^{in} \times w^{in} \times c^{in}} \rightarrow \mathbb{R}^{h^{out} \times w^{out} \times c^{out}}$, now parameterized by the set of $c^{out} = K$ kernels as $\mathbf{W} \in \mathbb{R}^{k \times k \times c^{out}}$.

A common choice is to keep a small kernel size at $k = \{3, 5, 7\}$ to learn small patterns, that can be easily computed. Notice that the convolution operation requires some hyperparameters to be set that control the height and width of the output feature map. The dilation ratio r enlarges the receptive field without increasing the number of parameters or the amount of computation. Finally, the stride s controls the number of pixels by which the convolution kernel moves after each operation. Values of $r = 1$ and $s = 1$ are the typical configuration of these hyperparameters. Then, the spatial resolution of the output feature map can be calculated as follows:

$$h^{out} = \left\lfloor \frac{h^{in} + 2p - r \times (k-1) - 1}{s} + 1 \right\rfloor, \tag{2.2}$$

$$w^{out} = \left\lfloor \frac{w^{in} + 2p - r \times (k-1) - 1}{s} + 1 \right\rfloor, \tag{2.3}$$

where $\lfloor \cdot \rfloor$ is the round toward negative infinity and p is the padding determining how many zero values are added to the image's border.

An *activation function* is usually applied after the convolution operation to clip the range of values of the feature map and increase the non-linearity aimed at solving the vanishing and exploding gradients problem. The standard activation functions include: Sigmoid, Tanh, ReLU, and leaky ReLU, shown in Figure 2.2. The characteristic curve of the Sigmoid curve is S-shaped with a consistently positive output in the range $[0, 1]$. The

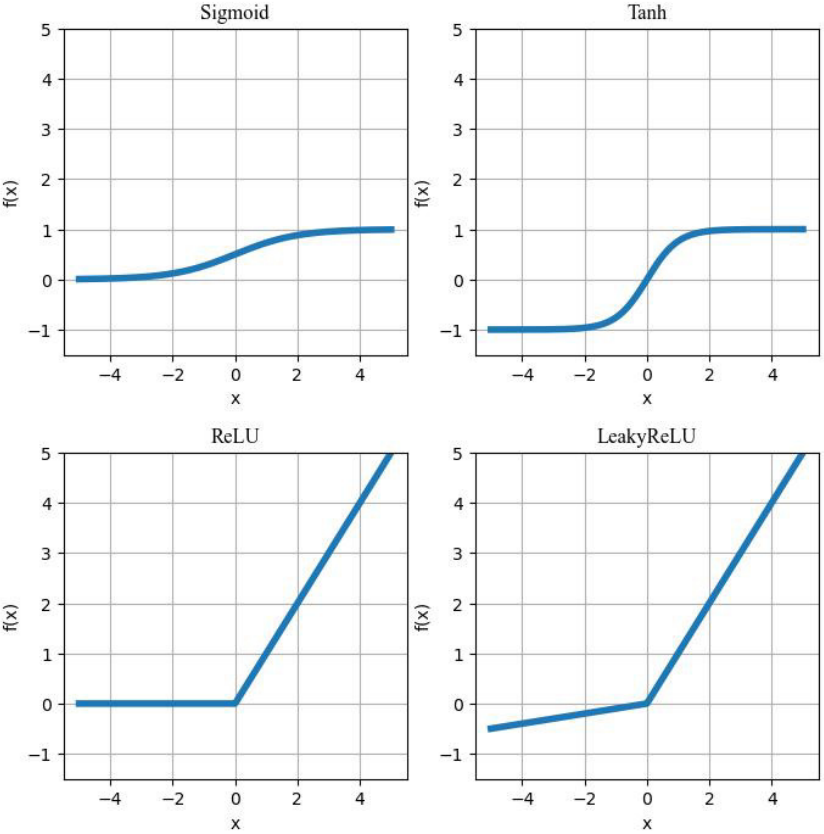


Figure 2.2: *Common activation functions. An activation function introduces the non-linearity and clips the range of values of the feature map after the convolution operation.*

Sigmoid tends to be zero when the input is large or small; thus, the gradient becomes very small. The formula of the Sigmoid function is defined as:

$$\sigma(x) = \frac{1}{1+\mathrm{e}^{-x}}. \tag{2.4}$$

Like the sigmoid function, the Tanh function's input is real numbers but with a range from $[-1,1]$. In addition, this function is characterized by exhibiting a fast saturation behavior, leading that the gradient approaching zero when the output is close to ± 1. Bearing in mind that:

$$\tanh(x) = \frac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}}. \tag{2.5}$$

The ReLU function is a piece-wise function that forces the output to be zero if the input value is less than or equal to zero. Otherwise, the output value is equal to the input value. The ReLU function lacks saturation and is computationally more efficient to compute, it is defined as follows:

$$\delta(x) = (x)^{+} = \max(0,x). \tag{2.6}$$

The leaky ReLU is based on a ReLU and includes a slight slope, such as $\alpha = 0.1$ for negative values, ensuring that these inputs are never ignored, as given next,

$$\delta_L(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha x, & \text{otherwise.} \end{cases} \tag{2.7}$$

In summary, the ReLU is a good default choice of activation function after the convolution operation since there is no vanishing gradient problem, unlike sigmoid and Tanh functions, and there is no hyperparameter to set like leaky-ReLU. Notice that these activation functions were defined for scalar input. For the vector input, the scalar activation function is element-wisely applied.

The *pooling layer*, also known as the downsampling layer, reduces the spatial size of the input feature map $\mathbf{X}^{in}$. Consequently, the number of parameters in the network is also reduced and controls overfitting [Scherer et al., 2010]. Typically, the pooling layer is applied between successive convolution layers so that deeper layers learn different scale features. The most common pooling operations are max-pooling and average pooling which operate in a channel-wise manner; thus, they map $\mathbb{R}^{h^{in}\times w^{in}\times c^{in}} \to \mathbb{R}^{h^{out}\times w^{out}\times c^{in}}$.

In max-pooling, the maximum value in a local neighborhood of size $k \times k$ centered at (i,j) is taken in each channel of the input feature map as follows:

$$\mathbf{X}_m^{out}(i,j) = f_{\max}(\mathbf{X}_m^{in}) = \max_{u,v} \mathbf{X}_m^{in}(i+u, j+v), \tag{2.8}$$

where $u = \{1, 2, \cdots, k\}$ and $v = \{1, 2, \cdots, k\}$ are the vertical and horizontal index in the local neighborhood, respectively.

On the other hand, average pooling computes the mean value of the window, such as:

$$\mathbf{X}_m^{out}(i,j) = f_{avg}(\mathbf{X}_m^{in}) = \frac{1}{n}\sum_{u=1}^{k}\sum_{v=1}^{k}\mathbf{X}_m^{in}(i+u, j+v), \tag{2.9}$$

where $n = k \times k$. Like the convolution operation, the pooling layer requires a size and a stride. A size of 3×3 pixels with a 2-pixel stride is common practice. It is worth noting that a strided convolution can be employed instead of a pooling layer for down-sampling.

After feature extraction performed by the convolutional layers, *fully-connected* or dense layers, learn nonlinear combinations given a flattened version of these features to accomplish the classification task. Multiple dense layers can be stacked to reduce the dimensionality of the features (*i.e.*, a Multi-Layer Perception Head). Hence, the dense layer is characterized by connecting every neuron in the previous layer to every neuron in the next layer. In such a way, the output size of the last dense layer corresponds to the number of classes to be classified. The dense layer implements the following operation:

$$\mathbf{y} = f_{dense}(\mathbf{x}^{in}, \mathbf{w}) = f_{act}(\mathbf{w}^{\intercal}\mathbf{x}^{in} + \mathbf{b}), \tag{2.10}$$

where f_{act} is an activation function, $\mathbf{x}^{in}$ is the flattened feature vector, $\mathbf{w}$ is the weights vector of the layer, and $\mathbf{b}$ is the bias term. Ergo, the head of the CNN can be expressed as $\mathcal{L}_{head}(\cdot, \mathbf{w}) : \mathbb{R}^N \to \mathbb{R}^c$, with N standing for the flattened dimension of the last feature maps, and c is the number of classes. Therefore, a CNN can be defined as follows:

$$\mathcal{N} = \mathcal{L}_{head}(\mathbf{x}^{(l)}, \mathbf{w}^{(l)}) \circ \mathcal{L}_{conv}(\mathbf{X}^{(l-1)}, \mathbf{W}^{(l-1)}) \circ \cdots \circ \mathcal{L}_{conv}(\mathbf{X}^{(1)}, \mathbf{W}^{(1)}), \tag{2.11}$$

where (l) is the l-th layer and $\circ$ defines the stack of layers.

Figure 2.3 illustrates a typical CNN architecture consisting of blocks of convolutional layers followed by pooling layers (feature extraction). Finally, consecutive fully connected layers (feature selection) are used to generate the required output neurons.

2.2 | SoftMax and Cross-Entropy Loss

The *SoftMax* function is usually the activation function in the last fully connected layer. The main purpose of the SoftMax function is to map the vector from the last fully connected layer of arbitrary real numbers (*logits*-$\mathbf{l}$) into probabilities with real values

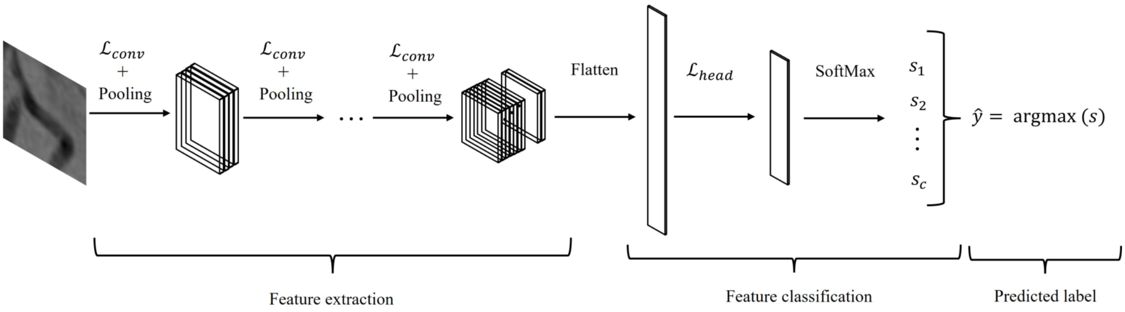


Figure 2.3: *Typical CNN architecture with convolutional, pooling, and dense layers.*

in the range $(0, 1)$ that sum up to 1.0. Formally, the SoftMax function $\varsigma(\mathbf{l}) : \mathbb{R}^C \rightarrow \mathbb{R}^C$ is defined as:

$$\hat{y}^{(c)} = \varsigma(\mathbf{l})^{(c)} = \frac{\mathrm{e}^{l^{(c)}}}{\sum\limits_{k=1}^{C} \mathrm{e}^{l^{(k)}}}, \tag{2.12}$$

where $l^{(j)}$ denotes the j-th element ($j = \{1, 2, \ldots, C\}$, where C is the number of classes). The SoftMax can be interpreted as the probabilities that the target class be $t = c$, for $c = \{1, 2 \cdots, C\}$ given the input $\mathbf{l}$ as:

$$\hat{y}^{(c)} = \varsigma(\mathbf{l})^{(c)} = P(t = c|\mathbf{l}), \tag{2.13}$$

Any classification algorithm minimizes the number of misclassified examples in the training data. In the NN, optimizing the weights with the backpropagation algorithm and any gradient descent optimizer is necessary. The model weights are optimized to increase the probabilities for the correct classes and decrease them otherwise for all training examples. Let $\mathcal{D}_{train} = \{\mathbf{X}, \mathcal{Y}\}$, $\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(n)}\}$, $\mathcal{Y} = \{y^{(1)}, y^{(2)}, \cdots, y^{(n)}\}$, be the training dataset where $\mathbf{X}$ is the input data and $\mathcal{Y}$ their respective classes, $\boldsymbol{\theta}$ the parameters of the model, and $\mathbf{l} = \{l^{(1)}, l^{(2)}, \cdots, l^{(n)}\}$ the logits generated by the last layer of the model, the likelihood can be defined as follows:

$$L(\boldsymbol{\theta}) = P(\mathcal{Y}|\mathbf{l}, \boldsymbol{\theta}) = \prod_{i=1}^{n} P\left(y^{(i)}|l^{(i)}, \boldsymbol{\theta}\right). \tag{2.14}$$

Maximizing the likelihood is the same as maximizing the log-likelihood because taking the log allows replacing the product into a sum, which is numerically more stable and easier to optimize:

$$\ell(\boldsymbol{\theta}) = \log P(\mathcal{Y}|\mathbf{l}, \theta) = \sum_{i=1}^{n} \log P(y^{(i)}|l^{(i)}, \boldsymbol{\theta}). \tag{2.15}$$

Given a one-hot (or probabilistic) $y^{(c)}$, and the model prediction $\hat{y}^{(c)}$, we can write the log-likelihood function as:

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{c=1}^{C} y^{(c)} \log \hat{y}^{(c)} \tag{2.16}$$

Notice that this maximization problem needs to be changed using duality into a minimization problem in order to use gradient descent optimizers. Thus, the Negative log-likelihood (NLL) used in the *Cross-Entropy loss* is taken as $\xi(\boldsymbol{\theta}) = -\ell(\boldsymbol{\theta})$.

2.3 | Modern Convolutional Neural Networks

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was developed in 2010 to serve as a standard benchmark for image classification. The task consists of classifying imagery obtained from different search engines into the correct one of 1,000 possible classes. The training set has around 1.2 million images, with 732 to 1,300 training images per class. In 2012, Krizhevsky et al. [2012] proposed a CNN called AlexNet, to solve the ILSVRC. AlexNet vastly outpaced the performance of traditional machine learning algorithms. The AlexNet architecture had eight hidden layers: the first five were convolutional layers with ReLU activation function, some followed by max-pooling layers, and the last three fully connected layers. Different kernel size was employed in the convolutional layers. In the first layer, an 11×11 kernel; in the second, a 5×5, and 3×3 in the subsequent three layers. However, it shows a significant problem: it had around 60 million parameters in only five layers, which led to overfitting.

Subsequently, different CNN dominated the ILSVRC. Then, in 2014, Simonyan and Zisserman [2015b] proposed the Visual Geometry Group (VGG) architecture that follows the critical ideas of AlexNet. The fundamental idea behind this architecture is to increase the network depth with small 3×3 convolution filters with a stride of 1 pixel, which significantly improves the AlexNet. The VGG has convolutional layers starting with 64 feature maps per convolution block (two to four successive convolutions layers) and increasing to 512 feature maps, each with a ReLU activation function, followed by max-pooling layers and three fully connected layers at the top of the model. This configuration allows pushing the network depth to 11-19 weight layers, considered very deep at the time.

Another winner architecture of the ILSVRC was Inception, which was proposed in 2014 by Szegedy et al. [2015]. Inception contains 48 convolutional layers that form three distinct types of Inception modules (A, B, and C) containing parallel 1×1, 3×3, and 5×5 convolution kernels whose outputs are concatenated to reduce the number of connections/parameters without decreasing network efficiency. Later the authors proposed an improved version of Inception, the so-called Inception-v3 [Szegedy et al., 2016]. Here, the 5 × 5 convolutions were replaced by two consecutive 3 × 3 convolutions and 3 × 3 convolutions with a 1 × 3 followed by a 3 × 1 convolution. Additionally, a Batch Normalization (BN) layer was applied after each convolution operation [Ioffe and Szegedy, 2015]. This factorization mechanism can replace any $k \times k$ convolution with a $1 \times k$ convolution followed by a $k \times 1$ convolution. Let $\mathbf{X}^{conv}$ be the feature maps after the convolution operation and before a non-linearity function, a BN layer is computed over a batch of size B, $\mathbf{X}_B^{conv} = \{\mathbf{X}_i^{conv}; i = 1, \cdots, b\}$ such that:

$$\hat{\mathbf{X}}_i^{conv} = \text{BN}(\mathbf{X}_i^{conv}) = \gamma \frac{\mathbf{X}_i^{conv} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta, \tag{2.17}$$

where γ and β are the hyperparameters to learn that normalize the batch. By noting that μ_B and σ_B^2 are the mean and variance of the batch, and ϵ is a small value avoiding zero divisions.

The 2015 ILSVRC winner was the Residual Network (ResNet) proposed by He et al. [2016]. This family of networks introduces a residual connection to the model that successfully tackled the vanishing gradient problem. A residual connection allows skipping to process a few layers. Residual blocks formed by convolutional layers and skip connections increased the number of convolutional layers in order of magnitude, from tens to hundreds, such as ResNet18, 34, 50, and 101. Formally, a residual block is defined as:

$$\mathbf{X}^{out} = \delta\left(\mathbf{F}_{res}(\mathbf{X}^{in}, \mathbf{W}_i) + \mathbf{F}_{down}(\mathbf{X}^{in}, \mathbf{W}_s)\right), \tag{2.18}$$

where $\mathbf{F}_{res}(\cdot, \mathbf{W}_i)$ represents the residual mapping to be learned, *i.e.*, multiple convolutional layers, $\mathbf{F}_{down}(\cdot, \mathbf{W}_s)$ performs a linear projection to match the dimensions (*e.g.*, when the input/output channels changed), and δ is the ReLU function. The residual mapping follows the order of execution as Convolution → BN → ReLU → Convolution → BN. Additionally, the ResNet replaced the fully connected layers of the VGG with a Global Average Pooling (GAP) [Lin et al., 2013] layer and only one fully connected layer with 1,000 neurons, which produces the output class probabilities. This operation is parameter-free and applies a dimensionality reduction; thus, it reduces each feature map $\mathbf{X}_m \in \mathbb{R}^{h^{in} \times w^{in}}$ to a single scalar value as follows:

$$z_m = \mathcal{L}_{\text{GAP}}(\mathbf{X}_m) = \frac{1}{h^{in} \times w^{in}} \sum_{i=1}^{h^{in}} \sum_{j=1}^{w^{in}} \mathbf{X}_m(i, j). \tag{2.19}$$

Table 2.1: *Top-1 and Top-5 comparison performance of modern CNN architectures on the ImageNet validation set. The model size is given in Megabytes (MB), and the number of parameters in Millions (M).*

Model	Size [MB]	Top-1 / Top-5 Accuracy	Parameters [M]	Depth
AlexNet	238	0.63/0.84	62.3	8
VGG16	528	0.71/0.90	138.4	16
ResNet50	98	0.74/0.92	25.6	107
Inception-v3	92	0.77/0.93	23.9	189

To sum up, Table 2.1 compares these four families of models on the ILSVRC concerning model size, classification error rate, and model depth. Only the best VGG and ResNet configurations are displayed.

2.4 | Gradient Class Activation Map

The Gradient-weighted Class Activation Map (GradCAM) [Selvaraju et al., 2017] technique visually explains the prediction for a CNN model. GradCAM uses the gradients of a given class, backpropagating the information into a particular convolutional layer (in most cases into the last convolutional layer) to generate a coarse localization map highlighting the input image regions strongly influencing the predicted class. Taking a particular class c, the GradCAM is obtained as follows:

$$L^{c}_{\text{Grad-CAM}} = \delta\left(\sum_{k} \alpha^{c}_{k}\mathbf{X}^{(k)}\right), \tag{2.20}$$

where δ is the ReLU activation function, $\mathbf{X}^{(k)}$ are the feature maps at the k-th layer that receives the gradient of the class, and α^{c}_{k} is the neuron importance weight defined as:

$$\alpha^{c}_{k} = \mathcal{L}_{GAP}\left(\frac{\partial l^{(c)}}{\partial \mathbf{X}^{(k)}}\right). \tag{2.21}$$

Notice that the score gradient $l^{(c)}$ for class c are the logits computed before the SoftMax function. Then, these gradients are backpropagated and global-average-pooled. Hence, $\alpha^{(c)}_{k}$ captures the importance of their respective feature maps for a target class c. The general pipeline of the GradCAM is displayed in Figure 2.4.

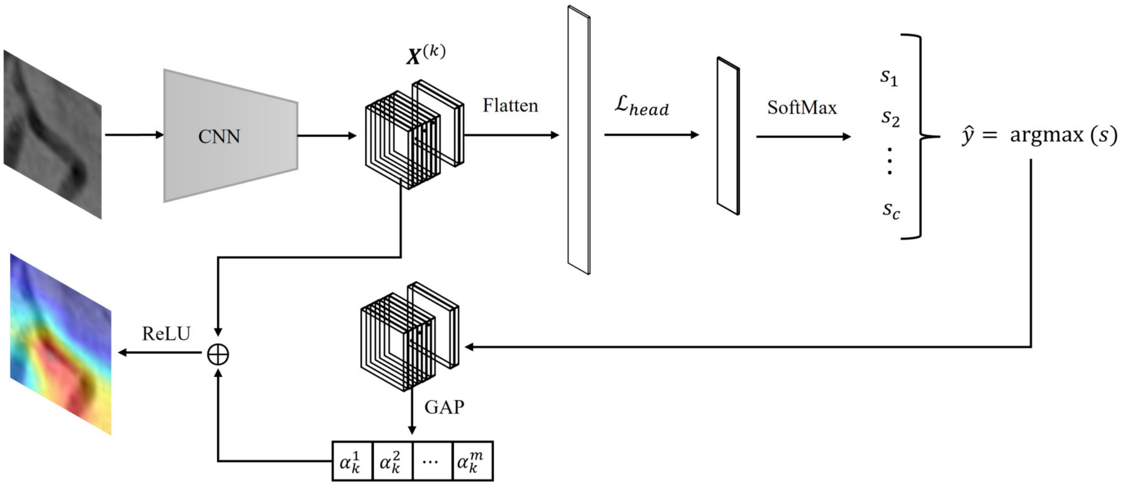


Figure 2.4: *GradCAM pipeline. The GradCAM method generates a coarse localization map highlighting regions strongly influencing the predicted class.*

2.5 | Classification Performance Metrics

In this dissertation, five binary metrics were used for measuring the performance of the stenosis classification algorithms. Those metrics can be derived from a binary confusion matrix, which is a 2 × 2 matrix with the structure described in Table 2.2

Table 2.2: *Confusion matrix for binary classification.*

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

In the binary confusion matrix, TP refers to the number of true positives, which are the stenosis cases correctly classified, and TN is the number of true negatives; thus, the no stenosis cases correctly classified. Similarly, FP denotes the false positives cases, and FN represents the number of false positives, which are the incorrectly classified instances of no-stenosis and stenosis, respectively.

In such a way, the evaluation metrics are Accuracy, Sensitivity, Specificity, Precision, and F_1-score, defined as follows. Accuracy is the ratio of the correctly classified test instances over the total number of test cases and is formally defined as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{2.22}$$

Sensitivity or true positive rate measures the correctly classified positive instances as follows:

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{2.23}$$

Specificity or true negative rate gives the rate of correctly classified negative instances, which is given by:

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}. \tag{2.24}$$

Precision gives a positive predictive value. This value provides how efficiently the classifier avoids FP. It can be formally defined as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{2.25}$$

The F_1-score is a balanced measure that computes the harmonic mean of precision and sensitivity and is given by:

$$\text{F}_1\text{-score} = \frac{\text{TP}}{\text{TP} + 0.5 \times (\text{FP} + \text{FN})}. \tag{2.26}$$

After presenting the foundation of CNNs, the GradCAM technique, and the classification metrics that will be employed in this work, stenosis detection based on transfer learning is discussed in the next chapter.

3

Transfer Learning for Stenosis Detection

"Human has always striven to retain the past, to keep it convincing; there's nothing wicked in that. Without it we have no continuity; we have only the moment."

— Philip K. Dick, *Now Wait for Last Year*

3.1 | Mathematical Foundations

A *domain* $\mathcal{D}$ is composed of a feature space $\mathcal{X}$ and a marginal distribution $P(\mathbf{X})$ with $\mathbf{X} = \{\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(n)} \in \mathcal{X}\}$ such that:

$$\mathcal{D} = \{\mathcal{X}, P(\mathbf{X})\}. \tag{3.1}$$

Two domains are different if they have distinct feature spaces or other marginal probability distributions. Hence, given a specific domain, a *task* $\mathcal{T}$ is defined by:

$$\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}, \tag{3.2}$$

which consists of a label space $\mathcal{Y}$ and a predictive function $f(\cdot)$. The function $f(\cdot)$ is expected to be learned from the training data within pairs $\{\mathbf{x}^{(i)}, y^{(i)}\}$, where $\mathbf{x}^{(i)} \in \mathbf{X}$ and $y^{(i)} \in \mathcal{Y}$. Notice that once the function $f(\cdot)$ is trained, it can be used to predict the label of a new instance $\hat{\mathbf{x}}$. Some machine learning models compute a conditional distribution of the classes, such as:

$$f(\hat{\mathbf{x}}) = \{P(y^{(k)}|\hat{\mathbf{x}})\}, \quad y^{(k)} \in \mathcal{Y}. \tag{3.3}$$

In this manner, *transfer learning* employs the knowledge implied in a given source domain $\mathcal{D}_{src}$ with their respective learning task $\mathcal{T}_{src}$ to improve the learning of a target decision function $f_{trg}(\cdot)$ on the target domain $\mathcal{D}_{trg}$ with its target task $\mathcal{T}_{trg}$. Transfer learning in the deep learning context requires a pre-trained model $f_{src}(\mathbf{X}_{src}) = f_{src}^{(h)} \circ f_{src}^{(n)} \circ \cdots \circ f_{src}^{(1)}(\mathbf{X}_{src})$, where each layer $f^{(i)}$ has parameters $\boldsymbol{\theta}_i$ that optimized a source loss ξ_{src} employing a large dataset following the source marginal distribution as $\mathcal{D}_{src} \sim P_{src}(\cdot)$. Notice that $f_{src}^{(h)}$ is the head of the model, typically the dense layers (feature selection), and the first n-layers are convolutional layers (feature extraction).

In such a way, given a smaller dataset following the target distribution as $\mathcal{D}_{trg} \sim P_{trg}(\cdot)$ the key idea is to re-train (fine-tune) some layers of the pre-trained model while keeping some frozen in training, solving the optimization problem:

$$\arg\min_{\boldsymbol{\theta}^{(i)}} \xi_{trg}\left(f_{src}(\boldsymbol{\theta}^{(1)}, \cdots, \boldsymbol{\theta}^{(n)}, \boldsymbol{\theta}^{(h)})\right), \quad \forall i \in S \tag{3.4}$$

where S is the subset of layers to optimize, typical choices are fine-tuning all $S = \{1, \ldots, n\}$, freezing all $S = \{\}$, and fine-tuning the last k-layers $S = \{n-k, \cdots, n\}$ resulting in the fine-tuned model $f_{trg}(\mathbf{X}_{trg})$. It is important to note that the last dense layer with weights $\boldsymbol{\theta}^{(h)}$ is changed to match the target task (*i.e.,* the number of classes). Also, notice that this layer is always trainable. Figure 3.1 illustrates the general pipeline of transfer learning for deep learning, particularly for CNN.

