Development of Deep Convolutional Neural Networks for the classification of the aortic valve using echocardiographic data

by

Stylianos Zafeiris

Department of Electrical and computer Engineering

Technical University of Crete

Examination Committee:

Prof. Michalis Zervakis, Supervisor

Assoc. Prof. Georgios Chalkiadakis

Assoc. Prof. Kostas Marias (Hellenic Mediterranean University)

13 October 2020

Acknowledgements

First and foremost, I would like to thank my supervisor Prof. Michalis Zervakis for his support, encouragement and guidance throughout my thesis. His valuable suggestions and deep insights helped me at various stages of the implementation.

I would also like to express my appreciation to Ms. Konstantina Moirogiorgou for supporting me with useful ideas and for providing the data used in this thesis, in collaboration with Nikolaos Anousakis-Vlachochristou, MD, Naval Hospital of Athens.

Many thanks to all my close friends and the people who are next to me for always being here for me and for encouraging me not to give up. Special thanks to all the new people I met and have created tons of memories that I will remember for the rest of my life.

Finally, I would like to express my gratitude to my family for their constant and unconditional support. The tremendous help of my parents had a catalytic role throughout all the years of my studies in order to achieve my goals. ***I would not be here if you hadn't been always there****.*

Abstract

The heart is one of the most important organs of the human body, which is responsible for the circulation of blood in it. Many times, however, various cardiovascular diseases cause problems in its functionality and need immediate treatment. These diseases are either caused by lifestyle, or exist in the form of congenital anomalies and cause problems later in the patient's life. One such abnormality is the bicuspid aortic valve, which affects approximately 1% to 2% of the world's population. It might cause various other cardiovascular diseases such as aortic valve stenosis, which