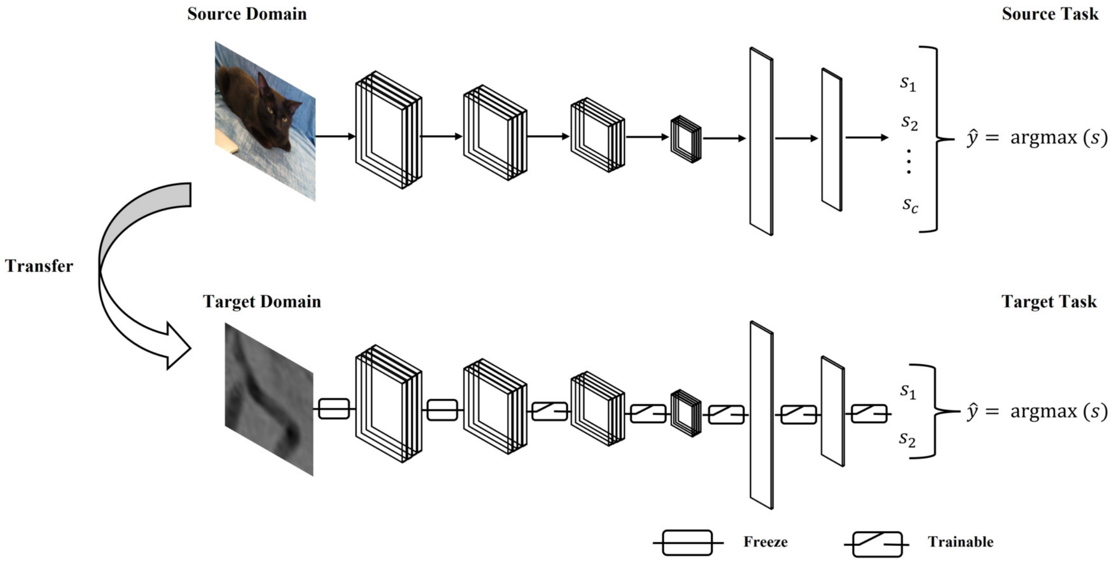


Figure 3.1: *Transfer Learning for CNN pipeline. The top layers are set as trainable; meanwhile, the bottom ones remain frozen.*

3.2 | Related Work

Machine learning-based methods have been proposed to detect automatic stenosis in XCA images [Kishore and Jayanthi, 2019; Sameh et al., 2017; Wan et al., 2018]. These studies first extract discriminative features based on texture and shape information. Then, a feature selection process is performed to choose the most suitable features to feed a classifier. Finally, different classifiers, such as Naive Bayes and Support Vector Machine, accomplish stenosis detection. However, features extracted in a hand-crafted manner limit the effectiveness of feature selection and, consequently, the classification performance.

Despite the dissimilarity between natural and medical imaging, recent studies have developed deep learning methods based on transfer learning to tackle medical imaging domain detection tasks, such as chest imaging [Xu et al., 2019; Yadav and Jadhav, 2019], breast imaging [Shen et al., 2019; Wu et al., 2018], and retinal imaging [Chakravarthy et al., 2019], showing outstanding performance compared to the hand-extracted feature-based methods. However, the success of transfer learning with convolutional networks relies on the generality of the learned representations constructed from a large database like ImageNet [Azizpour et al., 2015].

For stenosis detection in XCA images, Wu et al. [2020] proposed a deep learning framework consisting of two stages. First, candidate frames were selected from the full raw XCA based on the segmentation results that produce a UNet [Ronneberger et al., 2015]. Subsequently, an object-based detection network employing a VGG [Simonyan and Zisserman, 2015a] as a backbone network classifies the stenosis regions. The model was trained from scratch (*i.e.,* the weights of the kernels were initialized randomly).

Following the same idea, Pang et al. [2021] detected stenotic regions, including prior coronary artery displacement information. They used a ResNet50 model trained on ImageNet as a backbone of the object detector network. Only the last full connection layer and SoftMax layer of ResNet-50 were fine-tuned. Later, Danilov et al. [2021] evaluated different object detection network configurations, including a Single Shot multi-box Detector (SSD) [Liu et al., 2016], Faster Region-Based Convolutional Neural Networks (Faster-RCNN) [Ren et al., 2015], and Region-based Fully Convolutional Networks (R-FCN) [Dai et al., 2016]. In their networks, different backbones networks have been employed, such as MobileNet-v2 [Sandler et al., 2018], ResNet (50, 101) [He et al., 2016], and Inception-v4 [Szegedy et al., 2017]. To train the above-mentioned backbone networks, they used models pre-trained on the Common Objects in Context (COCO) 2017 dataset [Lin et al., 2014]; thereafter, a fine-tuning strategy for the whole network was applied.

Instead of detecting and classifying a stenosis case as an object in the XCA image, it can be more suitable to classify the whole image as a single class into the stenosis and no stenosis. Therefore, Cong et al. [2019] put forward a previous step to separate the images by angle view. Then, the candidate frame selection is performed by a CNN composed of a pre-trained Inception-v3 as a feature extractor feeding a Bi-directional Long-Short-Term Memory (BiLSTM). Consequently, an independent pre-trained on the ImageNet dataset Inception-v3 network classified these filtered images into stenosis and non-stenosis classes. In this case, all the layers of the network were fine-tuned.

However, the previous methods require the whole angiographic test and assume that a single stenosis region is present in the image. Another approach to solving this task is using a patch-based classification network. In this way, the full-size XCA image generates n-patches to be classified as positive or negative stenosis cases. This patch-based approach can be seen as each patch representing a labeled object. In this context, Antczak and Liberadzki [2018] employed a VGG-based model of only five convolutional layers to classify XCA image patches into the stenosis and no stenosis categories. In addition, a pre-training strategy was performed by synthetic data, consisting of a Bezier-based generative model to improve the results. Subsequently, the pre-trained model was fully fine-tuned using a subset of negative data to maintain a balanced training and testing dataset.

3.3 | Transfer Learning and Network Cut

A novel method is proposed to detect coronary artery stenosis automatically in XCA images, employing three pre-trained CNNs (VGG16, ResNet50, and Inception-v3) on the ImageNet dataset. A transfer learning strategy is performed from these models. The method incorporates a network-cut approach, where after a pooling layer, the model can be trimmed and connected to the custom classifier layers (top dense layer). Hence, the number of transferred layers and parameters of each architecture decreases. Figure 3.2 shows the network-cut approach framework.

3.3.1 | Network Cut

The optimal cut and fine-tuned layers were selected by minimizing the loss target function ξ_{trg} such as:

$$\arg\min_{\boldsymbol{\theta}^{(i)}} \xi_{trg}\left(f_{src}(\boldsymbol{\theta}^{(1)}, \cdots, \boldsymbol{\theta}^{(m)}, \boldsymbol{\theta}^{(h)})\right), \quad \text{s.t.} \quad m \leq n \quad \forall i \in S \tag{3.5}$$

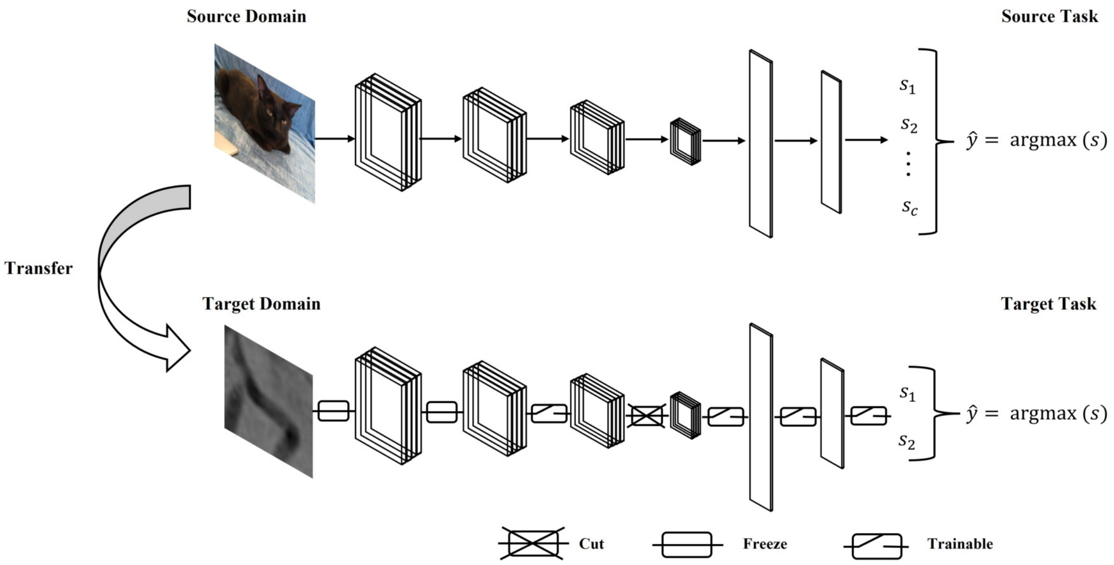


Figure 3.2: *Transfer Learning and Network Cut. The top dense layers are set as trainable, and the last convolutional block is trimmed from the network. Meanwhile, the new last convolutional block is set to trainable and the bottom ones remain frozen.*

where m is the position of the cut-layer, n is the total convolutional layers of the source model, and S is the subset of layers to be fine-tuned. The cut layer used at the end of the fine-tuned layer of the pre-trained network is carried out in descending and progressive ways. Accordingly, four characteristic behaviors can be distinguished from these network configurations:

1. Feature extractor: The weights of the pre-trained convolutional layers remain fixed, and only the dense layers are trained with the target task.

2. Fine-tuning: The whole network is fine-tuned in a layer-wise manner from top to low-level blocks, where the blocks of convolutional layers pass to trainable on a descendent way.

3. Network-cut and feature extractor: the network is trimmed on an early convolutional block, retaining its weights. Then only the last dense layers are trained with the target task.

4. Network-cut and fine-tuning: the network is cut up after a pooling layer, but a fine-tuning process is carried out in the remaining bottom layers as well as the top dense layers optimizing the target task.

In such a way, the VGG16 was divided into four cut blocks: one is set after the first convolutional block (*i.e.,* first convolution and pooling layer), and the remainder after each double convolutional block. The ResNet50 was also divided into four cut blocks: one before each residual block. In this manner, the first cut block only maintains the first convolutional, batch normalization, and pooling operation. Finally, for the Inception-v3, three cut blocks were set: one before each type of Inception block (A, B, and C). In general, if the backbone model has a cut block in the first block, only the first convolutional layer is maintained. On the other hand, if the cut block is set in the fourth block, the last block of layers is removed.

3.4 | Results and Discussion

3.4.1 | Implementation Details

The fine-tuning process employs the Stochastic Gradient Descent with Momentum (SGDM) optimizer [Qian, 1999] with a learning rate of 10^{-3} and a momentum of 0.9. The model was trained with a batch size of 32 for 100 epochs minimizing the Cross-Entropy Loss. If the validation loss is not improving during 20 epochs, the learning rate is decreased by a factor of $\sqrt{0.1}$. The model was implemented using the PyTorch framework, and the experiments ran on Google's cloud servers, including a Tesla P4 GPU with 2560 CUDA cores and 8 GB of RAM.

3.4.2 | Ablation Study

An ablation study is conducted on an ideally balanced dataset employing the ADSS dataset (see Section 1.2). From the original ADSS dataset, a balanced subset was extracted such that 250 real XCA image patches of size 32×32 pixels in grayscale, with 125 patches identified with stenosis and 125 with no stenosis. The image patches were resized to 96×96 pixels for the Inception-v3 to fit into the default model configuration. Moreover, the images were converted to 3-channel images, cloning the grayscale image into the other two channels. Additionally, the z-score normalization was performed, changing the image range to $[0, 1]$ and applying the ImageNet mean $\mu = [0.485, 0.456, 0.406]^\intercal$ and standard deviation $\sigma = [0.229, 0.224, 0.225]^\intercal$. The dataset was stratified into a fine-tuning (training) set, and testing set, each with 125 images. The training subset was additionally partitioned into 5-fold for cross-validation.

From the ablation study for the VGG16, it can be seen in Table 3.1 that the best model configuration (with the lowest validation loss) was the VGG16 without the last convolutional block (the cut block is the fourth) with all the parameters fine-tuned. This model required 5 million fewer parameters than the original (vanilla) VGG16. In the case of the ResNet50, Table 3.2 shows that the optimal model configuration was cutting the fourth residual block and fine-tuning the remaining residual blocks but keeping the first convolutional layer idle (block 0). This model cut reduced the vanilla ResNet50 size by 2.75x (*i.e.,* by 15 million parameters). Finally, applying the network cut to the Inception-v3, the best configuration was a trimmed version, without the InceptionC blocks (the last inception block), and applying a fine-tuning to the remaining of the model, as shown in Table 3.3. This configuration required 13.66 million fewer parameters than the vanilla Inception-v3, with a total of 27.49 million.

The three models had poor performance in minimizing the validation loss when the full model was frozen, and only the head layers (the dense layers) were trained. Ergo, when the pre-trained model acts as a feature extractor. This phenomenon was also observed when the models were trimmed. Scilicet, fine-tuning only the dense layers is not optimal. Similarly, when the models only maintained the first convolutional block and the rest of the model was trimmed, the model reached higher losses than keeping an extra convolutional, residual, or inception block respectively.

Therefore, the model configuration that minimizes each validation loss for the vanilla VGG16, ResNet50, and Inception-v3 was selected as the default model for subsequent comparison. Henceforth, Trim VGG16, Trim ResNet50, and Trim Inception-v3, respectively.

3.4.3 | Stenosis Classification Performance Comparison

The proposed network-cut approach aims to efficiently fine-tune a pre-trained network for stenosis detection employing a small and unbalanced dataset. Hence, the performance of the Trim VGG16, ResNet50, and Inception-v3 models were evaluated on the public dataset: DSSS (see Section 1.2). To sum up, the DSSS only contains 1,519 XCA image patches of size 32×32, where only 125 are positive cases of stenosis and 1,394 negative cases. The dataset was split into an 80:20 training/test partition and employed 5-fold cross-validation.

Table 3.4 shows the classification results comparing the vanilla version of the networks and their corresponding trained-from-scratch performance with the proposed fine-tuning network-cut approach. This table shows that fine-tuning and cutting the models boost the performance concerning their vanilla configuration. Also, the Trimmed models achieved higher classification rates when trained from scratch, except the Trim VGG16, which showed poor performance (*i.e.,* made all predictions as negative cases of stenosis). Particularly fine-tuned, the Trim VGG16 reached the best overall accuracy, specificity, precision, and F_1 score with 0.9717, 0.9900, 0.8726,

Table 3.1: *VGG16 ablation study for transfer learning and network cut. The optimal configuration is selected such that the mean validation loss is minimized. The validation loss standard deviation is also specified with ±. The number of parameters is given in Millions (M).*

Cut block				**Fine-tuned block**					**Best**	**Parameters**	**Trainable**
4	3	2	1	4	3	2	1	0	**Validation Loss**	**[M]**	**Parameters [M]**
✗	✗	✗	✗	✓	✓	✓	✓	✓	0.2044 (± 0.1363)	134.26	134.26
				✓	✓	✓	✓	✗	0.2015 (± 0.1160)		134.23
				✓	✓	✓	✗	✗	0.2364 (± 0.1405)		134.00
				✓	✓	✗	✗	✗	0.2358 (± 0.1129)		132.53
				✓	✗	✗	✗	✗	0.2683 (± 0.0993)		126.63
				✗	✗	✗	✗	✗	0.2911 (± 0.0778)		119.55
✓	✗	✗	✗	—	✓	✓	✓	✓	**0.0994 (± 0.1626)**	127.19	127.19
				—	✓	✓	✓	✗	0.1027 (± 0.1629)		127.15
				—	✓	✓	✗	✗	0.1049 (± 0.1627)		126.93
				—	✓	✗	✗	✗	0.1109 (± 0.1595)		125.45
				—	✗	✗	✗	✗	0.1112 (± 0.1493)		119.55
✓	✓	✗	✗	—	—	✓	✓	✓	0.1149 (± 0.1824)	31.63	31.63
				—	—	✓	✓	✗	0.1149 (± 0.1816)		31.59
				—	—	✓	✗	✗	0.1140 (± 0.1818)		31.37
				—	—	✗	✗	✗	0.1140 (± 0.1819)		29.89
✓	✓	✓	✗	—	—	—	✓	✓	0.2173 (± 0.1246)	7.74	7.74
				—	—	—	✓	✗	0.2161 (± 0.1251)		7.70
				—	—	—	✗	✗	0.2106 (± 0.1245)		7.48
✓	✓	✓	✓	—	—	—	—	✓	0.2942 (± 0.1836)	1.91	1.91
				—	—	—	—	✗	0.3193 (± 0.1964)		1.87

***Table 3.2:** ResNet50 ablation study for transfer learning and network cut. The optimal configuration is selected such that the mean validation loss is minimized. The validation loss standard deviation is also specified with ±. The number of parameters is given in Millions (M).*

Cut block				**Fine-tuned block**					**Best**	**Parameters**	**Trainable**
4	3	2	1	4	3	2	1	0	**Validation Loss**	**[M]**	**Parameters [M]**
✗	✗	✗	✗	✓	✓	✓	✓	✓	0.5913 (± 0.0110)	23.51	23.51
				✓	✓	✓	✓	✗	0.5614 (± 0.0363)		23.50
				✓	✓	✓	✗	✗	0.5198 (± 0.0506)		23.29
				✓	✓	✗	✗	✗	0.5190 (± 0.0412)		22.07
				✓	✗	✗	✗	✗	0.5432 (± 0.0301)		14.97
				✗	✗	✗	✗	✗	0.5533 (± 0.0313)		0.004
✓	✗	✗	✗	—	✓	✓	✓	✓	0.1298 (± 0.0889)	8.55	8.55
				—	✓	✓	✓	✗	**0.1261 (± 0.1113)**		8.54
				—	✓	✓	✗	✗	0.1500 (± 0.1035)		8.32
				—	✓	✗	✗	✗	0.1514 (± 0.1034)		7.10
				—	✗	✗	✗	✗	0.1546 (± 0.0969)		0.002
✓	✓	✗	✗	—	—	✓	✓	✓	0.1558 (± 0.1707)	1.45	1.45
				—	—	✓	✓	✗	0.1541 (± 0.1802)		1.44
				—	—	✓	✗	✗	0.1510 (± 0.1823)		1.22
				—	—	✗	✗	✗	0.1475 (± 0.1864)		0.001
✓	✓	✓	✗	—	—	—	✓	✓	0.2967 (± 0.1485)	0.2259	0.2259
				—	—	—	✓	✗	0.3361 (± 0.1658)		0.2163
				—	—	—	✗	✗	0.3707 (± 0.1495)		0.0005
✓	✓	✓	✓	—	—	—	—	✓	0.4245 (± 0.0788)	0.0096	0.0096
				—	—	—	—	✗	0.4428 (± 0.0749)		0.0001

Table 3.3: *Inception-v3 ablation study for transfer learning and network cut. The optimal configuration is selected such that the mean validation loss is minimized. The validation loss standard deviation is also specified with ±. The number of parameters is given in Millions (M).*

Cut block			Fine-tuned block				Best	Parameters	Trainable
3	2	1	3	2	1	0	Validation Loss	[M]	Parameters [M]
✗	✗	✗	✓	✓	✓	✓	0.3011 (± 0.1210)	41.15	41.15
			✓	✓	✓	✗	0.3273 (± 0.1279)		40.54
			✓	✓	✗	✗	0.2791 (± 0.1200)		36.96
			✓	✗	✗	✗	0.3552 (± 0.1050)		13.66
			✗	✗	✗	✗	0.4351 (± 0.0375)		0.003
✓	✗	✗	—	✓	✓	✓	**0.1520 (± 0.0982)**	27.49	27.49
			—	✓	✓	✗	0.2664 (± 0.1080)		26.88
			—	✓	✗	✗	0.2913 (± 0.1094)		23.30
			—	✗	✗	✗	0.3437 (± 0.0853)		0.003
✓	✓	✗	—	—	✓	✓	0.1804 (± 0.1314)	4.18	4.18
			—	—	✓	✗	0.1913 (± 0.1214)		3.58
			—	—	✗	✗	0.2576 (± 0.1019)		0.002
✓	✓	✓	—	—	—	✓	0.2924 (± 0.1447)	0.6065	0.6065
			—	—	—	✗	0.3353 (± 0.0866)		0.00007

0.8168, respectively, and the second-best sensitivity with 0.7680. The fine-tuned Trim Inception-v3 achieved the best sensitivity (0.7760), competitive accuracy (0.9651), and specificity (0.9821). The fine-tuned Trim ResNet50 also attained the best specificity (0.9900), competitive accuracy (0.9612), and precision (0.8551). It is worth noticing that although the Trim VGG16 obtained four of the five higher metrics, it is the model with the higher number of parameters, up to 15.8x more than Trim ResNet50 and 4.7x more than Trim Inception-v3. Moreover, Vanilla and Trim VGG16 trained from scratch showed lower performance.

3.4.4 | Class Activation Maps Visualization

A class activation map was obtained to visually explain the areas where the image features have the most significant impact on prediction. Figure 3.3 shows the case examples of predictions using the Trimmed versions of the VGG16, ResNet50, and

Table 3.4: *Transfer learning and network cut classification performance. The mean and standard deviation of each metric are shown.*

Model	**Pretrained**	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**F$_1$-score**
Vanilla VGG16	✗	0.9270 ± 0.0113	0.2000 ± 0.2479	0.9921 ± 0.0102	0.2800 ± 0.3436	0.2317 ± 0.2844
	✓	0.9691 ± 0.0039	0.7440 ± 0.0650	0.9892 ± 0.0060	0.8693 ± 0.0633	0.7975 ± 0.0283
Trim VGG16	✗	0.9178 ± 0.0000	0.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000
	✓	**0.9717** ± 0.0034	0.7680 ± 0.0299	**0.9900** ± 0.0014	**0.8726** ± 0.0187	**0.8168** ± 0.0235
Vanilla ResNet50	✗	0.9204 ±0.0092	0.2480 ±0.1568	0.9806 ±0.0058	0.4968 ±0.1073	0.3162 ±0.1597
	✓	0.9520 ± 0.0049	0.5520 ± 0.0531	0.9878 ± 0.0018	0.8016 ± 0.0285	0.6528 ± 0.0440
Trim ResNet50	✗	0.9349 ± 0.0092	0.4800 ± 0.0669	0.9756 ± 0.0062	0.6402 ± 0.0809	0.5471 ± 0.0674
	✓	0.9612 ± 0.0120	0.6400 ± 0.1734	**0.9900** ± 0.0042	0.8551 ± 0.0400	0.7154 ± 0.1320
Vanilla Inception-v3	✗	0.9507 ± 0.0075	0.5840 ± 0.0862	0.9835 ± 0.0058	0.7636 ± 0.0636	0.6581 ± 0.0630
	✓	0.9625 ± 0.0077	0.7040 ± 0.0967	0.9857 ± 0.0051	0.8175 ± 0.0515	0.7525 ± 0.0598
Trim Inception-v3	✗	0.9599 ± 0.0032	0.7280 ± 0.0854	0.9806 ± 0.0043	0.7746 ± 0.0247	0.7462 ± 0.0409
	✓	0.9651 ± 0.0064	**0.7760** ± 0.0320	0.9821 ± 0.0060	0.7988 ± 0.0605	0.7861 ± 0.0357

Inception-v3, respectively. In the GradCAM images, red tones stand for high-attention regions, and purple for low-attention ones. Bellow each image, the probability of stenosis is set. For values higher than 0.5, the models classify as stenosis cases. As one can see, Trim VGG16 and Trim Inception-v3 presented isolated horizontal high-attention regions; in most cases, the GradCAM image showed constant attention (purple tones).

Meanwhile, Trim ResNet50 obtained high-attention regions over blood vessel areas, identifying prominent features (regions in red tones). From this last case, when a false positive was predicted (subfigure (b), seventh image), the GradCAM retrieved high-attention regions in background areas. On the other hand, when a false negative was presented (subfigure (b), fourth image), the image presented non-highlighted regions in red tones. Thus, the model was not able to learn specific rich feature locations.

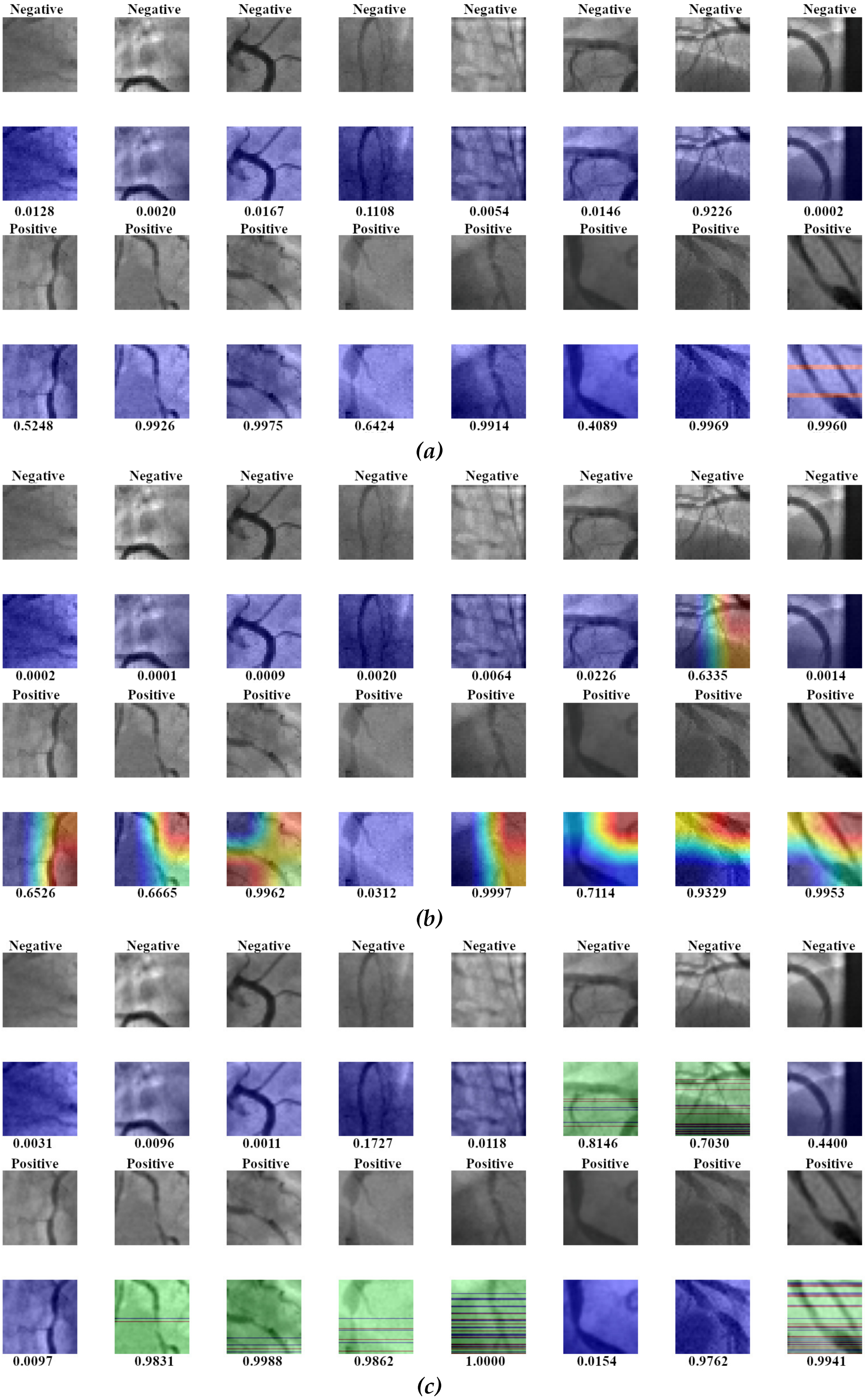
These images provide valuable information about the localization of features that significantly impact the prediction stage.

3.5 | Conclusion

This chapter introduced a network-cut and fine-tuning method for stenosis detection in XCA images. The extensive numerical experiments were implemented based on 20 different setups for the pre-trained (on the ImageNet dataset) with three different fine-tuning strategies for the VGG16, ResNet50, and Inception-v3 networks. The optimal cut and fine-tuned layers were selected by minimizing the loss function. They have demonstrated that employing a pre-trained network on a limited and unbalanced XCA dataset performs efficiently for stenosis detection. The results of the pre-trained networks showed that the optimal model configuration for the VGG16, ResNet50, and Inception-v3, required cutting the last convolutional, residual, or Inception block, respectively.

Moreover, the best loss was achieved when fine-tuning was applied in all remained layers for VGG16 and Inception-v3, and froze the first convolutional layer for the ResNet50. The proposed scheme allowed an accuracy, sensitivity, specificity, precision, and F_1-score improvement concerning the vanilla pre-trained networks and with the configurations trained from scratch. Furthermore, it allowed us to reduce the network complexity regarding parameters, where the Trim ResNet50 only required 8.55M of parameters compared with 27.49M for Trim Inception-v3 and 127.19M for Trim VGG16. Besides, a class activation map using the GradCAM technique was performed to provide a deep learning-based visual explanation for the areas where the image features significantly impact prediction. This visual study verified that Trim ResNet50 retrieved the best gradient maps over the image, *i.e.,* with high-attention regions in blood vessel pixels, contributing to computer-aided diagnosis in cardiology.

The classification results reached in the proposed network cut approach could be further improved by analyzing the family of residual networks, which achieved the best performance when trained from scratch and very competitive classification results with fewer parameters.



(a)

(b)

(c)

Figure 3.3: *Fine-tuning and the network cut GradCAM. (a) Trim VGG16, (b) Trim ResNet50, and (c) Trim Inceptionv3.*

4

Hybrid Classical-Quantum Convolutional Neural Networks for Stenosis Detection

> *"Nothing is real unless it is observed."*
>
> — John Gribbin, *In Search of Schrödinger's Cat: Quantum Physics and Reality*

4.1 | Mathematical Foundations

The classical computational unit is the bit, which can take one of two states for computation, either 0 or 1. On the other hand, the corresponding unit of quantum computing is the *qubit*. Instead of having a scalar value as classical bits, a qubit can be any linear combination (*superpositions*) of the *computational basis states* $|0\rangle = [1, 0]^\intercal$ and $|1\rangle = [0, 1]^\intercal$ (*i.e.*, any state ψ is written in Dirac notation $|\psi\rangle$). Hence, a *qubit* is described by a two-dimensional Hilbert space, whose state can be expressed as:

$$|\psi\rangle = \alpha\, |0\rangle + \beta\, |1\rangle\,, \tag{4.1}$$

where α and β are two complex numbers that satisfy $|\alpha|^2 + |\beta|^2 = 1$.

By writing a quantum state in polar form, *i.e.*, $\alpha = a\mathrm{e}^{\mathrm{i}\theta}$ and $\beta = b\mathrm{e}^{\mathrm{i}\varphi}$, where $\mathrm{e}^{\mathrm{i}\theta}$ is the global phase and $\mathrm{e}^{\mathrm{i}\phi}$ is the relative phase, with $\phi = (\varphi - \theta)$. In quantum mechanics, the global phase has no physical meaning; in this way, it can be omitted. Thus, Equation (4.1) can be written as follows:

$$|\psi\rangle = a\,|0\rangle + b\,\mathrm{e}^{\mathrm{i}\phi}\,|1\rangle\,. \tag{4.2}$$

Since $a^2 + b^2 = 1$ needs to be fulfilled, $a = \cos\left(\frac{\theta}{2}\right)$ and $b = \sin\left(\frac{\theta}{2}\right)$ have the same relationship. In such a way, a single-qubit state is parametrized by two angles (θ and ϕ)

as follows:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \mathrm{e}^{\mathrm{i}\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle\,. \tag{4.3}$$

Therefore, each qubit state vector and operation can be represented in 3D space (Bloch sphere), as illustrated in Figure 4.1.

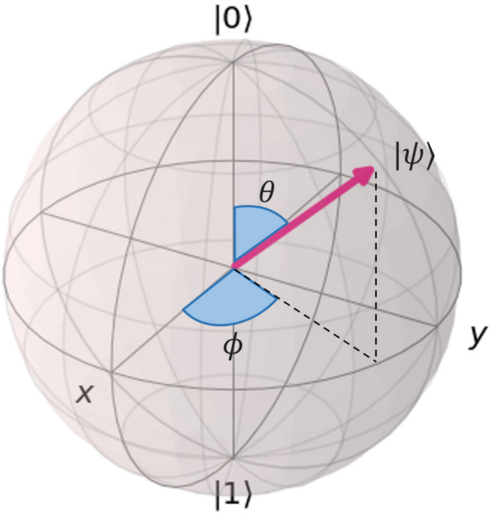


Figure 4.1: *3D representation of the state of a single qubit. θ is inclination angle from $+z$ direction and ϕ is azimuth from $+x$ direction.*

A multiple-qubit state consisting of n *unentangled* qubits can be represented as the tensor product ($\otimes$) of the states of the individual qubits given by:

$$|\Psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes \cdots \otimes |\psi_n\rangle\,. \tag{4.4}$$

The state can also be written using a bit-string representation. For instance, the 3-qubit state $|0\rangle \otimes |0\rangle \otimes |1\rangle$ is $|100\rangle$ (from right to left).

Notwithstanding, if the n qubit state cannot be decomposed into the tensor product of individual states, the qubits are considered *entangled*. If a pair of qubits are entangled, the measurement on one qubit instantaneously affects the other.

In order to manipulate single or multiple qubits, a quantum computer employs quantum gates, represented by a unitary matrix $\mathbf{U}$, such that $\mathbf{U}\mathbf{U}^{\dagger} = \mathbf{U}^{\dagger}\mathbf{U} = \mathbb{I}_n$, where $\mathbb{I}_n$ is the identity matrix in $\mathbb{R}^n$ and $\mathbf{U}^{\dagger}$ is the conjugate transpose. In this manner, a sequence of quantum gates forms a *quantum circuit* $\mathcal{C}$ defined as:

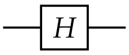
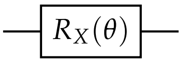
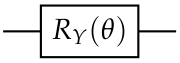
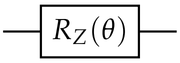
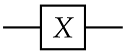
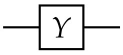
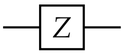
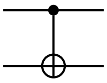
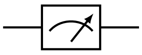
$$\mathcal{C}(\boldsymbol{\theta}) = \prod_{i=1}^{n} \mathbf{V}_i(\boldsymbol{\theta}_i)\mathbf{W}_i, \tag{4.5}$$

where $\mathbf{W}_i$ are un-parameterized gates (*e.g.*, CNOT gates) and $\mathbf{V}_i(\boldsymbol{\theta}_i)$ are a set of q quantum variational gates (*e.g.*, one of the rotation gates R_X, R_Y, R_Z) such as:

$$\mathbf{V}_i(\boldsymbol{\theta}_i) = \bigotimes_{j=1}^{q} R^{j,i}(\theta_{j,i}), \tag{4.6}$$

where $R^{j,i}$ is the j-th rotation gate acting on the i-th qubit, and $\theta_{j,i}$ its respective rotation angle. Table 4.1 illustrates the most frequently used quantum gates, circuit symbols, and mathematical expressions.

Table 4.1: *Graphical and mathematical notation of the most employed quantum gates.*

Name	Circuit	Notation
Hadamard		$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
X-Rotation		$R_X(\theta) = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$
Y-Rotation		$R_Y(\theta) = \begin{bmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$
Z-Rotation		$R_Z(\theta) = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$
Pauli-X		$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y		$\sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z		$\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
CNOT		$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Measurement		—

4.2 | Related Work

In recent years, a new paradigm in the context of Quantum Machine Learning [Biamonte et al., 2017] has appeared, which is paramount because it is founded on the power and properties of quantum computing. In the literature, a few works have explored including a quantum component in a CNN. Konar et al. [2020] proposed a

Quantum-inspired Neural Network (QIS-Net) to segment brain magnetic resonance images automatically. The quantum network comprises three layers of quantum neurons (input layer, intermediate layer, and output layer), introducing a Multi-level Sigmoid activation function at the last activation function. The input layer of the QIS-Net architecture maps each normalized image information (pixel intensities) into a quantum state for subsequent processing of the intermediate and output layers. The results showed good accuracy and Dice similarity scores concerning classical neural networks for image segmentation. However, the QIS-Net presented one main drawback, it requires a qubit per pixel, which results prohibitive because quantum devices (and simulators) have a limited number of qubits.

To deal with the aforementioned pitfall, a hybrid model has been proposed by Iyer et al. [2020], that classified pigmented skin lesions employing a variational classifier, extracting a feature descriptor from a classical neural network that feeds a 2-qubit quantum circuit to obtain the two predicted labels: melanoma or melanocytic nevi. Similarly, Sleeman et al. [2020] introduced a hybrid method connecting a classical convolutional autoencoder to a quantum Restricted Boltzmann Machine. This hybrid autoencoder algorithm showed competitive results using two representative datasets, the Modified National Institute of Standards and Technology (MNIST) [LeCun et al., 1998] and Fashion-MNIST [Xiao et al., 2017]. Henderson et al. [2020] introduced a new type of quantum convolution layer that transforms the data using several quantum circuits seen as filters, like a classical convolutional layer. Specifically, the quantum layer is the first transformation, and the remaining architecture on top is like a classical CNN. Such a work showed that the CNN disposing of a quantum layer had higher test accuracy using the MNIST [LeCun et al., 1998] dataset.

Moreover, Mari et al. [2020] introduced a hybrid transfer learning framework. In this method, a quantum circuit is connected at the top of a pre-trained classical CNN focused on image recognition (Hymenoptera subset of ImageNet and Canadian Institute For Advanced Research (CIFAR)-10 datasets Krizhevsky [2009]). Herein, the classical layers and the quantum circuit parameters were jointly trained to simplify the quantum encoding and decoding phases. In such a way, Acar and Yilmaz [2021] employed this hybrid transfer learning framework to perform COVID-19 detection by using a few training images, achieving higher accuracy rates than traditional methods. In summary, such a hybrid network evidenced a solid potential of quantum computing for image classification in small training datasets; this was the primary assumption in designing this study.

4.3 | Hybrid Classical-Quantum Convolutional Neural Network

This chapter proposes a Hybrid Classical-Quantum Network (H-CQN) for stenosis detection in XCA images. This hybrid approach was first introduced by Mari et al. [2020], consisting of a classical network, henceforth a *backbone network*, an embedding layer, a Quantum Network (QN), a decoding layer, and a classical SoftMax layer, as shown in Figure 4.2. Beneath the discoveries of the previous chapter, where the best network was a trimmed ResNet50 as a backbone network, the family of the ResNet was explored, including smaller models, such as ResNet18 and ResNet34. Furthermore, since the QN size depends on the number of features to be processed, a Distributed Variational Quantum Circuit is proposed to keep small quantum circuits.

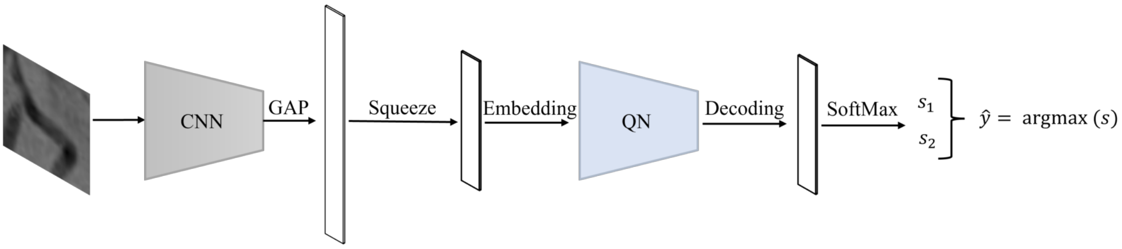


Figure 4.2: *Proposed Hybrid Classical-Quantum Network. This network comprises five main components: a pre-trained classical network, an embedding layer, a Quantum Network, a decoding layer, and a classical SoftMax layer.*

4.3.1 | Quantum Network

Analogously to a classical layer, a quantum circuit contains a set of quantum gates whose parameters can be learned to extract discriminant features from an input. Thereunto, a *Quantum Layer* can be defined as:

$$\mathcal{L}(|\Psi\rangle, \boldsymbol{\theta}) : |\Psi\rangle \rightarrow |\Psi'\rangle = \mathcal{C}(\boldsymbol{\theta})|\Psi\rangle, \tag{4.7}$$

where $|\Psi\rangle$ and $|\Psi'\rangle$ are the input and output quantum state, respectively, $\mathcal{C}$ is the quantum circuit defined by Equation (4.5), and $\boldsymbol{\theta}$ is the set of parameters of the layer.

Since the QN size depends on the number of qubits required by the input state, using a more flexible quantum network is convenient. For such a reason, r-independent Variational Quantum Circuit (VQC) operates a Distributed Variational Quantum Circuit (D-VQC) that distributes the data among them. Through this strategy, there is no tangible wire connecting the circuits. Following this idea, let n be the size of the input quantum state $|\Psi\rangle$, a quantum layer with D-VQCs is formally described as:

$$\mathcal{L}(|\Psi\rangle, \boldsymbol{\theta}) : |\Psi\rangle \rightarrow |\Psi'\rangle = \biguplus_{i}^{r} \mathcal{C}_i(\boldsymbol{\theta}_i)\,|\Psi_i\rangle\,, \tag{4.8}$$

where $\biguplus$ represents the concatenation operator that stacks the output state of each independent VQC in a single vector. Notice that each sub-state $|\Psi_i\rangle$ has $\frac{n}{r}$ qubits, as seen in Figure 4.3, where the inputs are split into two independent 2-qubit VQC instead of a single 4-qubit quantum circuit per layer.

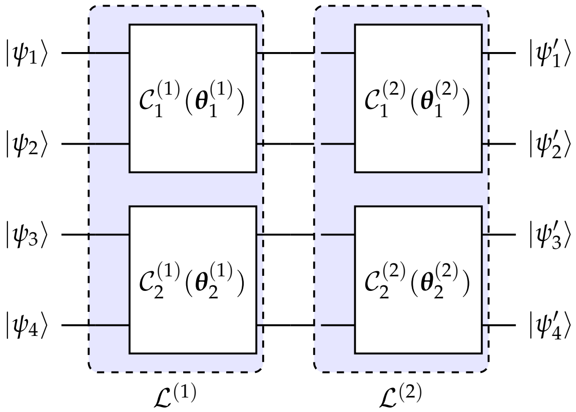

$|\psi_1\rangle$
$|\psi_2\rangle$
$|\psi_3\rangle$
$|\psi_4\rangle$
$\mathcal{C}_1^{(1)}(\boldsymbol{\theta}_1^{(1)})$
$\mathcal{C}_2^{(1)}(\boldsymbol{\theta}_2^{(1)})$
$\mathcal{C}_1^{(2)}(\boldsymbol{\theta}_1^{(2)})$
$\mathcal{C}_2^{(2)}(\boldsymbol{\theta}_2^{(2)})$
$|\psi'_1\rangle$
$|\psi'_2\rangle$
$|\psi'_3\rangle$
$|\psi'_4\rangle$
$\mathcal{L}^{(1)}$
$\mathcal{L}^{(2)}$

Figure 4.3: *Quantum Network employing layers with a D-VQC. Each quantum layer, $\mathcal{L}^{(q)}$, is divided into two independent quantum circuits $\mathcal{C}_i^{(q)}$ of 2-qubits, each parameterized by their corresponding weights $\boldsymbol{\theta}_i^{(q)}, i \in (1,2)$. ψ_i is the i-th input qubit, and ψ'_i its corresponding output qubit.*

Particularly, two types of VQCs were employed: a 2-qubits and a 4-qubits circuit. Therefore, a VQC of q_i-qubits is defined as:

$$\mathcal{C} = \mathbb{K} \bigotimes_{k=1}^{q_i} R_y(\boldsymbol{\theta}_k), \tag{4.9}$$

where $\mathbb{K}$ is obtained from the entangling unitary operation made of CNOT gates over each consecutive couple of qubits. Thereby, $\mathbb{K}$ is defined as follows:

$$\mathbb{K} = \mathbb{I}\{k, k+1\} \otimes CNOT, \quad \forall k \in q_i \tag{4.10}$$

where $\mathbb{I}$ is the identity matrix, and the indices $\{k, k+1\}$ denote the consecutive rows of the matrix, indicating the qubits to perform the CNOT gate. Figure 4.4 shows the architecture details of the 2-qubit and 4-qubit VQCs analyzed in this study.

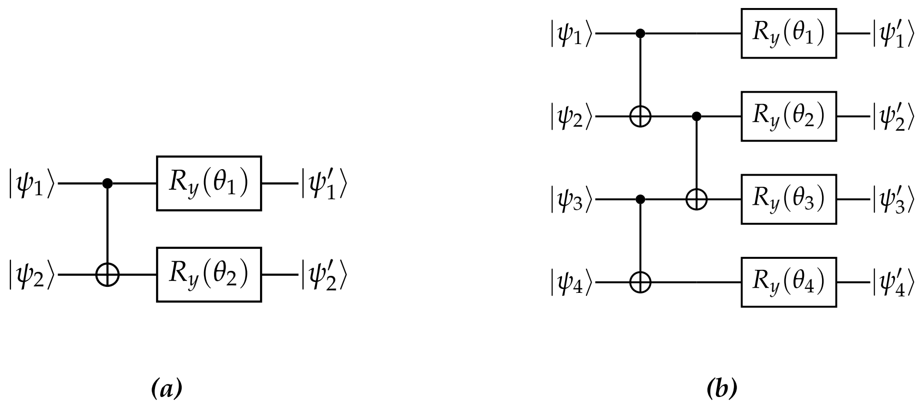


Figure 4.4: *VQC configurations used in the proposed quantum network architecture. a) 2-qubit VQC. b) 4-qubit VQC.*

4.3.2 | Encoding and Decoding Layers

In order to operate a quantum circuit and, consequently, a quantum network, it is paramount to *encode* classical data into quantum data. For example, to map classical data onto a quantum network, a real vector $\mathbf{x} \in \mathbb{R}^n$ must be embedded in a quantum state $|\Psi_{\mathbf{x}}\rangle$.

Let $E(\mathbf{x})$ be the encoding operator; hence, the encoded quantum state is obtained by:

$$\mathcal{E} : \mathbf{x} \rightarrow |\Psi_{\mathbf{x}}\rangle = E(\mathbf{x})\,|\Psi_0\rangle\,, \tag{4.11}$$

where $|\Psi_0\rangle$ is an initial state (*e.g.*, the ground state $|0\rangle^{\otimes n}$). The encoding determines how many qubits are required in the quantum circuit. Different encoding strategies can be found, such as threshold encoding [Henderson et al., 2020], angle encoding [Stoudenmire and Schwab, 2016], and amplitude encoding [LaRose and Coyle, 2020]. Given its simplicity, angle encoding is the most widely used encoding approach, where single-qubit rotation gates encode the classical input. Each element of the input determines the angle of the rotation gate (*e.g.*, an R_Y rotation gate). As such, this approach requires n qubits to encode n input variables and can be defined as:

$$|\Psi_{\mathbf{x}}\rangle = \bigotimes_{i=1}^{n} R(x_i)\,|\Psi_0\rangle\,, \tag{4.12}$$

where R is a rotation matrix, and x_i is the i-th element of $\mathbf{x}$.

The encoding step is usually applied to the initial state of the Hadamard gate H, leading to a uniform superposition state as follows:

$$|\Psi_{\mathbf{x}}\rangle = \bigotimes_{i=1}^{n} R(\theta_i)(H\,|\Psi_0\rangle). \tag{4.13}$$

Notice that θ_i is the rotation angle of the gate, and $\mathbf{x}$ is the output from an arbitrary pooling layer of a CNN; hence, to make a proper rotation, a base amplitude was defined and scaled by a normalized value of x_i. Accordingly, θ_i was obtained by:

$$\theta_i = \frac{1}{2}\pi \hat{x}_i, \tag{4.14}$$

where $\hat{x}_i \in \hat{\mathbf{x}}$ is computed as:

$$\hat{\mathbf{x}} = \tanh\left(\kappa \frac{\mathbf{x}}{||\mathbf{x}||_2}\right), \tag{4.15}$$

where κ is a regularization parameter to control the saturation interval of the hyperbolic tangent (tanh) function [Latha and Thangasamy, 2011], and $\mathbf{x}$ is the feature vector to be encoded. By doing so, the feature vector lies on a hypersphere of radius κ. Besides, the value of κ can be trained within the optimization process.

Finally, in the *decoding stage*, $m \leq n$ qubits are measured in the output quantum state $|\Psi'\rangle$ by a given local observable A, for instance, the Pauli operator σ_Z. Variance and expected value are examples of the most common observables. Measurements can be made globally, where all qubits are measured, or locally, where only a few qubits are measured individually or in pairs. In such a way, the decoded data can be obtained through repeated measures such as:

$$\mathcal{M} : |\Psi_{\mathbf{y}}\rangle \rightarrow \mathbf{y} =< \Psi_{\mathbf{y}}|A^{\otimes m}|\Psi_{\mathbf{y}} > . \tag{4.16}$$

4.3.3 | Classical-Quantum Network

Since the input and output of a quantum network are classical values, all the encoding, transformation, and decoding can be defined as:

$$\mathcal{Q}(\mathbf{x};\boldsymbol{\theta}) : \mathbf{x} \in \mathbb{R}^n \rightarrow \mathbf{y} \in \mathbb{R}^m = \mathcal{M} \circ \mathcal{C} \circ \mathcal{E}, \tag{4.17}$$

where the parameters $\boldsymbol{\theta}$ can be updated using optimization algorithms.

In a deep learning context, $\mathcal{Q}(\mathbf{x};\boldsymbol{\theta})$ can be seen as a layer in a deep neural network [Henderson et al., 2020]. Furthermore, it can be embedded in a classical CNN as follows:

$$\mathcal{N} = \mathcal{L}(\mathbf{x}^{(l)}, \boldsymbol{\theta}^{(l)}) \circ \mathcal{L}(\mathbf{x}^{(l-1)}, \boldsymbol{\theta}^{(l-1)}) \circ \cdots \circ \mathcal{L}(\mathbf{x}^{(1)}, \boldsymbol{\theta}^{(1)}), \tag{4.18}$$

where each layer $\mathcal{L}(\mathbf{x}^{(i)}, \boldsymbol{\theta}^{(i)})$ is a classical or a quantum layer.

Since real quantum computers provide minimal qubit circuits (*e.g.*, two or four-qubit systems), using the raw images to fit directly into a quantum network remains intractable. As such, in this work, the QN was taken as an architectural unit designed to improve the final feature representation of a classical CNN. Let $\mathbf{x} \in \mathbb{R}^c$ be the GAP output of the last convolutional layer of the CNN. This feature map captures channel-wise information about the whole network. Then, a classical linear layer maps the feature vector into a squeezed vector with a reduction ratio of r. This vector is then used as input for the quantum network. Therefore, the reduction layer is formally described as

$$F_{sq}(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^c \to \mathbf{x} \in \mathbb{R}^{\frac{c}{r}} = \mathbf{W}\mathbf{x} + b, \tag{4.19}$$

where $\mathbf{W} \in \mathbb{R}^{c \times \frac{c}{r}}$ is a and $b \in \mathbb{R}^{\frac{c}{r}}$ is the bias vector. Finally, a classical classification layer using a SoftMax activation function retrieved the class probabilities. Figure 4.2 shows an overview of the squeeze and encoding process to map classical to quantum data.

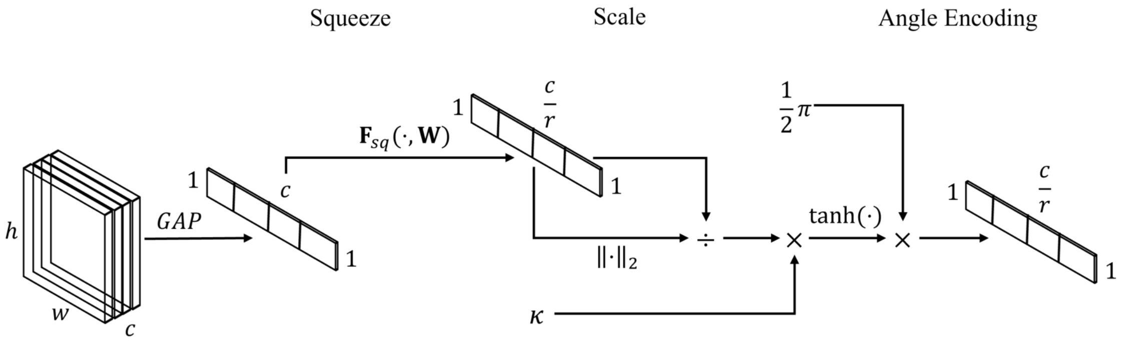


Figure 4.5: *Squeeze, scaling, and angle encoding process in order to map the classical feature vector into a quantum network.*

4.4 | Results and Discussion

4.4.1 | Implementation Details

All the models were trained employing the SGDM optimizer with a learning rate of 10^{-3} and a momentum of 0.9. The models were configured with a batch size of 32 and trained for 100 epochs minimizing the Cross-Entropy Loss. If the validation loss is not improving during 20 epochs, the learning rate is decreased by $\sqrt{0.1}$. The models were implemented using the Pytorch and the Pennylane framework, and the experiments ran on Google's cloud servers, including a Tesla P4 GPU with 2560 CUDA cores and 8 GB of RAM.

Table 4.2: *ResNet ablation study for quantum-transfer learning. The optimal configuration is selected such that the validation loss is minimized. The number of parameters is given in Millions (M). The mean and standard deviation for the validation loss are shown.*

Backbone Network	#-Qubits per circuit	#-Quantum Circuits	Best Validation Loss	Parameters [M]
ResNet18	2	1	0.3247 ($\pm$ 0.0436)	11.18
		2	0.6420 ($\pm$ 0.0334)	
		3	0.6240 ($\pm$ 0.0195)	
		4	0.6349 ($\pm$ 0.0197)	
	4	1	**0.2818 ($\pm$ 0.0952)**	11.18
		2	0.6426 ($\pm$ 0.0193)	
		3	0.6270 ($\pm$ 0.0186)	
		4	0.5887 ($\pm$ 0.0258)	
ResNet34	2	1	**0.2899 ($\pm$ 0.0706)**	21.29
		2	0.6065 ($\pm$ 0.0315)	
		3	0.6085 ($\pm$ 0.0339)	
		4	0.6195 ($\pm$ 0.0104)	
	4	1	0.3480 ($\pm$ 0.0751)	21.29
		2	0.6346 ($\pm$ 0.0263)	
		3	0.6528 ($\pm$ 0.0435)	
		4	0.6371 ($\pm$ 0.0112)	

4.4.2 | Ablation Study

In order to select the best performance Classical-Quantum model, an ablation study is first conducted on an ideally balanced dataset employing the ADSS dataset (see Section 1.2), keeping only a total of 250 real XCA image patches of size 32 × 32 pixels in grayscale, with 125 patches identified with stenosis and 125 with no stenosis. Additionally, the z-score normalization was performed, changing the image range to $[0, 1]$ and applying the ImageNet mean $\mu = [0.485, 0.456, 0.406]^{\intercal}$ and standard deviation $\sigma = [0.229, 0.224, 0.225]^{\intercal}$. The dataset was split in a stratified manner into a training and testing set, each with 125 images. The training subset was additionally partitioned into 5-fold for cross-validation.

The pre-trained model on the ImageNet dataset ResNet18 and ResNet34 were employed as backbone networks. Then, different Quantum Networks configurations were evaluated, including two and four qubits circuits. It is important to point out that the ResNet18 has 11.18 M of parameters and the ResNet34 has 21.29 M. The Quantum Network only contains $\frac{c}{r}$ parameters, therefore # Qubits × # Quantum circuits, which represents a slight increase in the number of parameters. The model with the best validation loss was selected for the upcoming experiments employing the unbalanced ADSS dataset. Particularly, for the ResNet18, the configuration with a single quantum circuit with four qubits achieved the lower loss, and for the ResNet34, the two-qubit single quantum circuit, as shown in Table 4.2.

4.4.3 | Stenosis Classification Performance Comparison

The proposed hybrid classical-quantum network approach aims to enhance the feature representation by fine-tuning a pre-trained network for stenosis detection. The performance was evaluated on the public dataset: ADSS (see Section 1.2) with 125 positive cases of stenosis and 1,394 negative cases of size 32 × 32, employing the best models obtained from the ablation study. Also, the hybrid classical-quantum networks were compared against their classical version, using pre-trained and scratch models.

A 5-fold cross-validation technique was employed to validate the model performance. Numerical results are shown in Table 4.3. The stenosis classification results showed that the pre-trained classical network results improved substantially when the quantum module was employed. For example, the Quantum ResNet18 enhances all the evaluation metrics concerning the Vanilla ResNet18, reaching an accuracy, sensitivity, specificity, precision, and F_1-score of 0.9691, 0.7360, 0.9900, 0.8679, and 0.7950, respectively. This hybrid network setting achieved the best specificity and precision. The Quantum ResNet34 also ameliorated the five evaluation metrics. With the top accuracy, sensitivity, and F_1-score of 0.9730, 0.8080, and 0.8310, respectively. The proposed Quantum ResNet18 obtained the second-best accuracy, sensitivity, and F_1-score. Notice that the Quantum ResNet18 requires 11.18 M of parameters and 21.29 M by the Quantum ResNet34, which is almost a 2x smaller model. The vanilla models trained from scratch surpassed the performance of the quantum models. However, they do not achieve competitive classification rates.

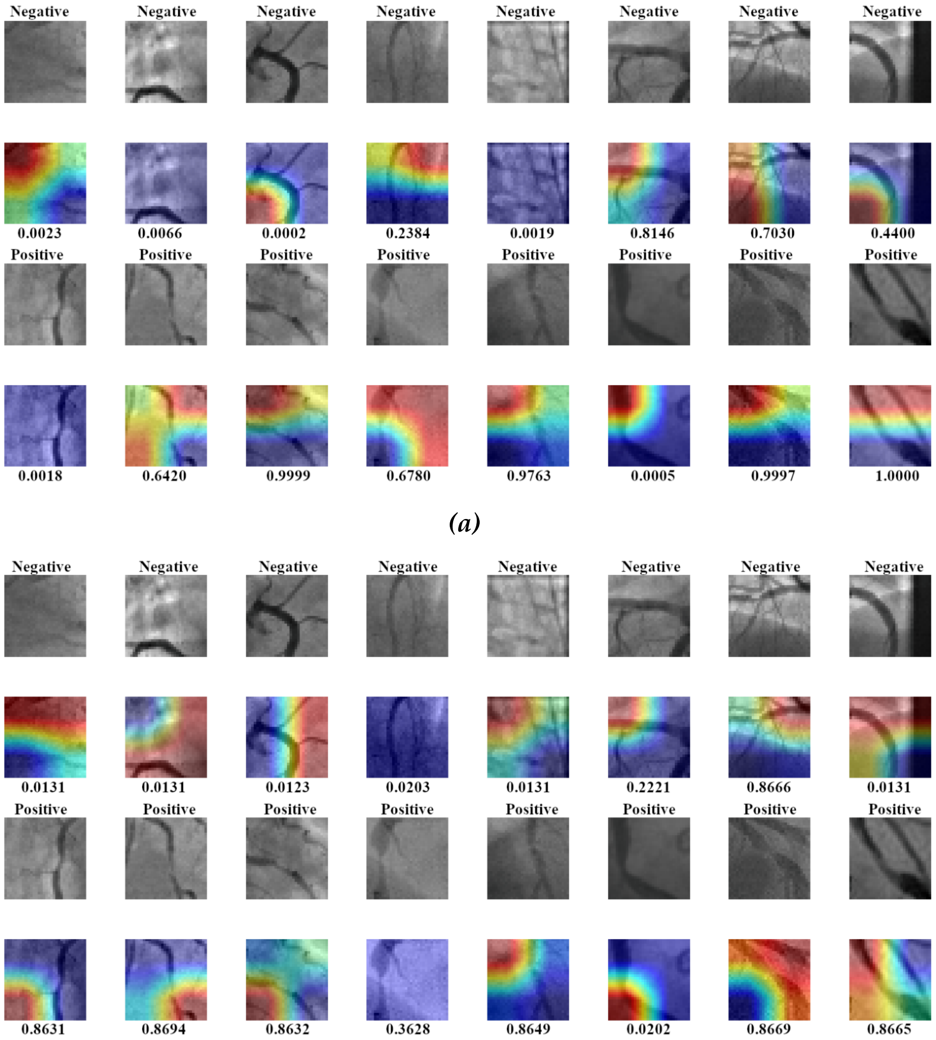
4.4.4 | Class Activation Maps Visualization

Figure 4.6 presents the GradCAM outcome comparing the Vanilla ResNet18 and the Quantum ResNet18 for some challenging test XCA images showing non-stenosis and stenosis cases in the first and third rows. Similarly, Figure 4.7 shows the GradCAM

Table 4.3: *Classical-Quantum Transfer Learning classification performance. The mean and standard deviation of each metric are shown.*

Model	**Pretrained**	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**F$_1$-score**
Vanilla ResNet18	✗	0.9507	0.6560	0.9771	0.7205	0.6844
		± 0.0055	± 0.0742	± 0.0049	± 0.0350	± 0.0453
	✓	0.9651	0.6960	0.9892	0.8583	0.7646
		± 0.0064	± 0.0784	± 0.0051	± 0.0571	± 0.0507
Quantum ResNet18	✗	0.9211	0.3600	0.9713	0.5255	0.4221
		± 0.0170	± 0.1431	± 0.0093	± 0.1334	± 0.1329
	✓	0.9691	0.7360	**0.9900**	**0.8679**	0.7950
		± 0.0064	± 0.0697	± 0.0027	± 0.0322	± 0.0483
Vanilla ResNet34	✗	0.9507	0.6240	0.9799	0.7444	0.6745
		± 0.0036	± 0.0599	± 0.0074	± 0.0573	± 0.0224
	✓	0.9671	0.7600	0.9857	0.8420	0.7901
		± 0.0029	± 0.0839	± 0.0082	± 0.0863	± 0.0250
Quantum ResNet34	✗	0.9151	0.5520	0.9477	0.6990	0.5358
		± 0.0654	± 0.2352	± 0.0850	± 0.2190	± 0.1738
	✓	**0.9730**	**0.8080**	0.9878	0.8572	**0.8310**
		± 0.0032	± 0.0392	± 0.0029	± 0.0288	± 0.0220

for the Vanilla ResNet34 and the Quantum ResNet34. The test images include high-contrasted background, blood vessel bifurcations, and multiple blood vessels, each of different widths. It is observed that high-intensity regions (red color) focused on a background region in the Vanilla models, while the Quantum models set great attention over blood vessel regions. Moreover, the Quantum models correctly classify challenging test images (see first positive image). However, in both cases (classical and quantum), the configurations were unable to classify as non-stenosis an XCA image with multiple blood vessel bifurcations (see seventh negative image) and as a stenosis case for an XCA image with a severe blood flow reduction (see fourth positive image). Also, the Quantum ResNet18 could locate high-attention regions over blood vessel pixels in negative stenosis cases but not the Quantum ResNet34. These visual examinations provide valuable information about the localization improvement of features with the Quantum models concerning their classical configuration.


Negative Negative Negative Negative Negative Negative Negative Negative
0.0023 0.0066 0.0002 0.2384 0.0019 0.8146 0.7030 0.4400
Positive Positive Positive Positive Positive Positive Positive Positive
0.0018 0.6420 0.9999 0.6780 0.9763 0.0005 0.9997 1.0000
Negative Negative Negative Negative Negative Negative Negative Negative
0.0131 0.0131 0.0123 0.0203 0.0131 0.2221 0.8666 0.0131
Positive Positive Positive Positive Positive Positive Positive Positive
0.8631 0.8694 0.8632 0.3628 0.8649 0.0202 0.8669 0.8665

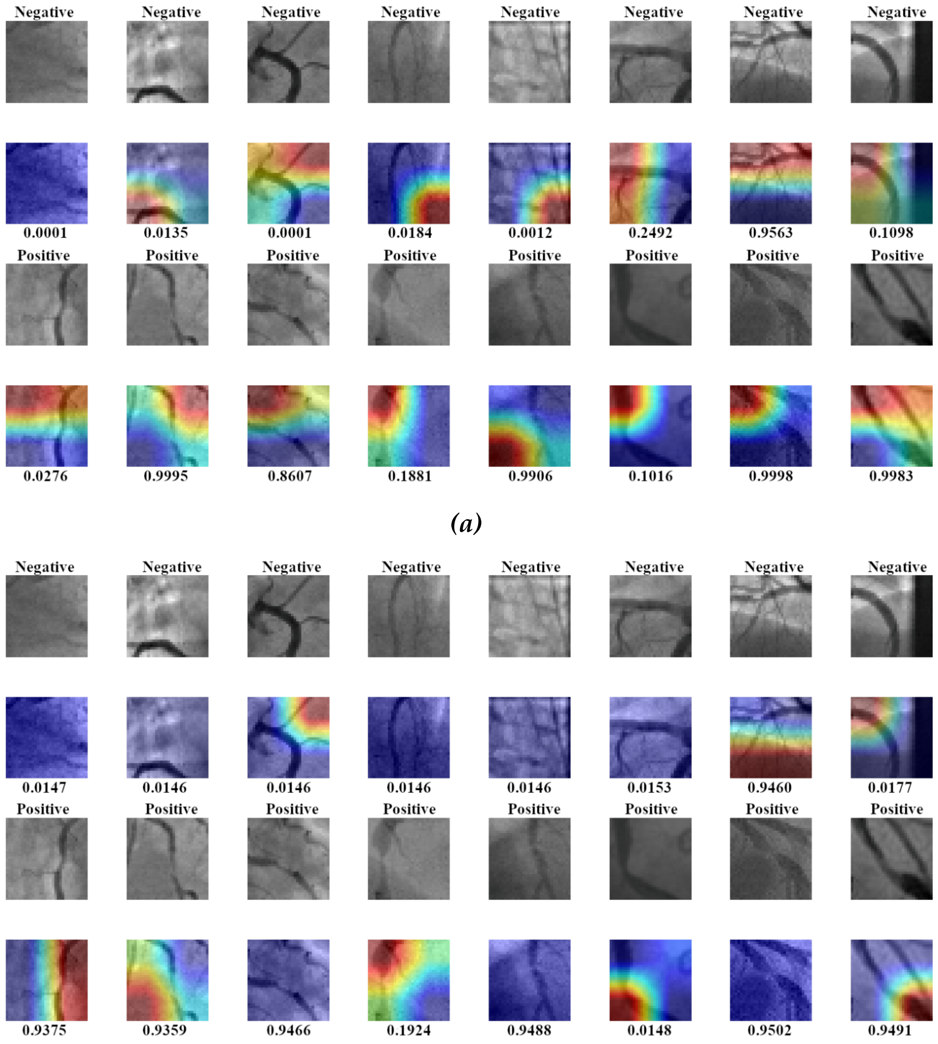
(a)

(b)

Figure 4.6: *Hybrid Classical-Quantum Transfer Learning GradCAM (a) Vanilla ResNet18 (b) Quantum ResNet18*

4.5 | Conclusion

This chapter presented a Hybrid Classical-Quantum Network for stenosis detection in XCA images demonstrating the quantum computing potential. The framework connects a QN plugged into a classical CNN that produces the primary features representation. This research contributes to the QN architecture, where multiple (and smaller) VQCs can replace a single VQC. Additionally, to facilitate overall training convergence, a novel squeeze, scaling, and angle encoding process map the classical



(b)

Figure 4.7: *Hybrid Classical-Quantum Transfer Learning GradCAM (a) Vanilla ResNet34 (b) Quantum ResNet34*

feature vector into a quantum network. This proposed approach bound the input features of the QN, avoiding and controlling saturation by a smoothing parameter κ that was learned during the optimization process. Numerical results demonstrate that the proposed hybrid model significantly improved the detection performance when the CNN sub-module was pre-trained with the ImageNet dataset. Concerning the classical transfer learning approach, the quantum models obtained the best evaluation metrics, with a significant boost in the sensitivity and F_1-score, both up to 4%. Also, the GradCAM technique was applied to obtain a heat map of the regions that have more influence in the classification process. This visualization reveals that if the high attention region is mostly on background pixels, the network misclassifies the input XCA image. Furthermore, the proposed approach can be easily customized and integrated into any CNN architecture.

5

Generative Model for Coronary Angiography Stenosis cases

"Victories over ingrained patterns of thought are not won in a day or a year."

— Isaac Asimov, *The Naked Sun*

5.1 | Mathematical Foundations

Given a finite set of samples $\mathbf{x} \in \mathbf{X} \sim P(\mathbf{X})$ the goal of a generative function $G(\boldsymbol{\theta})$ is to learn a set of parameters $\boldsymbol{\theta}$ such that unlimited synthetic samples $\tilde{\mathbf{x}} \sim P(\tilde{\mathbf{X}})$ can be generated. Another way to state this goal is that an optimal $\boldsymbol{\theta}$ needs to be found such that $P(\tilde{\mathbf{X}}) \approx P(\mathbf{X})$. In these settings, the generator needs to be trained to minimize a distance D between real and synthetic data as follows:

$$D = \min_{\boldsymbol{\theta}} d(P(\mathbf{X}), P(\tilde{\mathbf{X}})), \tag{5.1}$$

where $d(\cdot)$ is a similarity metric.

In order to represent the marginal distribution $P(\tilde{\mathbf{X}})$ it is important to have generative models of great capacity so no single expression defines it. However, it can be encoded in two main directions. One option is an explicit density estimation, where explicitly define and solve for $P(\tilde{\mathbf{X}})$ in a sequential mode, which results in computational slowness, *i.e.*, fully visible belief network and variational autoencoders. The second option is an implicit density estimation, such that a model learn $P(\tilde{X})$ without explicitly defining it, *i.e.*, Generative Stochastic Networks and Generative Adversarial Networks. However, the generator can generate similar-looking samples from the same data mode.

Instead of working in the distribution space, another option exists where the feature space is approximated. Feature space-based generative model defines a function family $g(\theta) \in G(\boldsymbol{\theta})$ that explicitly and efficiently generates and refines synthetic samples using prior modeling information about the samples $\mathbf{X}$, *i.e.*, curve modeling. In this way, the generative model minimizes a distance D between real and synthetic feature space such as:

$$D = \min_{\boldsymbol{\theta}} d(\mathcal{X}, \tilde{\mathcal{X}}), \tag{5.2}$$

where $d(\cdot)$ is a distance function.

In the deep learning context, the feature space is computed employing a CNN, as illustrated in Figure 5.1.

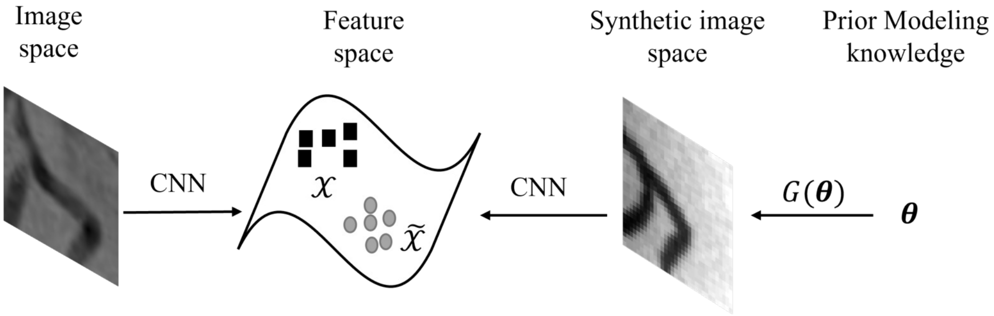


Figure 5.1: *Feature based generative model. The generative model $G(\boldsymbol{\theta})$ minimizes a distance D between real $\mathcal{X}$ and synthetic feature space $\tilde{\mathcal{X}}$. The same CNN computes the feature space.*

5.2 | Related Work

Only a handful of generative models have been explored in the medical image domain in the literature. For instance, Cohen et al. [2018] discussed the injected bias in Generative Adversarial Networks (GANs) and how distribution matching losses can lead to the misdiagnosis of medical conditions due to the lack of criteria for ensuring or preserving intra-operative content. Thus, the class labels might not be preserved. In this wise, Gil et al. [2019] proposed a multi-objective optimization strategy for a CycleGAN, ensuring a mapping between synthetic images and the real domain, preserving anatomical content. This approach has been applied to simulate intra-operative bronchoscopic videos and chest CT scans from simple graphical primitives that generate sketches. Following this research direction, Tetteh et al. [2020] presented a neural network to generate 3-D angiographic volumes, which implemented a simulator of a vascular tree that followed a generative process inspired by the biology of angiogenesis (a model that mimics arterial growth). This approach simulates a physiologically plausible blood vessel segment as a cylinder in 3D space that includes different segment types, such as root, bifurcation, and leaves. Additionally, different background and foreground intensity patterns with different signal-to-noise ratios where evaluated in the generative process.

On the other hand, classical machine learning techniques have also been explored. Keelan et al. [2016] developed a method to generate 3D coronary arterial trees based on the tissue structure and the entry point positions of the largest arteries. The parameters associated with an arbitrary tree configuration define a total cost function which gives a numeric measure of the fitness of a given tree. The optimized blood vessels were obtained using a simulated annealing-based approach and validated through comparison with morphological data from the porcine arterial tree. Also, Jaquet et al. [2018] proposed a patient-specific hybrid image-based and synthetic geometric model for generating cardiac vascular trees, emerging from actual vascular tree models segmented from CTTA images. The model consisted of a multiple tree angiogenesis simulation governed by minimizing the total tree volume with flow-related and geometrical constraints. Antczak and Liberadzki [2018] introduced a simplified X-ray coronary blood vessel model. A 2D image was generated based on the assumption that a Bezier curve involving additive random noise can parameterize a small blood vessel region. Under this approach, the generated patches can include stenosis areas according to the curve width. However, the curves are drawn with independent parameters, generating images with non-vascular structures and non-visible stenosis areas due to curves overlapping. Thus, bifurcation structures are not guaranteed. Nevertheless, studying bifurcations in the coronary vascular tree is crucial to accurately classify and localize coronary bifurcation lesions [Antoniadis et al., 2015; Chang et al., 2019; Chiastra et al., 2016; Iakovou et al., 2011]. For such a reason, it is paramount to model bifurcations and stenotic regions.

5.3 | Hierarchical Bezier-based Generative Model

A Hierarchical Bezier-based Generative Model (HBGM) is presented in this chapter. Small regions of XCA artery blood vessels are modeled as grayscale images representing a set of curves of several lengths, drawn on a gradient background, and noise-distorted. Notice that no previous information on real XCA is used. Moreover, two constraints are employed to accept a generated patch. First, the ratio of blood vessel pixels concerning the image size must be above a threshold. Secondly, for patches where stenosis is created, the ratio of stenosis blood vessel pixels must be greater than a fixed value. As such, the proposed generative model creates images including blood vessel structures with stenosis regions, containing 10k images, 50% with stenosis, and 50% with no-stenosis cases. The generative process is divided into three steps: drawing area and gradient background generation, Bezier curve parametrization, and Bezier curve drawing.

5.3.1 | Drawing Area and Background Generation

Let $\mathbf{I} \in \mathbb{R}^{w \times h}$ be the gray-scale patch generated with a given width w and height h, respectively. Firstly, a white drawing area or canvas DA $\in \mathbb{R}^{3w \times 3h}$ is created as

$$\mathrm{DA} = 255\,\mathbf{J}, \tag{5.3}$$

where $\mathbf{J} \in \mathbb{R}^{3w \times 3h}$ is an all-ones matrix. Then, the radial gradient background is generated over the DA, were the intensity of each pixel $u = (i, j) \forall i \in [0, w], j \in [0, h]$ is given by

$$\mathrm{DA}(u) = \alpha\ (1 - d(u, c_g)) + \beta\ d(u, c_g), \tag{5.4}$$

with c_g as the center of the radial gradient, $\alpha = rand(0, a)$ and $\beta = rand(a, b)$ with $0 < a < b \leq 255$ are the lower and upper pixel intensities, and $d(u, c_g)$ is a distance between the current position and the gradient center.

In this manner, let U be a uniformly distributed random variable in the interval $[0, 1]$, then a uniformly distributed function in the interval $[a, b]$, here denoted as $rand(a, b)$, can be given by

$$rand(a, b) = a + (b - a)\, U. \tag{5.5}$$

Thus, the radial gradient background is generated, centered as:

$$c_g = [rand(-w, 2w), rand(-h, 2h)]. \tag{5.6}$$

The distance d is defined by:

$$d(u, c_g) = \frac{1}{2} \frac{|u - c_g|}{\sqrt{w^2 + h^2}}, \qquad d \in (0, 1). \tag{5.7}$$

5.3.2 | Bezier Curve Parametrization

Under the prior assumption that a Bezier curve can parameterize a small region of a single blood vessel [Antczak and Liberadzki, 2018], an arterial vessel structure can be defined as a central (parent) Bezier curve $B^{(p)}$. Then this parent curve can hold a subset

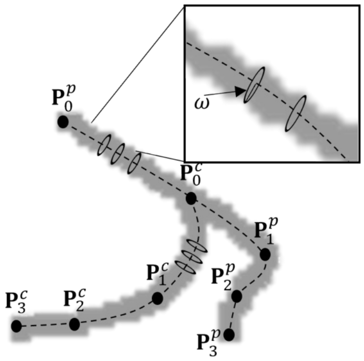


Figure 5.2: *Cubic Bezier curve example. Four control points define this Bezier curve. Points* $\mathbf{P}_0^p$ *and* $\mathbf{P}_3^p$ *are the ends of the curve, and points* $\mathbf{P}_1^p$*, and* $\mathbf{P}_2^p$ *determine the shape of the parent curve. Point* $\mathbf{P}_i^c$ *are the points of the child curve.*

of c Bezier curve children $\mathbf{B}^{(c)} = \{B_1^{(c)}, B_2^{(c)}, \cdots, B_c^{(c)}\}$. Moreover, it holds that any child's width is smaller than their respective parent's. In such a way, a more complex vascular model can be accomplished.

Any Bezier curves can be expressed as

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i \mathbf{P}_i, \quad 0 \leq t \leq 1 \tag{5.8}$$

where $\binom{n}{i}$ is the binomial coefficients, n is the curve degree, t is the number of points that a Bezier curve was discretized (*e.g.*, if $ts = 100$, the curve is constructed by the values $t = [0, 0.01, 0.02, \cdots, 0.98, 0.99, 1]^\top$, and $\mathbf{P}_i$ are the control points, were the first and the last control points, $\mathbf{P}_0, \mathbf{P}_n$, are always the curve's endpoints. Thus, the parent control points are randomly chosen to lie inside the canvas region, given by:

$$\mathbf{P}^{(p)} = \begin{bmatrix} rand(-w, 2w) & 0 \\ rand(\ 0, w) & rand(\ 0, h) \\ rand(\ 0, w) & rand(\ 0, h) \\ rand(-w, 2w) & h \end{bmatrix}, \tag{5.9}$$

where the control points $\mathbf{P}_1^{(p)}$ and $\mathbf{P}_2^{(p)}$ determining the shape of the curve. Then, it follows that the control point $\mathbf{P}_0^{(c)}$ of any child curve is subject to lie on the parent curve $B^{(p)}$, such as:

$$\mathbf{P}_0^{(c)} = [rand(B_0^{(p)}, B_t^{(p)})], \tag{5.10}$$

Figure 5.2 shows a cubic Bezier curve with one parent and one child Bezier curve, respectively.

5.3.3 | Bezier Curve Drawing

The widths for each Bezier curve are given by

$$W(k) = \begin{cases} \omega \ \max\left(0.3, 1.0 – 10\mathcal{N}\left(k, \mu, \sigma^2\right)\right) & \text{if stenosis (+)} \\ \omega & \text{otherwise,} \end{cases} \tag{5.11}$$

where $\mathcal{N}(t, \mu, \sigma^2)$ is a normal random variable with mean μ and variance σ^2 for a real number $k \in [0, ts]$, and ω is a basic width (given in pixels). Notice that if a curve, ergo, a blood vessel, is constructed with a stenosis region, its width is affected by a stenosis factor. Therefore, μ controls the position of the stenosis region center and σ^2 the stenosis' length, as shown in Figure 5.3.

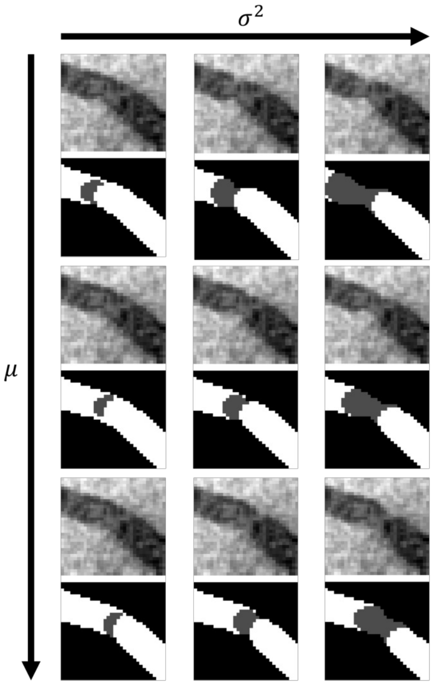


Figure 5.3: *Parameters μ and σ^2 affect the stenosis position and length. From top to bottom: $\mu = \{40, 50, 60\}$ and from left to right: $\sigma^2 = \{2, 4, 8\}$. The generated image is shown with the corresponding segmentation and stenosis location ground-truth.*

During image acquisition and transmission through analog circuitry in conventional X-ray techniques, the Additive White Gaussian Noise (AWGN) is the most prevalent. Moreover, X-ray imaging systems manifest blur that reduces the sharpness of image edges and the overall contrast. The image can also exhibit the effects of Poisson, Impulsive, and Quantization noises. However, these are rare occurrences related to faulty device manufacturing [Lee and Kang, 2021; Manson et al., 2019; Mohan et al., 2020]. Therefore, AWGN and Gaussian blur are applied to the generated image to simulate the image acquisition process. Algorithm 1 summarizes the proposed generative framework.

Algorithm 1: Hierarchical Bezier-based Generative Patch Model

```
Data: Patch size $(w, h)$, number of patches $N$, background limit intensities $(a, b)$,
      parent vessel basic width $\omega$, number of child's vessels $C$, stenosis case
      (True or False), stenosis position $\mu$, stenosis length $\sigma$.
Result: Artificial XCA dataset
for $n \leftarrow 0$ to $N$ do
    Create a canvas DA as (5.3);
    Draw the gradient background such as (5.4);
    Generate a parent Bezier curve $B^{(p)}$ following (5.8) with control points given
      by (5.9);
    Draw the curve $B^{(p)}$ with a width given by (5.11);
    for $c \leftarrow 0$ to $C$ do
        Generate each child Bezier curve $B_c^{(C)}$ following subject to the control
          point $\mathbf{P}_0^{(c)}$ lie in $B^{(p)}$;
        Draw the curve $\mathbf{B}_c^{(C)}$ with a width given by (5.11);
    end
    Add white noise in the image;
    Add Gaussian Blur in the image;
end
```

5.3.4 | Generative Model Performance Measure

The objective of a generative model is to produce data that matches the observed (real) data. Some widely used metrics in GANs [Borji, 2019] can be exploited to measure the distance between the feature maps of observing real-world data $\mathcal{X}$ and the generative model $\tilde{\mathcal{X}}$. The more relevant is the Average Log-likelihood [Goodfellow et al., 2014], the Wasserstein Critic (WC) [Arjovsky et al., 2017], the Inception Score (IS) [Salimans et al., 2016], and the Fréchet Inception Distance (FID) [Heusel et al., 2017]. Each one has its drawbacks. For instance, the Average Log-likelihood metric requires a vast number of samples to approximate the true log-likelihood. It also fails when the data dimensionality is high. The WC distance is not a smooth function, requiring high processing time to be computed. The IS is an asymmetric measure that only considers $P(\mathcal{X})$ and ignores $P(\tilde{\mathcal{X}})$. The FID assumes that features are of Gaussian distribution, which is often not guaranteed; however, it performs well in terms of discriminability, robustness, and computational efficiency [Dimitrakopoulos et al., 2020; Liu et al., 2020; Zhang et al., 2019]. Therefore, the FID is selected as a metric for the generative model, given that CNN conducts the classification process.

The FID requires a feature function ϕ (by default, the activation is from the penultimate pooling layer of a pre-trained Inception-v3 model) to evaluate the similarity of real data and generated data. However, ϕ can use any pre-trained model. In this work, ϕ is the feature vector obtained by applying a global average pooling over the third residual block of the ResNet18 (pre-trained on the ImageNet dataset). Therefore, an image is embedded into a vector with 256 features. These output vectors are summarized as a continuous multivariate Gaussian; the mean and covariance are estimated for the real XCA and the generated datasets. Thus, the FID is given by

$$\mathrm{FID}(\mathcal{X}, \tilde{\mathcal{X}}) = ||\mu_{\phi_r} - \mu_{\phi_g}||_2^2 + Tr\left(\Sigma_{\phi_r} + \Sigma_{\phi_g} - 2\left(\Sigma_{\phi_r}\Sigma_{\phi_g}\right)^{1/2}\right), \tag{5.12}$$

where ϕ_r and ϕ_g are the embedding feature vectors of the real and artificial images, with their respective means μ_{ϕ_r}, μ_{ϕ_g} and empirical covariance matrices Σ_{ϕ_r}, Σ_{ϕ_g}. Tr is the trace of the matrix. Accordingly, a lower FID indicates a better-looking image patch; conversely, a higher score indicates a poor-looking artificial patch.

5.4 | Results and Discussion

5.4.1 | Implementation Details

The HBGM is governed by a set of free parameters, summarized in Table 5.1. The grid search optimization algorithm [Bergstra and Bengio, 2012] was employed to search through the manually specified subset of the hyperparameter space of the HBGM algorithm. A total of 10K images were selected, where the $\mathrm{FID}(\mathcal{X}, \tilde{\mathcal{X}}) < \epsilon$, with $\epsilon = 400$.

Subsequently, three different ResNet models (18, 34, 50) were pre-trained using the synthetic dataset employing the SGD optimizer with an initial learning rate of 10^{-2} and a momentum of 0.9 during 500 epochs. Additionally, if the loss is not improving during 20 consecutive epochs, a learning rate decay policy was set by a factor of $\sqrt{0.1}$. Next, a fine-tuning step was carried out for 100 epochs and with an initial learning rate of 10^{-2} for the full ADSS dataset. All the models were implemented using the PyTorch framework, and the experiments ran on Google's cloud servers, including a Tesla P4 GPU with 2560 CUDA cores and 8 GB of RAM.

Table 5.1: *Generative model parameters. A total of 10k images was generated, 50% with stenosis cases and the remainder with non-stenosis.*

Parameter	Description	Value(s)
(w, h)	Patch size	(32×32)
a	Gradient background lower intensity limit	$rand(25, 50)$
b	Gradient background upper-intensity limit	$rand(85, 105)$
t	Number of points for each curve	100
ω	Basic width in pixels	$rand(1, 4)$
P	Number of parent's curves	$rand(1, 3)$
C	Number of child's curves	$rand(0, 3)$
μ	Stenosis position	$rand(\frac{1}{4}t, \frac{3}{4}t)$
σ	Stenosis length	$rand(4, 8)$

5.4.2 | Generative Model Performance

As expected, the FID retrieves a qualitative similarity measure between real and artificial images. Accordingly, in evaluating each image of the BGM [Antczak and Liberadzki, 2018], a mean FID of 92.7967 and a minimum and maximum score of 57.4544 and 376.6007 are obtained. Figure 5.4a shows the distribution of the similarity scores given by the FID, showing a lower and upper quartile of 73.8502 and 94.0127, respectively. Also, 1,159 of 10,000 samples have high FID values, indicating images with a higher dissimilarity, which are shown as upper outliers in the box-plot.

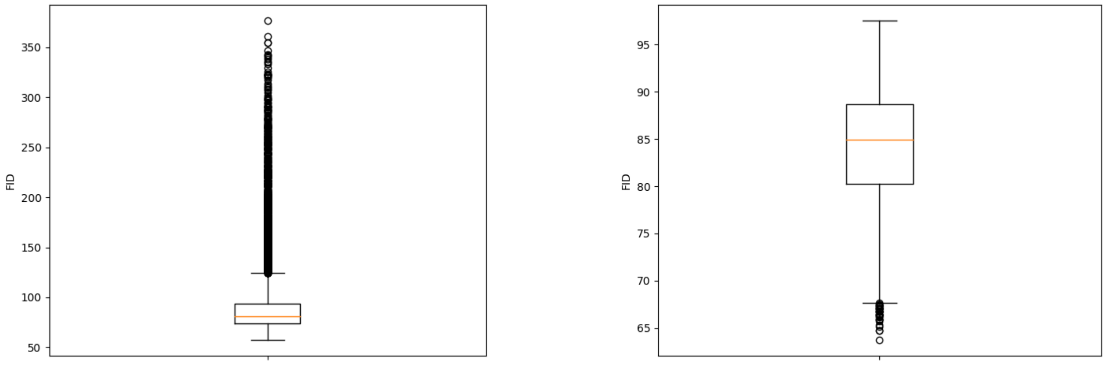
Further, the proposed HBGM reached a mean FID of 84.0886, and a minimum and maximum score of 63.7430 and 97.4983, respectively. Figure 5.4b shows the corresponding distribution of the FID, with a lower and upper quartile of 80.2559 and 88.6376, respectively. Therefore, the proposed HBGM obtains a lower average FID, indicating that more realistic visual images were generated.

Figure 5.5 shows a sample of the real and artificial images. Hence, the baseline generative model creates blood vessel intersections, not bifurcations, as the proposed generative model. Moreover, each parent blood vessel can contain c-bifurcations, being able to model different stenosis percentages and blood vessel widths (see the third row in Figure 5.5).

5.4.3 | Stenosis Classification Performance Comparison

Table 5.2 summarizes the stenosis detection results, highlighting more outstanding metrics. The evaluated ResNet networks were trained by using four different strategies:

1. Trained from scratch: only the real XCA dataset was used to optimize a random



***(a)** Baseline Generative Model.* ***(b)** Hierarchical Bezier Generative Model.*

***Figure 5.4:** Fréchet Inception Distance (FID) distributions. (a) Baseline and (b) Proposed Generative Models.*

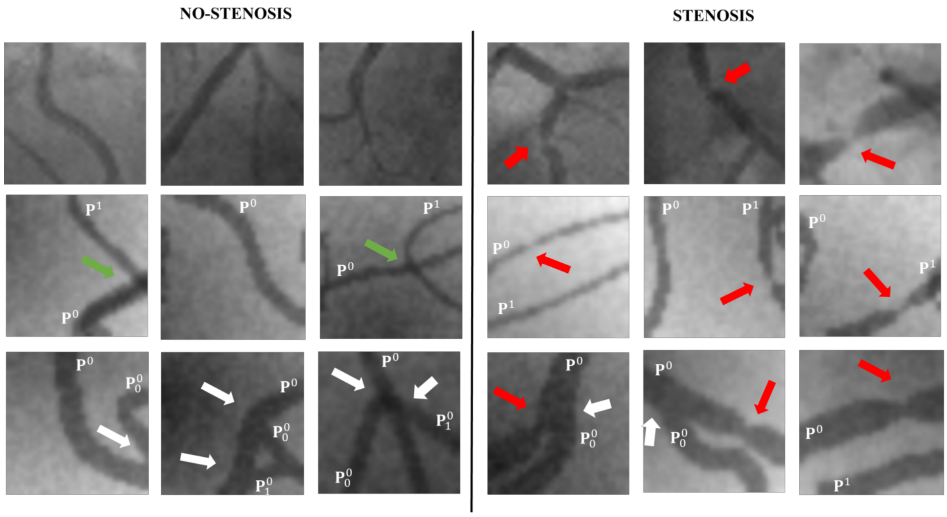


***Figure 5.5:** Generated XCA image patches with non-stenosis and stenosis. First row: real patches, the second row: Baseline generative model patches, and third row: proposed generative model patches. The proposed generative model creates patches with blood vessel bifurcations, with a clearer stenosis area and blood vessels with different weights. The red arrows indicate the stenosis case, the white ones the bifurcation points, and the green arrows the intersections of two-parent curves. Besides, $\mathbf{P}^i$ represents the i-th parent curve and $\mathbf{P}^i_j$ the j-th child curve of the i-th parent, respectively.*

weight initialized network.

2. Pre-trained on ImageNet: a pre-trained network on the ImageNet dataset was fine-tuned using only the real XCA dataset.

3. Pre-trained on BGM: the network was previously trained by employing the

baseline of the artificial dataset and then fine-tuned with the real XCA images.

4. Pre-trained on HBGM: the network was previously trained by employing the proposed artificial dataset and then fine-tuned with the real XCA images.

The fine-tuned models that employed the artificial baseline dataset show only an improvement in sensitivity compared to the trained-from-scratch configuration. For instance, in the ResNet18, the sensitivity obtained a gain from 0.6560 to 0.7280; in the ResNet34, from 0.6240 to 0.6800, and for the ResNet50, from 0.2480 to 0.7120. Precision and F_1-score were also improved; reaching 0.5002 and 0.5851, respectively.

On the other hand, when the proposed synthetic dataset was employed to pre-train the models, the best sensitivity and F_1-score were attained for the ResNet18, with 0.9200 and 0.7931, respectively. Thus, boost of 23% and 3% for each metric concerning the ImageNet pre-trained configuration. The ResNet18 pre-trained with the ImageNet achieved the best specificity (0.9892) and precision (0.8583), 2% and 16% higher than the pre-trained with the HBGM dataset.

The best accuracy was obtained for the ResNet50 pre-trained with ImageNet, with a 0.9671; this is a 3% gain compared to the ResNet50 pre-trained with the HBGM.

The comparative analysis proved the proposed generative model efficacy as a pre-trained dataset and as an alternative to ImageNet pre-trained models for stenosis detection. It is noteworthy that the ImageNet dataset has 1,281,167 images that optimize the network for 600,000 epochs [He et al., 2016]; meanwhile, the HBGM performs a pre-training step only for 200 epochs and fine-tuning for 100 epochs.

5.4.4 | Class Activation Maps Visualization

As mentioned before, the GradCAM is a procedure that generates a coarse localization map, highlighting the most important regions in the image for predicting a class of interest. Figure 5.6 shows the GradCAM visualization concerning the different ResNet18 pre-training configurations. A reasonable prediction explanation produces discriminative visualizations over blood vessel regions. Thus, discriminative regions were highlighted in red and those with lower contributions in purple. It can be seen that Resnet18 pre-trained with ImageNet showed high attention regions in background areas for negative stenosis cases.

In constrast, pre-trained with the generative model put high attention in blood vessel regions. Moreover, for positive stenosis cases, the ResNet18 pre-trained with the HBGM, generated higher and more refined attention maps over blood vessel regions than the pre-trained with BGM and ImageNet. Also, the stenosis probability

Table 5.2: *Transfer Learning classification performance from generative model. ∗ represents the proposed generative dataset. The mean and standard deviation of each metric are shown.*

Model	Pretrained	Accuracy	Sensitivity	Specificity	Precision	F$_1$-score
Vanilla ResNet18	✗	0.9507 ± 0.0055	0.6560 ± 0.0742	0.9771 ± 0.0049	0.7205 ± 0.0350	0.6844 ± 0.0453
	ImageNet	0.9651 ± 0.0064	0.6960 ± 0.0784	**0.9892** ± 0.0051	**0.8583** ± 0.0571	0.7646 ± 0.0507
	BGM	0.9289 ± 0.0199	0.7280 ± 0.1197	0.9470 ± 0.0251	0.5690 ± 0.0866	0.6293 ± 0.0748
	HBGM*	0.9605 ± 0.0045	**0.9200** ± 0.0320	0.9642 ± 0.0045	0.6970 ± 0.0191	**0.7931** ± 0.0195
Vanilla ResNet34	✗	0.9507 ± 0.0036	0.6240 ± 0.0599	0.9799 ± 0.0074	0.7444 ± 0.0573	0.6745 ± 0.0224
	ImageNet	**0.9671** ± 0.0029	0.7600 ± 0.0839	0.9857 ± 0.0082	0.8420 ± 0.0863	0.7901 ± 0.0250
	BGM	0.9355 ± 0.0103	0.6800 ± 0.0980	0.9584 ± 0.0077	0.5949 ± 0.0514	0.6323 ± 0.0646
	HBGM*	0.9539 ± 0.0105	0.8000 ± 0.0543	0.9677 ± 0.0085	0.6897 ±0.0567	0.7407 ± 0.0523
Vanilla ResNet50	✗	0.9204 ±0.0092	0.2480 ±0.1568	0.9806 ±0.0058	0.4968 ±0.1073	0.3162 ±0.1597
	✓	0.9520 ± 0.0049	0.5520 ± 0.0531	0.9878 ± 0.0018	0.8016 ± 0.0285	0.6528 ± 0.0440
	BGM	0.9164 ±0.0122	0.7120 ±0.0466	0.9348 ±0.0153	0.5002 ±0.0465	0.5851 ±0.0328
	HBGM*	0.9474 ± 0.0104	0.8400 ± 0.0891	0.9570 ± 0.0146	0.6364 ± 0.0384	0.7241 ± 0.0452

for true positive cases is higher than the other pre-training strategies. Thus, a visual and quantitative validation of the benefits of pre-training with the proposed generative model is presented.

5.5 | Conclusions

Herein, a Hierarchical Bezier Generative Model has been proposed to address the problem of a small and poorly diversified database for stenosis detection in XCA images. A large-scale labeled dataset consisting of 10k images was created using the proposed approach. Extensive experiments showed that pre-train ResNets using this dataset and a posterior fine-tuning with real XCA images achieved the best overall performance on two (of five) evaluation metrics and competitive results on the remainder. Moreover, it demonstrated the value of transferring the weights pre-trained using a more alike (artificial) dataset instead of the ImageNet dataset for stenosis detection tasks with only limited data available.

One drawback of the proposed generative model is that the FID between synthetic and real data is computed after creating the dataset. This implies that the parameters that govern the model are not optimized and were handcrafted. The generative model is governed by a set of free parameters that control the background of the image, the number, and the width of the Bezier curve that simulate real coronary artery vessels. Furthermore, parameters also include settings, allowing control of the stenosis grade of a blood vessel. At this point, the FID between the generated and the real images cannot be controlled as a discriminator loss function in the GANs. For this reason, one future extension of the proposed HGBM is to learn the parameters that govern the model in a deep learning way, such as classical GAN. The HGBM only requires a handful of free parameters compared to the millions of parameters that requires a traditional GAN.

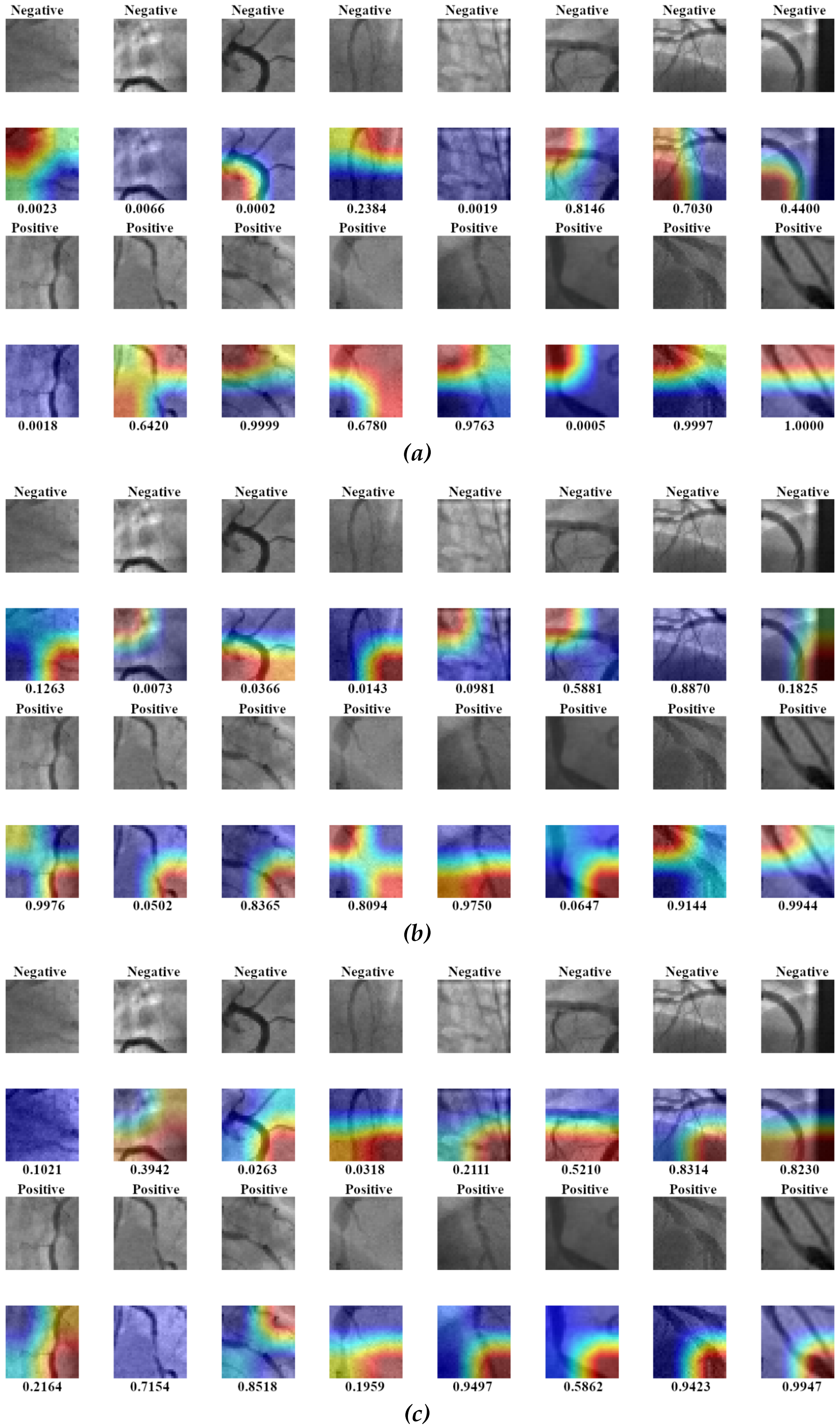

Negative Negative Negative Negative Negative Negative Negative Negative
0.0023 0.0066 0.0002 0.2384 0.0019 0.8146 0.7030 0.4400
Positive Positive Positive Positive Positive Positive Positive Positive
0.0018 0.6420 0.9999 0.6780 0.9763 0.0005 0.9997 1.0000
(a)
Negative Negative Negative Negative Negative Negative Negative Negative
0.1263 0.0073 0.0366 0.0143 0.0981 0.5881 0.8870 0.1825
Positive Positive Positive Positive Positive Positive Positive Positive
0.9976 0.0502 0.8365 0.8094 0.9750 0.0647 0.9144 0.9944
(b)
Negative Negative Negative Negative Negative Negative Negative Negative
0.1021 0.3942 0.0263 0.0318 0.2111 0.5210 0.8314 0.8230
Positive Positive Positive Positive Positive Positive Positive Positive
0.2164 0.7154 0.8518 0.1959 0.9497 0.5862 0.9423 0.9947
(c)

Figure 5.6: *HBGM Transfer Learning GradCAM for Vanilla ResNet18: (a) pre-trained with ImageNet, (b) pre-trained with BGM, (c) pre-trained with HBGM.*

6

Attention-based Convolutional Neural Network for Stenosis Detection

"Did you ever feel ... some sort of extra power that you aren't using - you know, like all the water that goes down the falls instead of through the turbines?"

— Aldous Huxley, *Brave New World*

6.1 | Mathematical Foundations

State-of-the-art networks for natural image classification have recently utilized attention mechanisms to enhance network performance [Hu et al., 2018; Wang et al., 2020; Woo et al., 2018]. These attention mechanisms aim to improve feature map learning by refining channel attention or spatial attention relationships between features. This section will examine three attention modules evaluated in XCA images for stenosis detection.

6.1.1 | Squeeze-and-Excitation Attention Mechanism

A Squeeze-and-Excitation (SE) block [Hu et al., 2018] integrates two operations, a squeeze operation and an excitation operation, to model channel-wise feature relationships as a gating mechanism. This allows the network to enhance hierarchical features in a channel-wise manner. The structure of an SE block is illustrated in Figure 6.1.

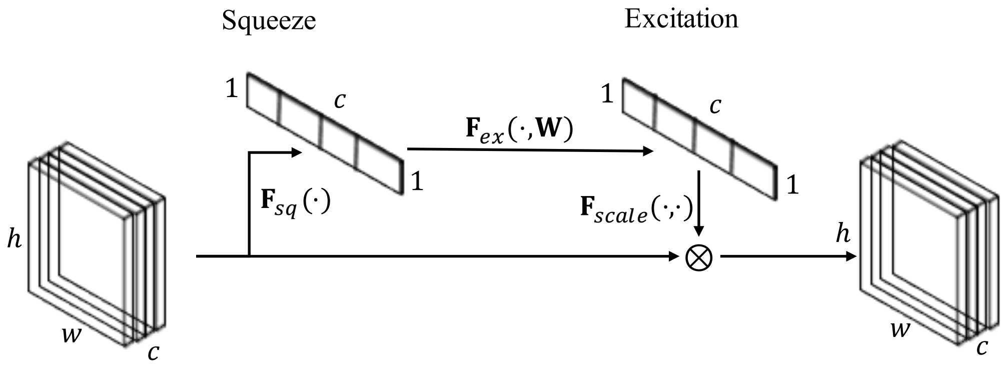

Squeeze
Excitation
1
c
1
$\mathbf{F}_{ex}(\cdot,\mathbf{W})$
1
c
1
$\mathbf{F}_{sq}(\cdot)$
$\mathbf{F}_{scale}(\cdot,\cdot)$
h
w
c
h
w
c

Figure 6.1: *Squeeze-and-Excitation block. The input features are recalibrated ($\mathbf{F}_{scale}(\cdot,\cdot)$) by learnable weights ($\mathbf{F}_{ex}(\cdot,\mathbf{W})$) that capture the channel dependencies ($\mathbf{F}_{sq}(\cdot)$).*

Squeeze operation

To capture channel dependencies between the input feature maps $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$, where $h \times w$ is the spatial size of the features and c is the number of channels, a GAP [Lin et al., 2013] is used. GAP calculates the global spatial information by averaging the features across the spatial dimensions, which results in a statistic $\mathbf{z} \in \mathbb{R}^{c}$ (*squeeze*). Each m-element of the statistic is given by:

$$z_m = \mathbf{F}_{sq}(\mathbf{x}_m) = \frac{1}{h \times w} \sum_{i=1}^{h} \sum_{j=1}^{w} \mathbf{x}_m(i,j). \tag{6.1}$$

Notice that this operation is parameter-free and applies a dimensionality reduction; thus, it reduces each feature map $\mathbf{x}_m \in \mathbb{R}^{h \times w}$ to a single scalar value z_m.

Excitation Operation

The excitation operation is designed to reduce channel-wise feature complexity and improve generalization. To accomplish this, a simple gating mechanism $g(\cdot,\mathbf{W})$ is applied, defined as:

$$\mathbf{s} = \mathbf{F}_{ex}(\mathbf{z},\mathbf{W}) = \sigma(g(\mathbf{z},\mathbf{W})) = \sigma(\mathbf{W}_2\delta(\mathbf{W}_1\mathbf{z})), \tag{6.2}$$

where σ and δ refer to the Sigmoid and ReLU activation function, respectively, and noticing that $\sum_{m=1}^{c} s_m = 1$. The gating mechanism acts as a bottleneck with two fully connected layers $\mathbf{W}_1 \in \mathbb{R}^{c \times \frac{c}{r}}$ and $\mathbf{W}_2 \in \mathbb{R}^{\frac{c}{r} \times c}$. Here, the parameter r is a reduction ratio controlling the number of parameters of the SE block. In such a way, a Squeeze–Excitation operation $\mathbf{SE}(\cdot,\mathbf{W}) : \mathbb{R}^{h \times w \times c} \rightarrow \mathbb{R}^{1 \times 1 \times c}$ can be defined as:

$$\mathbf{s} = \mathbf{SE}(\mathbf{X},\mathbf{W}) = \mathbf{F}_{ex}(\mathbf{F}_{sq}(\mathbf{X}),\mathbf{W}). \tag{6.3}$$

Finally, the obtained values $\mathbf{s}$ are used to weight the input feature maps $\mathbf{X}$, resulting in a learnable recalibration that emphasizes or ignores specific channels. The rescaling procedure is performed by:

$$\hat{\mathbf{x}}_m = \mathbf{F}_{scale}(\mathbf{x}_m, s_m) = s_m \mathbf{x}_m, \tag{6.4}$$

where $\mathbf{F}_{scale}(\mathbf{x}_m, s_m)$ is a channel-wise multiplication between the feature map $\mathbf{x}_m \in \mathbb{R}^{h \times w}$ and the scalar s_m.

6.1.2 | Efficient Channel Attention

Wang et al. [2020] proposed an Efficient Channel Attention (ECA) attention mechanism based on SE blocks without dimensionality reduction. This approach uses a local cross-channel interaction strategy, enabling each channel of a given input feature map $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$ to have interdependence on every other channel within a small local group.

Local Cross-Channel Interaction

Once computed the statistic vector $\mathbf{z} \in \mathbb{R}^c$ by Equation (6.1), channel attention can be learned by:

$$\mathbf{s} = \sigma\left(\mathbf{W}\mathbf{z}\right), \tag{6.5}$$

where $\mathbf{W} \in \mathbb{R}^{c \times c}$ parameter matrix. Aiming at guaranteeing both efficiency and effectiveness, only local cross-channel interaction is considered between z_m and its k neighbors, *i.e.*,

$$s_m = \sigma\left(\sum_{j=1}^{k} w_m^j z_m^j\right), \quad z_m^j \in \Omega_m^k, \tag{6.6}$$

where Ω_m^k indicates the set of k adjacent channels of z_m. Furthermore, if the channels share the same weights w_m, the parameters can be reduced from ck to k. Thus, notice that such a strategy resembles a 1D convolution operation with a kernel size of k, such as:

$$\mathbf{s} = \sigma\left(f_{\text{conv1D}}(\mathbf{z}; k)\right). \tag{6.7}$$

The size of the kernel is obtained adaptive and proportional to the number of channels c as follows:

$$k = \left|\frac{\log_2(c)}{\gamma} + \frac{\beta}{\gamma}\right|_{\text{odd}}, \tag{6.8}$$

here $|\cdot|_{\text{odd}}$ indicates the nearest odd number, $\beta = 1$ and $\gamma = 2$. The weighting process is carried out like the SE module, obtaining each re-calibrated channel by $\hat{\mathbf{x}}_m = s_m \mathbf{x}_m$.

6.1.3 | Convolutional Block Attention

The Convolutional Block Attention Module (CBAM) [Woo et al., 2018] consists of two sub-modules, the Channel Attention Module (CAM) and the Spatial Attention Module (SAM). Given an input feature map $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$, CBAM generates a refined feature map $\hat{\mathbf{X}} \in \mathbb{R}^{h \times w \times c}$ by inferring attention maps along the channel and spatial dimensions.

Channel Attention Module

The CAM in CBAM is an extension of the SE module, now using a max-pooling and an average-pooling operation to generate two spatial feature vectors: $\mathbf{z}^c_{\text{max}}$ and $\mathbf{z}^c_{\text{avg}} \in \mathbb{R}^c$, respectively. Notice that $\mathbf{z}^c_{\text{avg}}$ is given by Equation (6.1), such as $\mathbf{z}^c_{\text{avg}} = \mathbf{F}_{sq}(\mathbf{X})$, and each element of $\mathbf{z}^c_{\text{max}}$ by:

$$z_m = \mathbf{F}_{max}(\mathbf{x}_m) = \max_m \mathbf{x}_m. \tag{6.9}$$

These feature vectors are then passed through a shared Multi-Layer Perceptron (MLP) network with a Squeeze-and-Excitation mechanism, with a ReLU activation function in-between. Keep in mind that the weights of the MLP network are shared between the two input feature vectors, allowing them to influence the channel-wise attention weights jointly. Hence, the channel attention map $\mathbf{s}^c \in \mathbb{R}^c$ is computed as

$$\mathbf{s}^c = \mathbf{F}_{\text{CAM}}(\mathbf{z}^c_{\text{avg}}, \mathbf{z}^c_{\text{max}}, \mathbf{W}) = \sigma\left(\mathbf{W}_2\, \delta(\mathbf{W}_1 \mathbf{z}^c_{\text{avg}}) + \mathbf{W}_2\, \delta(\mathbf{W}_1 \mathbf{z}^c_{\text{max}})\right). \tag{6.10}$$

It is noteworthy that the number of parameters of the CAM is the same as the SE attention module with $\frac{2c^2}{r}$, where r is the feature reduction ratio involving the MLP. Figure 6.2 illustrates the CAM procedure. At this stage, an intermediate refined feature map $\mathbf{X}'$ is obtained by the element-wise multiplication of the channel-wise attention vector $\mathbf{s}^c$ and the input feature map $\mathbf{X}$, as shown in the equation below:

$$\mathbf{X}' = \mathbf{s}^c \otimes \mathbf{X}, \tag{6.11}$$

where $\otimes$ is the Hadamard product.

Spatial Attention Module

The SAM, as shown in Figure 6.3, takes the refined intermediate features $\mathbf{X}'$ as input. First, two pooling operations are applied along the channel axis: an average-pooling and max-pooling, which generate two 2D maps denoted as $\mathbf{z}^s_{\text{avg}}, \mathbf{z}^s_{\text{max}} \in \mathbb{R}^{h \times w}$, respectively. The two maps are then concatenated along the channel axis and fed into a 2D convolutional layer, which computes the spatial attention feature map. The equation

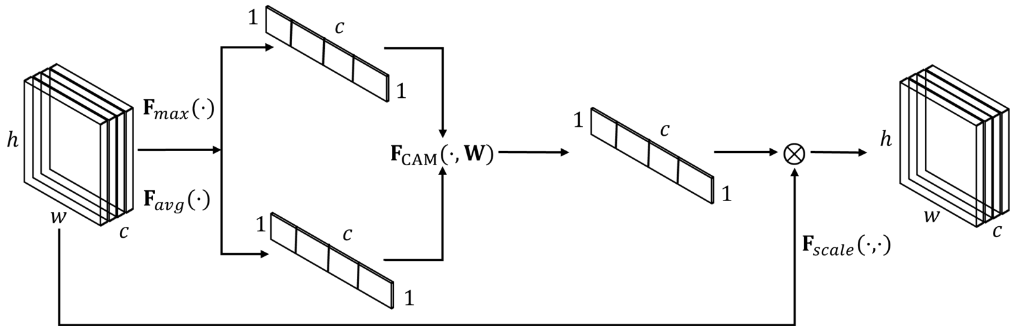

1
c
1
$\mathbf{F}_{max}(\cdot)$
h
$\mathbf{F}_{avg}(\cdot)$
w
c
1
c
1
$\mathbf{F}_{\mathrm{CAM}}(\cdot,\mathbf{W})$
1
c
1
⊗
h
w
c
$\mathbf{F}_{scale}(\cdot,\cdot)$

Figure 6.2: *Channel Attention block. The sub-module includes two pooling $\mathbf{F}_{max}, \mathbf{F}_{avg}$ layers, and a shared MLP $\mathbf{F}_{CAM}$ to exploit the inter-channel relationship of the features.*

for computing spatial attention is:

$$\mathbf{s}^s = \sigma\left(f_{\text{conv2D}}(\mathbf{z}^s_{\text{avg}} \oplus \mathbf{z}^s_{\text{max}}; k)\right), \tag{6.12}$$

where k is the filter size, set as $k = 7 \times 7$ by default, and $\oplus$ represents the concatenation operation.

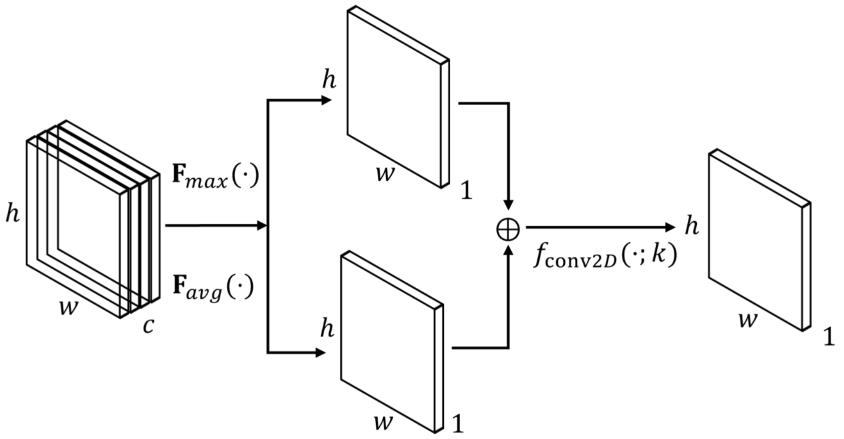

h
$\mathbf{F}_{max}(\cdot)$
$\mathbf{F}_{avg}(\cdot)$
w
c
h
w
1
h
w
1
⊕
$f_{\mathrm{conv2D}}(\cdot;k)$
h
w
1

Figure 6.3: *Spatial Attention block. The sub-module includes two pooling $\mathbf{F}_{max}, \mathbf{F}_{avg}$ layers along the channel axis and a 2D convolution layer to capture the spatial relationship of the features.*

6.2 | Related Work

As discussed in previous chapters, different deep learning approaches have been used to develop strategies to detect stenosis in XCA images through object-based or patch-based models. These methods have shown notable performance; nevertheless, object-based approaches are limited to detecting a single stenosis case in the whole image. Meanwhile, patch-based methodologies are restricted to detecting small stenotic regions (*i.e.*, based on the patch size). Moreover, both approaches take as their backbone network architectures designed for the ImageNet dataset, changing only the head of the model.

Additionally, to only change the top layers to match the target domain, Moon et al. [2021] utilized a pre-trained Inception-v3 model to classify stenosis in XCA images, but after each inception module, a CBAM is included to enhance the channel and spatial information. Only a subset of images extracted through automatic key-frame detection algorithms is kept to train. Next, each image feeds the attention CNN model to classify the stenosis cases. For every key-frame, distinct types of augmentation strategies were deployed and evaluated. Pang et al. [2021] proposed a two-stage object detector based on a ResNet50 architecture. Firstly, the feature maps were extracted, generating candidate boxes. Secondly, the candidate boxes were classified as stenosis or non-stenosis, employing a multi-head attention mechanism [Vaswani et al., 2017]. Finally, feature extraction and fusion sequences were introduced to correlate the candidate boxes from consecutive XCA frames to increase classification accuracy.

In the medical domain, different attention mechanisms have been used within the model to improve the network capabilities. For instance, Lu et al. [2022] presented a CAD system combined with a 3D residual region proposal network and the SE blocks for pulmonary nodule detection in CT images. This modified model can detect more suspicious regions or nodules than vanilla models. Similarly, Gong et al. [2019] proposed a novel approach employing a 3D CNN based on SE and residual networks for pulmonary nodule detection. Specifically, a 3D region proposal network with a U-Net-like structure was designed for detecting pulmonary nodule candidates. Then, for the subsequent false-positive reduction, the classifier model was employed to discriminate the true nodules from candidates accurately. In addition, the classification performance was boosted by adaptively recalibrate the channel-wise residual feature responses. Also, Cao et al. [2022] introduced a semantic segmentation deep glioma model. The model used an encoder-decoder structure, where the encoder part uses an improved Xception backbone network. In the improved Xception backbone network, CBAM is added after each convolutional layer, thereby improving the segmentation accuracy.

6.3 | Lightweight Residual Attention Networks

The proposed Lightweight Residual Attention Networks (LRA-Nets) consist of SE, ECA, or CBAM attention layers and Depthwise Separable Convolution (DSC) with residual connections layers, as illustrated in Figure 6.4. The network follows the structure of ResNet, where residual connections accelerate the training efficiency and resolve the gradient degradation problem. Moreover, a pruning strategy was employed in the convolutional layers by removing kernels. Thus unnecessary parts of neural networks are discarded.

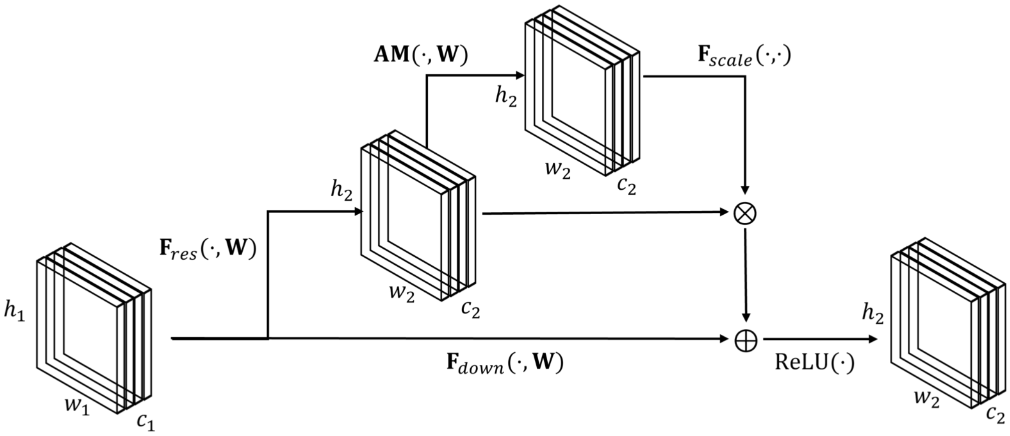

$\mathbf{AM}(\cdot,\mathbf{W})$
$\mathbf{F}_{scale}(\cdot,\cdot)$
h_2
w_2
c_2
h_2
w_2
c_2
$\mathbf{F}_{res}(\cdot,\mathbf{W})$
h_1
w_1
c_1
$\otimes$
$\oplus$
$\mathbf{F}_{down}(\cdot,\mathbf{W})$
ReLU(·)
h_2
w_2
c_2

*Figure 6.4: Attention ResNet. A Residual block within an attention module (**AM**) enhances the feature maps of the block.*

6.3.1 | Depthwise Separable Convolution

Let $f_{conv}(\cdot,\mathbf{W}) : \mathbb{R}^{h_1\times w_1\times c_1} \to \mathbb{R}^{h_2\times w_2\times c_2}$ be a standard convolution operation (without dilation) that takes as input $\mathbf{X}^{in}$ and produces $\mathbf{X}^{out}$ parameterized by the kernel $\mathbf{W} \in \mathbb{R}^{k\times k\times c_1\times c_2}$ computed as:

$$\mathbf{x}_{c_2}^{out}(i,j) = f_{conv}(\mathbf{x}_{c_1}^{in},\mathbf{W}) = \sum_{u=1}^{k}\sum_{v=1}^{k}\sum_{m=1}^{c_1}\mathbf{W}_m(i,j) * \mathbf{x}_m^{in}(i+u,j+v), \tag{6.13}$$

where $*$ represents the convolution operation and k the filter size, DSC factorizes a standard convolution by two independent convolutions: (1) depthwise convolution and (2) point-by-point convolution (1×1 convolution), as shown in Figure 6.5. The depthwise convolution $f_{dw-conv}(\cdot,\mathbf{W}) : \mathbb{R}^{h_1\times w_1\times c_1} \to \mathbb{R}^{h_1\times w_1\times c_1}$ decoupled the input feature map from its channels, applying a single filter to each input channel, as follows:

$$\mathbf{x}_{c_1}^{dw}(i,j) = f_{dw-conv}(\mathbf{x}_{c_1}^{in},\mathbf{W}) = \sum_{u=1}^{k}\sum_{v=1}^{k}\mathbf{W}_m(i,j) * \mathbf{x}_m^{in}(i+u,j+v). \tag{6.14}$$

Then, the pointwise $f_{pw-conv}(\cdot, \mathbf{W}) : \mathbb{R}^{h_1 \times w_1 \times c_1} \rightarrow \mathbb{R}^{h_2 \times w_2 \times c_2}$ convolution combines the features of each channel through a 1×1 standard convolution, such as:

$$\mathbf{x}_{c_2}^{out}(i, j) = f_{pw-conv}(\mathbf{x}_{c_1}^{dw}, \mathbf{W}) = \sum_{m=1}^{c_1} \mathbf{W}_m * \mathbf{x}_m^{dw}(i, j). \tag{6.15}$$

This factorization reduces the number of parameters and computation operations.

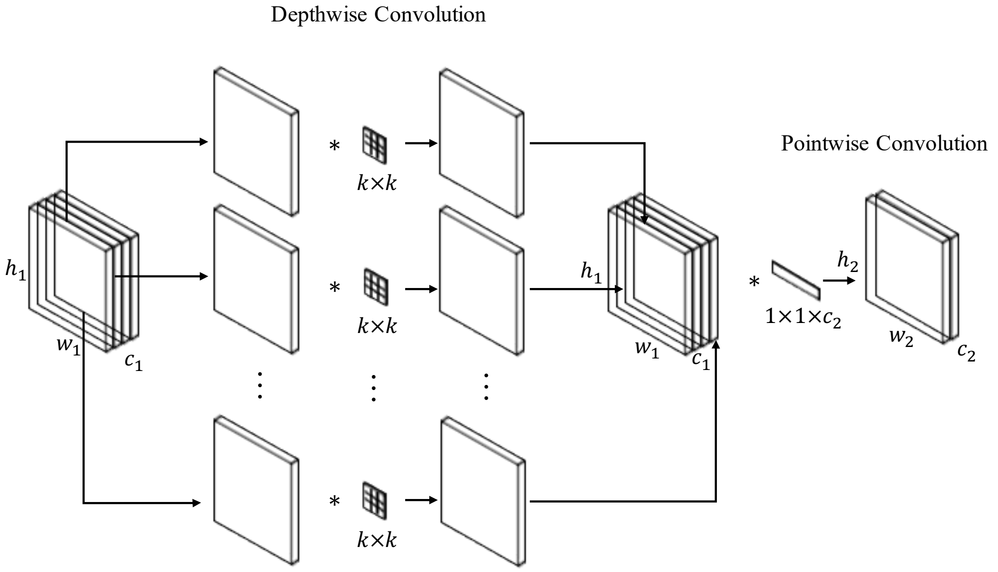


Figure 6.5: Depthwise Separable Convolution. A depthwise convolution and a point-by-point convolution factorize a standard convolution.

6.3.2 | Pruning Convolutional Kernels

Kernel pruning methods have been proposed to speed up (simplify) pre-trained CNNs models [Li et al., 2019; Osaku et al., 2021]. However, their effectiveness tends to be below the original one and requires additional fine-tuning steps. In this sense, it is straightforward to prune a scratch model where the channel and spatial relationship have not been learned yet.

Let be a convolutional layer that uses the filters $\mathbf{W} \in \mathbb{R}^{k \times k \times c \times N}$. Here, c is the number of input feature maps, N is the number of filters (*i.e.*, the number of output feature maps), and k are the height and width of a (square) filter, respectively. A simple but effective kernel pruning strategy is to reduce the number of filters by a ratio $p \geq 1$ such that $N' = \frac{N}{p}$. For instance, in a ResNet18 with 64, 128, 256, and 512 kernels per residual block, each residual block will be reduced equally by a pruning ratio p in order to maintain their baseline structure employed for the ImageNet challenge.

6.4 | Results and Discussion

6.4.1 | Implementation Details

The proposed networks took as a backbone network the ResNet18, which is mainly characterized by consisting of one 7 × 7 convolutional layer, with a stride of two pixels, followed by a max-pooling of size two; four residual blocks within 64, 128, 256, and 512 kernels, respectively, come after. Then, redundant kernels were removed in the convolutional layers to obtain a smaller model. Similarly, the first convolution layer kernel was changed for a 3 × 3, and the first max-pooling was removed.

The training process employs the Stochastic Gradient Descent with Momentum (SGDM) optimizer with a learning rate of 10^{-3} and a momentum of 0.9. The model was trained with a batch size of 32 for 100 epochs minimizing the Cross-Entropy Loss. The model was implemented using the PyTorch framework, and the experiments ran on Google's cloud servers, including a Tesla P4 GPU with 2560 CUDA cores and 8 GB of RAM.

Moreover, k-fold cross-validation (5-fold) was set following an 80:20 ratio from the validation subset. The validation step allows for saving the best weights during the training process.

6.4.2 | Ablation Study

In order to select the dilation ratios for the attention modules and the pruning ratio, an ablation study is first conducted using the Tree-structured Parzen Estimator (TPE) algorithm [Bergstra et al., 2011, 2013], minimizing the validation Cross-Entropy Loss of the first fold.

Table 6.1 summarizes the obtained LRA-Net architectures parameters. Notice that these models employed the DSC and the pruning kernel strategy. This table shows that the three attention-based models require a pruning ratio of two; thus, the number of kernels was reduced by 50%. Also, each model's default attention ratio (16) from the vanilla attention models is reduced. In such a way, from more than 11 M parameters of the vanilla attention ResNet18 models, the Light-weighted versions are 27.5x smaller, with around 0.4 M parameters.

6.4.3 | Stenosis Classification Performance Comparison

The proposed LRA-Net architectures drastically reduce the number of parameters of the networks and boost the performance metrics when trained from scratch. First, a

Table 6.1: *ResNet18 attention ablation study. The L before the attention module name (i.e., LECA) stands for Lightweight, that is the proposed model. n/a stands for Not Applicable.*

Attention	Attention Ratios	Pruning ratio	Best Val loss	Parameters [M]
Vanilla ECA	n/a	n/a	–	11.18
Vanilla SE	16, 16, 16, 16	n/a	–	11.27
Vanilla CBAM	16, 16, 16, 16	n/a	–	11.27
LECA	n/a	2	0.0716	0.381
LSE	13, 7, 2, 14	2	0.0807	0.434
LCBAM	12, 14, 3, 9	2	0.0902	0.434

comparative study evaluated vanilla attention ResNet models trained from scratch and pre-trained with the ImageNet dataset. Moreover, the best models obtained from the ablation study were compared against vanilla attention ResNet models trained from scratch. All the experiments employed the DDSS dataset containing 125 positive stenosis cases and 1,394 negative cases of size 32 × 32. Also, a 5-fold cross-validation strategy was applied.

Tables 6.2 to 6.4 show the comparative study for vanilla attention models when the backbone network was taken pre-trained with the ImageNet and trained from scratch. In these comparative studies, different reduction ratios were evaluated for the attention modules, keeping the same ratio from all the modules as typically employed.

For the vanilla SEResNet18 (Table 6.2), with a reduction ratio of 8 when the model was trained from scratch, achieved four-of-five best metrics, with an accuracy of 0.9224, a sensitivity of 0.2880, precision of 0.5245, and F_1-score of 0.3508. On the other hand, when the backbone network was taken pre-trained, the SEResNet18 achieved three-of-five best classification metrics (0.9704/0.7520/0.8066 for accuracy, sensitivity, and F_1-score, respectively) when the reduction ratio was set to 16. For specificity and precision, it obtained the second best (0.9900 and 0.8704).

The vanilla CBAMResNet18 trained from scratch (Table 6.3) obtained the best specificity and precision with a reduction ratio of 8, with 0.9871 and 0.6213, respectively. When the backbone model was pre-trained, it attained a substantial boost in all the metrics, mainly when the reduction ratio was 16, and the best accuracy (0.9684), sensitivity (0.7200), and F_1-score (0.7832) were obtained. Also, the second-best precision (0.8851) and the third-best specificity (0.9907). The best of these two metrics was achieved with a reduction ratio of 2 and 1, respectively.

Table 6.2: *Vanilla SE ResNet18 classification performance. The mean and standard deviation of each metric are shown.*

Pretrained	Reduction ratio	Accuracy	Sensitivity	Specificity	Precision	F_1-score	Parameters [M]
✗	1	0.9224 ± 0.0082	0.1360 ± 0.1592	0.9928 ± 0.0060	0.3629 ± 0.3039	0.1860 ± 0.2065	12.57
	2	0.9184 ± 0.0064	0.1760 ± 0.1444	0.9849 ± 0.0077	0.3932 ± 0.2344	0.2337 ± 0.1761	11.88
	4	0.9184 ± 0.0048	0.1440 ± 0.1061	0.9878 ± 0.0066	0.4343 ± 0.2334	0.2083 ± 0.1341	11.53
	8	0.9224 ± 0.0133	0.2880 ± 0.1849	0.9792 ± 0.0105	0.5245 ± 0.1477	0.3508 ± 0.1842	11.35
	16	0.9151 ± 0.0092	0.1520 ± 0.1143	0.9835 ± 0.0105	0.4181 ± 0.2260	0.2103 ± 0.1420	11.27
✓	1	**0.9704** ± 0.0066	0.7440 ± 0.0480	**0.9907** ± 0.0058	**0.8819** ± 0.0656	0.8054 ± 0.0418	12.57
	2	0.9678 ± 0.0044	0.7280 ± 0.0466	0.9892 ± 0.0039	0.8611 ± 0.0412	0.7875 ± 0.0296	11.88
	4	0.9684 ± 0.0045	0.7440 ± 0.0742	0.9885 ± 0.0042	0.8570 ± 0.0378	0.7932 ± 0.0372	11.53
	8	0.9658 ± 0.0090	0.7280 ± 0.0588	0.9871 ± 0.0087	0.8436 ± 0.0849	0.7785 ± 0.0533	11.35
	16	**0.9704** ± 0.0051	**0.7520** ± 0.0392	0.9900 ± 0.0027	0.8704 ± 0.0350	**0.8066** ± 0.0347	11.27

The vanilla ECAResNet18 does not need a reduction ratio to be specified, as seen in Table 6.4. Thus, the pre-trained version obtained the best evaluation metric, with an accuracy of 0.9704, a sensitivity of 0.7600, an specificity of 0.9892, a precision of 0.8675, and F_1-score of 0.8085.

In Summary, taking a pre-trained backbone model improves the classification performance for the three attention-based models. Concerning these models, the Vanilla ECAResNet18 obtained the overall best accuracy and sensitivity (0.9704 and 0.7600), the Vanilla CBAMResNet18 the broad best specificity (0.9928) and precision (0.9156), with a dilation ratio of 1 and 2, respectively, and the overall best F_1-score (0.8066) by the Vanilla SEResNet18 with a dilation ratio of 16.

It is important to point out that the CBAM and SE versions, with a reduction ratio of 16, required 11.27 M of parameters, while the ECA version required 11.18 M. Also, notice that low-performance metrics, particularly sensitivity, precision, and F_1-score, were obtained when the models were trained from scratch.

For such a reason, light-weighted models were proposed to improve the

Table 6.3: *Vanilla CBAMResNet18 classification performance. The mean and standard deviation of each metric are shown.*

Pretrained	Reduction ratio	Accuracy	Sensitivity	Specificity	Precision	F_1-score	Parameters [M]
✗	1	0.9349 ± 0.0118	0.5360 ± 0.1835	0.9706 ± 0.0086	0.6114 ± 0.0695	0.5594 ± 0.1309	12.57
	2	0.9250 ± 0.0092	0.2560 ± 0.2080	0.9849 ± 0.0125	0.4809 ± 0.2580	0.3129 ± 0.2119	11.87
	4	0.9230 ± 0.0094	0.2240 ± 0.1727	0.9857 ± 0.0104	0.4489 ± 0.2709	0.2861 ± 0.2092	11.53
	8	0.9263 ± 0.0097	0.2480 ± 0.1462	0.9871 ± 0.0066	0.6213 ± 0.1201	0.3344 ± 0.1617	11.35
	16	0.9368 ± 0.0214	0.4000 ± 0.2896	0.9849 ± 0.0097	0.6060 ± 0.2788	0.4527 ± 0.2900	11.27
✓	1	0.9678 ± 0.0038	0.6880 ± 0.0392	**0.9928** ± 0.0039	0.8991 ± 0.0464	0.7780 ± 0.0270	12.57
	2	0.9671 ± 0.0036	0.6640 ± 0.0320	0.9943 ± 0.0037	**0.9156** ± 0.0526	0.7685 ± 0.0241	11.87
	4	0.9632 ± 0.0038	0.7200 ± 0.0438	0.9849 ± 0.0027	0.8115 ± 0.0270	0.7622 ± 0.0291	11.53
	8	0.9625 ± 0.0026	0.6800 ± 0.0253	0.9878 ± 0.0043	0.8371 ± 0.0458	0.7490 ± 0.0125	11.35
	16	**0.9684** ± 0.0057	**0.7200** ± 0.1315	0.9907 ± 0.0058	0.8851 ± 0.0490	**0.7832** ± 0.0602	11.27

Table 6.4: *Vanilla ECAResNet18 classification performance. The mean and standard deviation of each metric are shown.*

Pretrained	Accuracy	Sensitivity	Specificity	Precision	F_1-score	Parameters [M]
✗	0.9257 ± 0.0147	0.3360 ± 0.2537	0.9785 ± 0.0085	0.5123 ± 0.1520	0.3797 ± 0.2261	11.18
✓	**0.9704** ± 0.0029	**0.7600** ± 0.0358	**0.9892** ± 0.0045	**0.8675** ± 0.0470	**0.8085** ± 0.0170	11.18

performance of scratch models and, simultaneously, reduce the number of parameters. Numerical results are shown in Table 6.5, where a comparative study is performed with the best models obtained in the ablation study and the best vanilla attention models (trained from scratch).

The light-weighted models (LECA, LSE, LCBAM) optimized the reduction ratios of each residual block and the pruning ratio of the convolutional layers. Bear in mind that the convolutional layers perform a deep-wise separable convolution. The LECA achieved the best accuracy (0.9625), specificity (0.9892), and precision (0.8440) with gains of 4%, 1%, and 33%, respectively, concerning the vanilla ECA variant. For sensitivity and F_1-score, the LCBAM model obtained the best values, with 0.7600 and 0.7625, representing a boost of 23% and 21%, respectively, compared to the vanilla CBAM.

Table 6.5: *ResNet18 attention comparative study. The L before the attention module name (i.e., LECA) stands for Lightweight, which is the proposed model, and n/a for not applicable. The mean and standard deviation of each metric are shown.*

Attention	Attention Ratios	Pruning ratio	Accuracy	Sensitivity	Specificity	Precision	F_1-score
Vanilla ECA	n/a	n/a	0.9257 ± 0.0147	0.3360 ± 0.2537	0.9785 ± 0.0085	0.5123 ± 0.1520	0.3797 ± 0.2261
Vanilla SE	8,8,8,8	n/a	0.9224 ± 0.0133	0.2880 ± 0.1849	0.9792 ± 0.0105	0.5245 ± 0.1477	0.3508 ± 0.1842
Vanilla CBAM	1,1,1,1	n/a	0.9349 ± 0.0118	0.5360 ± 0.1835	0.9706 ± 0.0086	0.6114 ± 0.0695	0.5594 ± 0.1309
LECA	n/a	2	**0.9625** ± 0.0099	0.6640 ± 0.0933	**0.9892** ± 0.0032	**0.8440** ± 0.0517	0.7416 ± 0.0764
LSE	13, 7, 2, 14	2	0.9559 ± 0.0087	0.6640 ± 0.1203	0.9821 ± 0.0082	0.7762 ± 0.0691	0.7072 ± 0.0797
LCBAM	12, 14, 3, 9	2	0.9612 ± 0.0070	**0.7600** ± 0.0669	0.9792 ± 0.0066	0.7695 ± 0.0519	**0.7625** ± 0.0441

6.4.4 | Class Activation Maps Visualization

To visually evaluate the attention modules, the GradCAM method provides a heat map highlighting the most important regions in the image in red tones and low attention regions in purple tones for predicting stenosis.

Figure 6.6 illustrates the GradCAM response for the SE attention modules when trained from scratch (a), with the backbone network pre-trained (b), and the proposed light-weighted approach (c). It can be seen that when the model was trained from scratch, (a) the high attention regions lie over the corners of the images. Attention improves when the backbone network was taken pre-trained (b), obtaining blood vessel regions with high attention and background zones with low attention. The light-weighted model presented greater attention over the blood vessel with non-false positive or negative cases, producing more accurate attention zones. Also, notice that the probability of stenosis for true positive cases in the light-weighted SE model is higher than the trained from scratch and pre-trained models and lower for true negative cases.

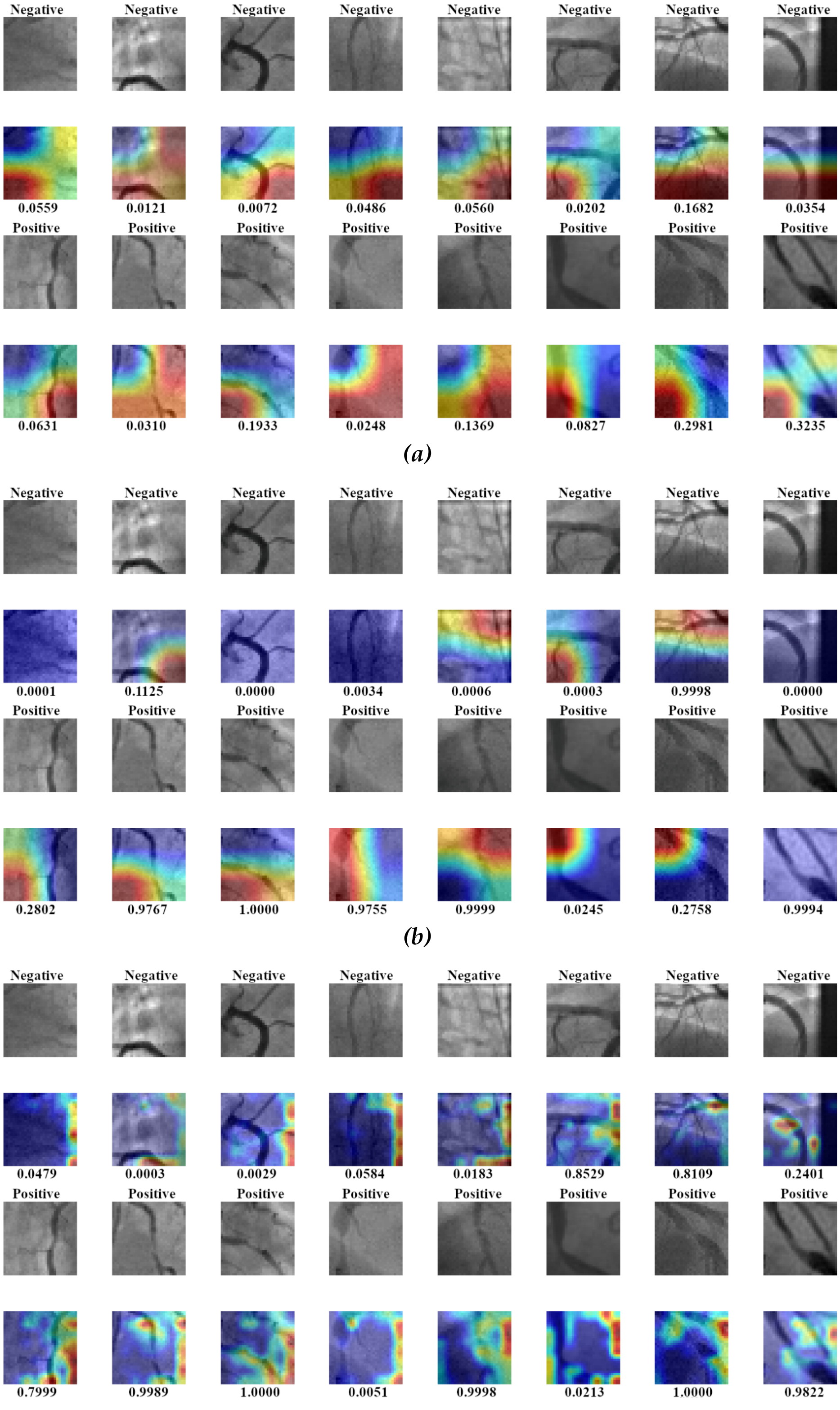
For the CBAMResNet18 variants (see Figure 6.7), the GradCAM put high attention regions over non-blood vessels when the model was trained from scratch. These regions were refined when the pre-trained model was fine-tuned, showing more accurate high attention zones over the blood vessel pixels. In the case of the light-weighted model, a higher probability for stenosis cases and a lower one for negative ones, than in the vanilla configurations. This is visually reflected in a detailed gradient map with red tones over blood vessel pixels and stenosis regions.

The third attention variant, the ECAResNet18, and the GradCAM are shown in Figure 6.8, which featured more isolated high-attention regions for the light-weighted model. These regions are located over blood vessel pixels. In addition, the LECAResNet18 Figure 6.8(c) showed low attention to the negative stenosis cases in the background zones of the image, contrary to the vanilla models (see Figure 6.8(a) and (b)), where attention regions are not well defined over the blood vessel pixel.

6.5 | Conclusion

This chapter proposed Lightweight Residual Attention Networks (LRA-Nets) to classify stenosis cases from XCA images. The models consist of three main elements, a DSC, a pruning convolution kernel ratio, and an attention module: SE, ECA, and CBAM, which reflect high classification rates with lower computational requirements regarding the required parameters. The proposed model is 27.5× smaller than Vanilla Attention ResNet18. The experimental results demonstrate that LRA-Nets consistently outperformed Residual models with or without attention mechanisms. Additionally, the individual selection of dilation ratios for the attention blocks improved the classification performance, including a smaller dilation ratio than the default configuration. Also, the pruning ratio drastically reduced the required kernels by each convolution layer. In particular, more significant boosts were achieved with the LECA, achieving the best accuracy (0.9625), specificity (0.9892), and precision (0.8440) with gains of 4%, 1%, and 33%, respectively, concerning the vanilla ECA variant. For sensitivity and F_1-score, the LCBAM model obtained the best values, with 0.7600 and 0.7625, representing a boost of 23% and 21%, respectively, compared to the vanilla CBAM. Moreover, the LECAResNet18 GradCAM maps retrieved a refined region proposal of the stenosis location, which could support the physician's decision-making process.

Although the recognition rates are high, further improvements can be explored, such as an object-based recognition system and detecting stenosis cases from the full XCA test. A future direction of this work concerning model compression may be to analyze other approaches, such as quantization, different low-rank-tensor decomposition, and knowledge distillation.


Negative Negative Negative Negative Negative Negative Negative Negative
0.0559 0.0121 0.0072 0.0486 0.0560 0.0202 0.1682 0.0354
Positive Positive Positive Positive Positive Positive Positive Positive
0.0631 0.0310 0.1933 0.0248 0.1369 0.0827 0.2981 0.3235
Negative Negative Negative Negative Negative Negative Negative Negative
0.0001 0.1125 0.0000 0.0034 0.0006 0.0003 0.9998 0.0000
Positive Positive Positive Positive Positive Positive Positive Positive
0.2802 0.9767 1.0000 0.9755 0.9999 0.0245 0.2758 0.9994
Negative Negative Negative Negative Negative Negative Negative Negative
0.0479 0.0003 0.0029 0.0584 0.0183 0.8529 0.8109 0.2401
Positive Positive Positive Positive Positive Positive Positive Positive
0.7999 0.9989 1.0000 0.0051 0.9998 0.0213 1.0000 0.9822

(a)

(b)

(c)

Figure 6.6: *GradCAM for different variants of SEResNet18: (a) Trained from scratch, (b) Pretrained, (c) Lightweighted.*

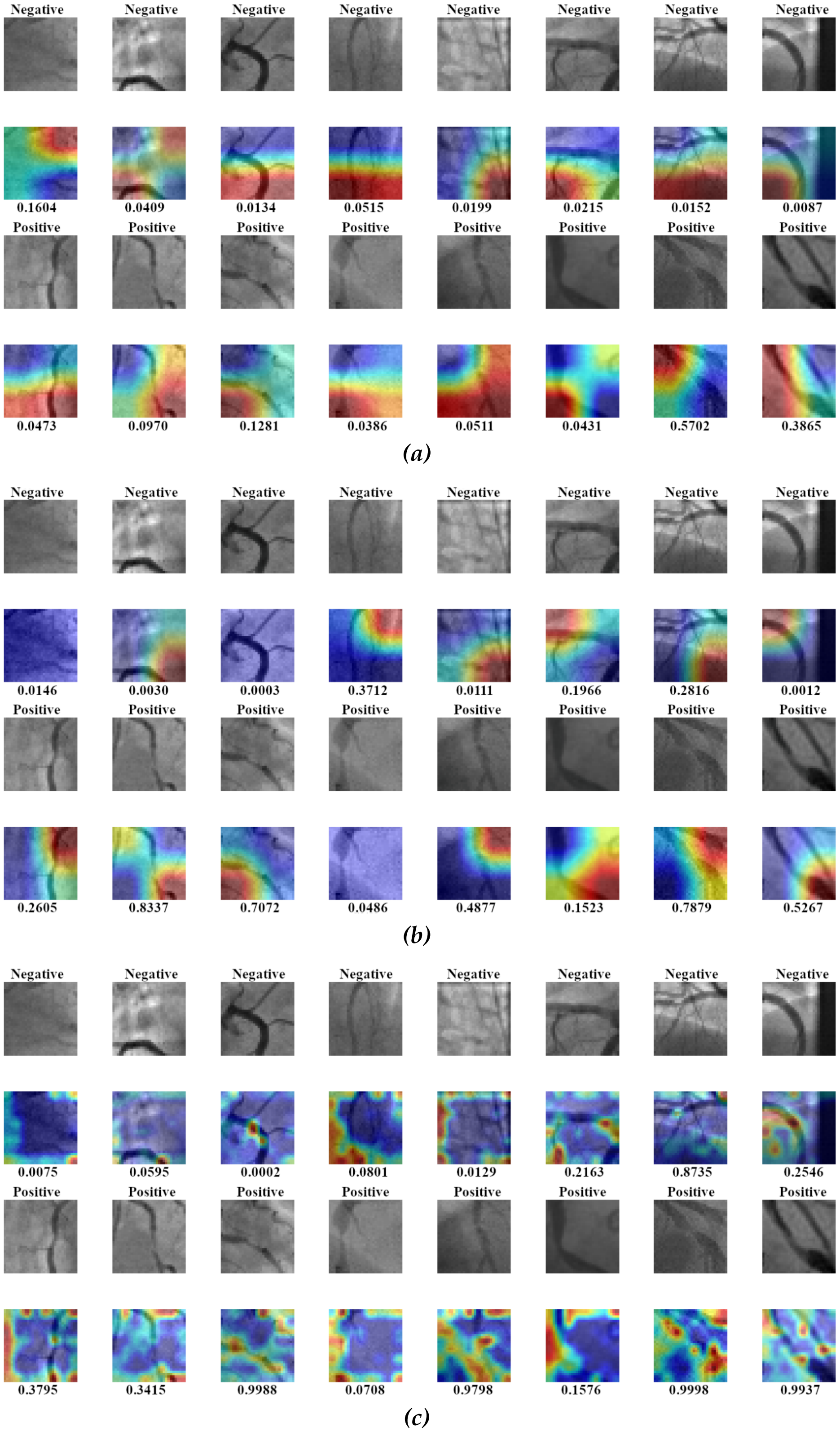

Negative Negative Negative Negative Negative Negative Negative Negative
0.1604 0.0409 0.0134 0.0515 0.0199 0.0215 0.0152 0.0087
Positive Positive Positive Positive Positive Positive Positive Positive
0.0473 0.0970 0.1281 0.0386 0.0511 0.0431 0.5702 0.3865
(a)
Negative Negative Negative Negative Negative Negative Negative Negative
0.0146 0.0030 0.0003 0.3712 0.0111 0.1966 0.2816 0.0012
Positive Positive Positive Positive Positive Positive Positive Positive
0.2605 0.8337 0.7072 0.0486 0.4877 0.1523 0.7879 0.5267
(b)
Negative Negative Negative Negative Negative Negative Negative Negative
0.0075 0.0595 0.0002 0.0801 0.0129 0.2163 0.8735 0.2546
Positive Positive Positive Positive Positive Positive Positive Positive
0.3795 0.3415 0.9988 0.0708 0.9798 0.1576 0.9998 0.9937
(c)

Figure 6.7: *GradCAM for different variants of CBAMResNet18: (a) Trained from scratch, (b) Pretrained, (c) Lightweighted.*

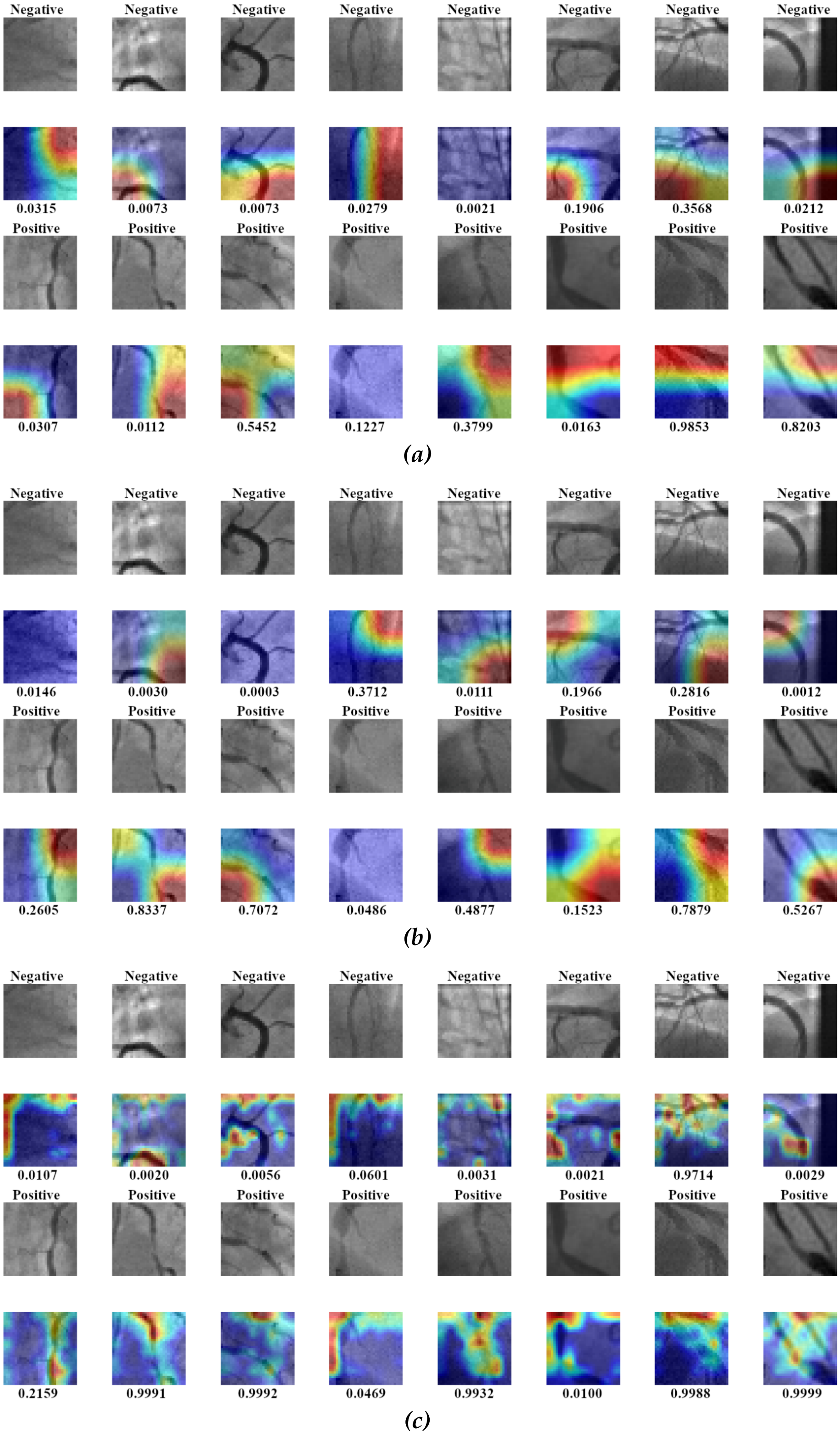

Negative Negative Negative Negative Negative Negative Negative Negative
0.0315 0.0073 0.0073 0.0279 0.0021 0.1906 0.3568 0.0212
Positive Positive Positive Positive Positive Positive Positive Positive
0.0307 0.0112 0.5452 0.1227 0.3799 0.0163 0.9853 0.8203
(a)
Negative Negative Negative Negative Negative Negative Negative Negative
0.0146 0.0030 0.0003 0.3712 0.0111 0.1966 0.2816 0.0012
Positive Positive Positive Positive Positive Positive Positive Positive
0.2605 0.8337 0.7072 0.0486 0.4877 0.1523 0.7879 0.5267
(b)
Negative Negative Negative Negative Negative Negative Negative Negative
0.0107 0.0020 0.0056 0.0601 0.0031 0.0021 0.9714 0.0029
Positive Positive Positive Positive Positive Positive Positive Positive
0.2159 0.9991 0.9992 0.0469 0.9932 0.0100 0.9988 0.9999
(c)

Figure 6.8: *GradCAM different variants of ECAResNet18: (a) Trained from scratch, (b) Pretrained, (c) Lightweighted.*

7

Conclusions

"Every real story is a never ending story."

— Michael Ende, *The Neverending Story*

In summary, this thesis explored different deep learning-based methods for stenosis detection in XCA images. The first method introduced a network-cut and fine-tuning approach for stenosis detection in XCA images. The method was evaluated through extensive numerical experiments based on 20 different setups for pre-trained networks (VGG16, ResNet50, and Inception-v3) with three different fine-tuning strategies. The optimal cut and fine-tuned layers were selected by minimizing the loss function. The results showed that employing this approach on a limited and unbalanced XCA dataset performed efficiently for stenosis detection. Furthermore, the proposed scheme allows accuracy, sensitivity, specificity, precision, and F_1-score improvement concerning vanilla pre-trained networks and configurations trained from scratch. Moreover, it allowed reducing the network complexity in terms of parameters.

The second method presented a Hybrid Classical-Quantum Network (HCQN) for stenosis detection. The framework involved connecting a QN to the head of a classical CNN to enhance the feature representation. The main contribution of this research was related to the QN architecture, where multiple (and smaller) VQCs can replace a single VQC. Additionally, to facilitate overall training convergence, a novel squeeze, scaling, and angle encoding process was introduced that maps the classical feature vector into a quantum network. Numerical results validate that the proposed hybrid model significantly improves the detection performance when the CNN sub-module is pre-trained with the ImageNet dataset, demonstrating the quantum computing potential. Furthermore, the proposed approach can be easily customized and integrated into any CNN architecture.

Next, a Hierarchical Bezier Generative Model (HBGM) was proposed to address the problem of a small and poorly diversified database for stenosis detection in XCA images. A large-scale labeled dataset consisting of 10k images was created using the proposed approach. Extensive experiments showed that pre-training ResNets using this dataset and a posterior fine-tuning with real XCA images achieved the best overall performance on two (of five) evaluation metrics and competitive results on the remainder. Moreover, it demonstrates the value of transferring the weights pre-trained using a more alike (artificial) dataset instead of the ImageNet dataset for stenosis detection tasks with only limited data available.

The last proposal was a Lightweight Residual Attention Network (LRA-Nets) to classify stenosis cases from XCA images. The models have three main elements: a DSC, a pruning convolution kernel ratio, and an attention module (SE, ECA, and CBAM). The proposed model is 27.5× smaller than Vanilla Attention ResNet18. However, the experimental results demonstrate that LRA-Nets consistently outperformed Residual models with or without attention mechanisms. Additionally, the individual selection of dilation ratios for the attention blocks contributed to the improved classification accuracy of the proposed LRA-Nets. As a result, the proposed model achieves high classification rates with lower computational requirements regarding the required parameters.

Overall, these methods provide insights into the potential of deep learning-based approaches to improve stenosis detection in XCA images and pave the way for future research in this field.

7.1 | Algorithmic Limitations

The proposed models, such as the fine-tuning approach and hybrid classical-quantum network, were developed using a limited and unbalanced dataset, which may not generalize well to other XCA datasets. Moreover, these models relied on pre-trained models from the ImageNet dataset, resulting in suboptimal GradCAMs. Although the heat maps provided insight into the areas of interest detected by the network, high relevance was assigned to regions without blood vessels. Despite the notorious classification improvement with the hybrid classical-quantum network, it relied on a quantum-device simulator that may contain bias.

The HBGM relied on handcrafted parameters, which could restrict the flexibility and control of the generated images. Furthermore, the FID between the generated and real images was not optimized as in the discriminator loss function of GANs. Finally, the proposed lightweight residual attention network was developed using attention modules intended for natural images, limiting its ability to exploit the unique features of XCA images.

7.2 | Future Work

Based on the limitations mentioned above, here are some potential avenues for future work:

- Exploring loss functions for unbalanced datasets, which would allow for better generalization of the proposed models.
- Propose a more efficient quantum encoding and quantum circuit design to improve the classical-quantum bottleneck.
- Develop new methods for generating synthetic data optimizing within the generative process to increase flexibility and control over the generated images.
- Design attention modules specifically designed for XCA images to improve the performance of the lightweight residual attention network.
- Explore the latest deep learning models (*i.e.*, Visual Transformers) in the XCA image domain.

7.3 | Scientific Contributions

The main contributions of each chapter and collaborations can be found in the following articles:

Articles in indexed journals

- **Ovalle-Magallanes, E**., Alvarado-Carrillo D. E., Avina-Cervantes, J. G., Cruz-Aceves, I., & Ruiz-Pinales, J. (2023). Quantum angle encoding with learnable rotation applied to quantum–classical convolutional neural networks. *Applied Soft Computing*, 141, 110307. https://doi.org/10.1016/j.asoc.2023.110307. **I.F.(2022) 8.7**.
- **Ovalle-Magallanes, E**., Avina-Cervantes, J. G., Cruz-Aceves, I., & Ruiz-Pinales, J. (2022). LRSE-Net: Lightweight Residual Squeeze-and-Excitation Network for

Stenosis Detection in X-ray Coronary Angiography. *Electronics*, 11(21), 3570. https://doi.org/10.3390/electronics11213570. **I.F.(2022) 2.9**.

- **Ovalle-Magallanes, E.**, Avina-Cervantes, J. G., Cruz-Aceves, I., & Ruiz-Pinales, J. (2022). Improving convolutional neural network learning based on a hierarchical bezier generative model for stenosis detection in X-ray images. *Computer Methods and Programs in Biomedicine*, 219, 106767. https://doi.org/10.1016/j.cmpb.2022.106767. **I.F.(2022) 6.1**.
- **Ovalle-Magallanes, E.**, Avina-Cervantes, J. G., Cruz-Aceves, I., & Ruiz-Pinales, J. (2022). Hybrid classical–quantum Convolutional Neural Network for stenosis detection in X-ray coronary angiography. *Expert Systems with Applications*, 189, 116112. https://doi.org/10.1016/j.eswa.2021.116112. **I.F.(2022) 8.5**.
- **Ovalle-Magallanes, E.**, Aldana-Murillo, N. G., Avina-Cervantes, J. G., Ruiz-Pinales, J., Cepeda-Negrete, J., & Ledesma, S. (2021). Transfer learning for humanoid robot appearance-based localization in a visual map. *IEEE Access*, 9, 6868-6877. https://doi.org/10.1109/ACCESS.2020.3048936. **I.F.(2022) 3.9**.
- **Ovalle-Magallanes, E.**, Avina-Cervantes, J. G., Cruz-Aceves, I., & Ruiz-Pinales, J. (2020). Transfer learning for stenosis detection in X-ray coronary angiography. *Mathematics*, 8(9), 1510. https://doi.org/10.3390/math8091510. **I.F.(2022) 2.4**.

Book Chapters

- **Ovalle-Magallanes, E.**, Alvarado-Carrillo, D.E., Avina-Cervantes, J.G., Cruz-Aceves, I., Ruiz-Pinales, J., Correa, R. (2023). Deep Learning-based Coronary Stenosis Detection in X-ray Angiography Images: Overview and Future Trends. *Artificial Intelligence and Machine Learning for Healthcare. Intelligent Systems Reference Library, Springer, Cham*, 229, 197-223. https://doi.org/10.1007/978-3-031-11170-9_8. **I.F.(2022) 0.85**

Articles in peer-reviewed journals

- **Ovalle-Magallanes, E.**, Alvarado-Carrillo, D. E., Avina-Cervantes, J. G., Cruz-Aceves, I., Ruiz-Pinales, J., & Contreras-Hernandez, J. L. (2022). Attention Mechanisms Evaluated on Stenosis Detection using X-ray Angiography Images. *Journal of Advances in Applied & Computational Mathematics*, 9, 62-75. https://doi.org/10.15377/2409-5761.2022.09.5.

Articles in conference proceedings

- Zambrano-Gutierrez, D. F., **Ovalle-Magallanes, E.**, Yanez-Borjas, J. J., Rostro-Gonzalez, H., & Avina-Cervantes, J. G. (2021, October). Path planning for mobile robots based on optimal interaction of electrostatic fields. *In Memorias del Congreso Nacional de Control Automático*, 115-120. Asociación de México de Control Automático.

- Alvarado-Carrillo, D. E., **Ovalle-Magallanes, E.**, & Dalmau-Cedeño, O. S. (2021, January). D-GaussianNet: Adaptive distorted Gaussian matched filter with convolutional neural network for retinal vessel segmentation. *In International Symposium on Geometry and Vision*, 378-392. Springer, Cham. https://doi.org/10.1007/978-3-030-72073-5_29. **I.F.(2022) 0.209**.

References

Erdi Acar and Ihsan Yilmaz. COVID-19 detection on IBM quantum computer with classical-quantum transfer learning. *Turkish Journal of Electrical Engineering & Computer Sciences*, 29(1):46–61, 2021. 10.3906/elk-2006-94.

Karol Antczak and Łukasz Liberadzki. Stenosis Detection with Deep Convolutional Neural Networks. In *MATEC Web of Conferences*, Volume 210, page 04001. EDP Sciences, 2018. 10.1051/matecconf/201821004001.

Karol Antczak and Łukasz Liberadzki. Deep Stenosis Detection Dataset. `https://github.com/KarolAntczak/DeepStenosisDetection`, Aug 2022.

Antonios P Antoniadis, Peter Mortier, Ghassan Kassab, Gabriele Dubini, Nicolas Foin, Yoshinobu Murasato, Andreas A Giannopoulos, Shengxian Tu, Kiyotaka Iwasaki, Yutaka Hikichi, et al. Biomechanical Modeling to Improve Coronary Artery Bifurcation Stenting: Expert Review Document on Techniques and Clinical Implementation. *Cardiovascular Interventions*, 8(10):1281–1296, 2015. 10.1016/j.jcin.2015.06.015.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein Generative Adversarial Networks. In *International Conference on Machine Learning*, pages 214–223, Sydney, Australia, Aug 2017. PMLR.

Lambros S Athanasiou, Dimitrios I Fotiadis, and Lampros K Michalis. *Atherosclerotic Plaque Characterization Methods Based on Coronary Imaging*. Academic Press, 1 edition, 2017.

Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. From generic to specific deep representations for visual recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 36–45, Boston, MA, USA, Jun 2015. 10.1109/CVPRW.2015.7301270.

James Bergstra and Yoshua Bengio. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, 13(2):281–305, 2012. 10.5555/2188385.2188395.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing Systems*, Volume 24, Red Hook, NY, USA, Dec 2011. Curran Associates Inc. URL https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf.

James Bergstra, Daniel Yamins, and David Cox. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. In Sanjoy Dasgupta and David McAllester, editors, *International Conference on Machine Learning*, Volume 28, pages 115–123, Jun 2013. URL https://proceedings.mlr.press/v28/bergstra13.html.

Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017. 10.1038/nature23474.

Ali Borji. Pros and cons of GAN evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019. 10.1016/j.cviu.2018.10.009.

Britannica, The Editors of Encyclopaedia. Coronary Heart Disease, Oct 2021. URL https://www.britannica.com/science/coronary-heart-disease.

Yuzhen Cao, Qinhao Zhang, Jinqiu Li, Yuhu Wang, Dongyi Liu, and Hui Yu. An automated segmentation model based on CBAM for MR image of glioma tumors. In *2nd International Conference on Bioinformatics and Intelligent Computing*, pages 385–388, Jan 2022. 10.1145/3523286.3524575.

Adithi Deborah Chakravarthy, Dilanga Abeyrathna, Mahadevan Subramaniam, Parvathi Chundi, Muhammad Sohail Halim, Murat Hasanreisoglu, Yasir J. Sepah, and Quan Dong Nguyen. An Approach Towards Automatic Detection of Toxoplasmosis using Fundus Images. In *2019 IEEE 19th International Conference on Bioinformatics and Bioengineering (BIBE)*, pages 710–717, Athens, Greece, Oct 2019. 10.1109/BIBE.2019.00134.

Chih-Feng Chang, Keng-Hao Chang, Chih-Hung Lai, Tzu-Hsiang Lin, Tsun-Jui Liu, Wen-Lieng Lee, and Chieh-Shou Su. Clinical outcomes of coronary artery bifurcation disease patients underwent Culotte two-stent technique: a single center experience. *BMC cardiovascular disorders*, 19(1):1–8, 2019. 10.1186/s12872-019-1192-2.

Claudio Chiastra, Francesco Iannaccone, Maik J Grundeken, Frank JH Gijsen, Patrick Segers, Matthieu De Beule, Patrick W Serruys, Joanna J Wykrzykowska, Antonius FW van der Steen, and Jolanda J Wentzel. Coronary fractional flow reserve measurements of a stenosed side branch: a computational study investigating the influence of the bifurcation angle. *Biomedical engineering online*, 15(1):1–16, 2016. 10.1186/s12938-016-0211-0.

Joseph Paul Cohen, Margaux Luck, and Sina Honari. Distribution Matching Losses Can Hallucinate Features in Medical Image Translation. In *Medical Image Computing and Computer Assisted Intervention*, pages 529–536, Granada, Spain, Sep 2018. Springer. 10.1007/978-3-030-00928-1_60.

Chao Cong, Yoko Kato, Henrique Doria Vasconcellos, Joao Lima, and Bharath Venkatesh. Automated Stenosis Detection and Classification in X-ray Angiography Using Deep Neural Network. In *International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1301–1308, San Diego, CA, USA, Nov 2019. IEEE. 10.1109/BIBM47256.2019.8983033.

Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*, pages 379–387, Red Hook, NY, USA, 2016. Curran Associates Inc.

Viacheslav V Danilov, Kirill Yu Klyshnikov, Olga M Gerget, Anton G Kutikhin, Vladimir I Ganyukov, Alejandro F Frangi, and Evgeny A Ovcharenko. Real-time coronary artery stenosis detection based on modern neural networks. *Scientific reports*, 11(1):1–13, 2021. 10.1038/s41598-021-87174-2.

Panagiotis Dimitrakopoulos, Giorgos Sfikas, and Christophoros Nikou. Wind: Wasserstein Inception Distance For Evaluating Generative Adversarial Network Performance. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 3182–3186. IEEE, 2020. 10.1109/ICASSP40776.2020.9053325.

Joachim Eckert, Marco Schmidt, Annett Magedanz, Thomas Voigtländer, and Axel Schmermund. Coronary CT angiography in managing atherosclerosis. *International Journal of Molecular Sciences*, 16(2):3740–3756, 2015. 10.3390/ijms16023740.

Debora Gil, Antonio Esteban-Lansaque, Sebastian Stefaniga, Mihail Gaianu, and Carles Sanchez. Data Augmentation from Sketch. In *Uncertainty for Safe Utilization of Machine Learning in Medical Imaging and Clinical Image-Based Procedures: First International Workshop, UNSURE 2019, and 8th International Workshop, CLIP 2019, Held in Conjunction with MICCAI 2019, Shenzhen, China, October 17, 2019, Proceedings 8*, pages 155–162, Shenzhen, China, Oct 2019. Springer. 10.1007/978-3-030-32689-0_16.

Li Gong, Shan Jiang, Zhiyong Yang, Guobin Zhang, and Lu Wang. Automated pulmonary nodule detection in CT images using 3D deep squeeze-and-excitation networks. *International journal of computer assisted radiology and surgery*, 14:1969–1979, 2019. 10.1007/s11548-019-01979-1.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative Adversarial Nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, Las Vegas, NV, USA, Jun 2016. 10.1109/CVPR.2016.90.

Maxwell Henderson, Samriddhi Shakya, Shashindra Pradhan, and Tristan Cook. Quanvolutional neural networks: powering image recognition with quantum circuits. *Quantum Machine Intelligence*, 2(1):1–9, 2020. 10.1007/s42484-020-00012-y.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

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, Salt Lake City, UT, USA, Jun 2018. 10.1109/CVPR.2018.00745.

I Iakovou, N Foin, A Andreou, N Viceconte, and C Di Mario. New strategies in the treatment of coronary bifurcations. *Herz*, 36(3):198–213, 2011. 10.1007/s00059-011-3459-y.

Instituto Nacional de Estadística y Geografía. Estadisticas de Defunciones Registradas 2021. (Preliminar), Sep 2022. URL https://www.inegi.org.mx/contenidos/saladeprensa/boletines/2022/dr/dr2021_07.pdf.

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, Lille, France, Jul 2015. 10.5555/3045118.

Vijayasri Iyer, Bhargava Ganti, AM Hima Vyshnavi, PK Krishnan Namboori, and Sriram Iyer. Hybrid quantum computing based early detection of skin cancer. *Journal of Interdisciplinary Mathematics*, 23(2): 347–355, 2020. 10.1080/09720502.2020.1731948.

Clara Jaquet, Laurent Najman, Hugues Talbot, Leo Grady, Michiel Schaap, Buzzy Spain, Hyun Jin Kim, Irene Vignon-Clementel, and Charles A Taylor. Generation of Patient-Specific Cardiac Cascular Networks: A Hybrid Image-Based and Synthetic Geometric Model. *IEEE Transactions on Biomedical Engineering*, 66(4):946–955, 2018. 10.1109/TBME.2018.2865667.

Gurpreet S Johal, Sunny Goel, and Annapoorna Kini. Coronary Anatomy and Angiography. In *Practical Manual of Interventional Cardiology*, pages 35–49. Springer, 2021.

Jonathan Keelan, Emma ML Chung, and James P Hague. Simulated annealing approach to vascular structure with application to the coronary arteries. *Royal Society Open Science*, 3(2):150431, 2016. 10.1098/rsos.150431.

AH Nandhu Kishore and VE Jayanthi. Automatic stenosis grading system for diagnosing coronary artery disease using coronary angiogram. *International Journal of Biomedical Engineering and Technology*, 31(3): 260–277, 2019. 10.1504/IJBET.2019.102974.

Debanjan Konar, Siddhartha Bhattacharyya, Tapan Kumar Gandhi, and Bijaya Ketan Panigrahi. A Quantum-Inspired Self-Supervised Network model for automatic segmentation of brain MR images. *Applied Soft Computing*, 93:106348, 2020. 10.1016/j.asoc.2020.106348.

Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105, Lake Tahoe, Nevada, Dec 2012. 10.5555/2999134.2999257.

Ryan LaRose and Brian Coyle. Robust data encodings for quantum classifiers. *Physical Review A*, 102(3): 032420, 2020. 10.1103/PhysRevA.102.032420.

L Latha and S Thangasamy. Efficient approach to Normalization of Multimodal Biometric Scores. *International Journal of Computer Applications*, 32(10):57–64, 2011.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 10.1109/5.726791.

Sangyoon Lee and Moon Gi Kang. Poisson-Gaussian Noise Reduction for X-Ray Images Based on Local Linear Minimum Mean Square Error Shrinkage in Nonsubsampled Contourlet Transform Domain. *IEEE Access*, 9:100637–100651, 2021. 10.1109/ACCESS.2021.3097078.

Yuchao Li, Shaohui Lin, Baochang Zhang, Jianzhuang Liu, David Doermann, Yongjian Wu, Feiyue Huang, and Rongrong Ji. Exploiting Kernel Sparsity and Entropy for Interpretable CNN Compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2800–2809, Jun 2019. https://doi.org/10.1109/CVPR.2019.00291.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in Network. *arXiv preprint arXiv:1312.4400*, 2013.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, pages 740–755, Zurich, Switzerland, Sep 2014. Springer. 10.1007/978-3-319-10602-1_48.

Bin Liu, Cheng Tan, Shuqin Li, Jinrong He, and Hongyan Wang. A Data Augmentation Method Based on Generative Adversarial Networks for Grape Leaf Disease Identification. *IEEE Access*, 8:102188–102198, 2020. 10.1109/ACCESS.2020.2998839.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single Shot MultiBox Detector. In *European Conference on Computer Vision*, pages 21–37. Springer, 2016. 10.1007/978-3-319-46448-0_2.

Xiaoxi Lu, Xingyue Wang, Jiansheng Fang, Na Zeng, Yao Xiang, Jingfeng Zhang, Jianjun Zheng, and Jiang Liu. Pulmonary Nodule Detection Based on RPN with Squeeze-and-Excitation Block. In *5th International Conference on Control and Computer Vision*, pages 85–92, Aug 2022. 10.1145/3561613.3561627.

EN Manson, V Atuwo Ampoh, E Fiagbedzi, JH Amuasi, JJ Flether, and C Schandorf. Image noise in radiography and tomography: Causes, effects and reduction techniques. *Current Trends in Clinical & Medical Imaging*, 2(5):555620, 2019. 10.19080/CTCMI.2019.03.555620.

Andrea Mari, Thomas R Bromley, Josh Izaac, Maria Schuld, and Nathan Killoran. Transfer learning in hybrid classical-quantum neural networks. *Quantum*, 4:340, 2020. 10.22331/q-2020-10-09-340.

K Aditya Mohan, Robert M Panas, and Jefferson A Cuadra. SABER: A Systems Approach to Blur Estimation and Reduction in X-Ray Imaging. *IEEE Transactions on Image Processing*, 29:7751–7764, 2020. 10.1109/TIP.2020.3006339.

Subhashree Mohapatra, Tripti Swarnkar, and Jayashankar Das. Deep convolutional neural network in medical image processing. In *Handbook of deep learning in biomedical engineering*, pages 25–60. Elsevier, 2021. 10.1016/B978-0-12-823014-5.00006-5.

Jong Hak Moon, Won Chul Cha, Myung Jin Chung, Kyu-Sung Lee, Baek Hwan Cho, Jin Ho Choi, et al. Automatic stenosis recognition from coronary angiography using convolutional neural networks. *Computer Methods and Programs in Biomedicine*, 198:105819, 2021. 10.1016/j.cmpb.2020.105819.

Kiran R Nandalur, Ben A Dwamena, Asim F Choudhri, Mohan R Nandalur, and Ruth C Carlos. Diagnostic performance of stress cardiac magnetic resonance imaging in the detection of coronary artery disease: a meta-analysis. *Journal of the American College of Cardiology*, 50(14):1343–1353, 2007. 10.1016/j.jacc.2007.06.030.

National Heart, Lung, and Blood Institute. Atherosclerosis, Oct 2021. URL https://www.nhlbi.nih.gov.

Daniel Osaku, Jancarlo Ferreira Gomes, and Alexandre Xavier Falcão. Convolutional neural network simplification with progressive retraining. *Pattern Recognition Letters*, 150:235–241, 2021. 10.1016/j.patrec.2021.06.032.

Kun Pang, Danni Ai, Huihui Fang, Jingfan Fan, Hong Song, and Jian Yang. Stenosis-DetNet: Sequence consistency-based stenosis detection for X-ray coronary angiography. *Computerized Medical Imaging and Graphics*, 89:101900, 2021. 10.1016/j.compmedimag.2021.101900.

Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1): 145–151, 1999. 10.1016/S0893-6080(98)00116-6.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, Volume 28, pages 91–99, Cambridge, MA, USA, 2015. MIT Press.

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. 10.1007/978-3-319-24574-4_28.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. *Advances in Neural Information Processing Systems*, 29:2234–2242, Dec 2016.

Salma Sameh, Mostafa Abdel Azim, and Ashraf AbdelRaouf. Narrowed coronary artery detection and classification using angiographic scans. In *2017 12th International Conference on Computer Engineering and Systems (ICCES)*, pages 73–79, Cairo, Egypt, Dec 2017. IEEE. 10.1109/ICCES.2017.8275280.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

DR Sarvamangala and Raghavendra V Kulkarni. Convolutional neural networks in medical image understanding: a survey. *Evolutionary intelligence*, pages 1–22, 2021. 10.1007/s12065-020-00540-3.

Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. In *International Conference on Artificial Neural Networks*, pages 92–101, Thessaloniki, Greece, Sep 2010. Springer. 10.1007/978-3-642-15825-4_10.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *IEEE International Conference on Computer Vision (ICCV), 2017*, pages 618–626, Venecia, Italia, Oct 2017. IEEE Computer Society. 10.1109/ICCV.2017.74.

Li Shen, Laurie R Margolies, Joseph H Rothstein, Eugene Fluder, Russell McBride, and Weiva Sieh. Deep Learning to Improve Breast Cancer Detection on Screening Mammography. *Scientific Reports*, 9(1):1–12, 2019. 10.1038/s41598-019-48995-4.

Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, pages 1–14, 2015a. URL http://arxiv.org/abs/1409.1556.

Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations*, San Diego, CA, USA, May 2015b. URL http://arxiv.org/abs/1409.1556.

Jennifer Sleeman, John Dorband, and Milton Halem. A hybrid quantum enabled RBM advantage: convolutional autoencoders for quantum image compression and generative learning. In Eric Donkor and Michael Hayduk, editors, *Quantum Information Science, Sensing, and Computation XII*, Volume 11391, pages 23–38, Online Only, CA, USA, apr 2020. SPIE. 10.1117/12.2558832.

E. M. Stoudenmire and David J. Schwab. Supervised Learning with Tensor Networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4806–4814, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819. 10.5555/3157382.3157634.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, Boston, MA, Jun 2015. 10.1109/CVPR.2015.7298594.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, Las Vegas, NV, USA, Jun 2016. 10.1109/CVPR.2016.308.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In *Thirty-first AAAI Conference on Artificial Intelligence*, 2017.

Giles Tetteh, Velizar Efremov, Nils D Forkert, Matthias Schneider, Jan Kirschke, Bruno Weber, Claus Zimmer, Marie Piraud, and Bjoern H Menze. Deepvesselnet: Vessel segmentation, centerline prediction, and bifurcation detection in 3-d angiographic volumes. *Frontiers in Neuroscience*, 14:1285, 2020. 10.3389/fnins.2020.592352.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, Volume 30, pages 1097–1105, Long Beach, CA, USA, Dec 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Tao Wan, Hongxiang Feng, Chao Tong, Deyu Li, and Zengchang Qin. Automated Identification and Grading of Coronary Artery Stenoses with X-ray Angiography. *Computer Methods and Programs in Biomedicine*, 167:13–22, 2018. 10.1016/j.cmpb.2018.10.013.

Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11531–11539, Seattle, WA, USA, Jun 2020. 10.1109/CVPR42600.2020.01155.

Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: Convolutional Block Attention Module. In *Proceedings of the European Conference on Computer Vision*, pages 3–19, Munich, Germany, Sep 2018. 10.1007/978-3-030-01234-2_1.

World Health Organization. Cardiovascular Diseases (CVDs), Oct 2021. URL https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds).

Jimmy Wu, Diondra Peck, Scott S. Hsieh, Vandana M Dialani, Constance D. Lehman, Bolei Zhou, Vasilis Syrgkanis, Lester W. Mackey, and Genevieve Patterson. Expert identification of visual primitives used by CNNs during mammogram classification. In *Medical Imaging 2018: Computer-Aided Diagnosis*, Volume 10575, pages 633–641, Houston, Texas, United States, Feb 2018. SPIE. 10.1117/12.2293890.

Wei Wu, Jingyang Zhang, Hongzhi Xie, Yu Zhao, Shuyang Zhang, and Lixu Gu. Automatic detection of coronary artery stenosis by convolutional neural network with temporal constraint. *Computers in Biology and Medicine*, 118:103657, 2020. 10.1016/j.compbiomed.2020.103657.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017. https://arxiv.org/abs/1708.07747.

Shuaijing Xu, Hao Wu, and Rongfang Bie. CXNet-m1: Anomaly Detection on Chest X-Rays With Image-Based Deep Learning. *IEEE Access*, 7:4466–4477, 2019. 10.1109/ACCESS.2018.2885997.

Samir S Yadav and Shivajirao M Jadhav. Deep convolutional neural network based medical image classification for disease diagnosis. *Journal of Big Data*, 6(1):1–18, 2019. 10.1186/s40537-019-0276-2.

Han Zhang, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena. Self-Attention Generative Adversarial Networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Volume 97 of *Proceedings of Machine Learning Research*, pages 7354–7363. PMLR, 2019.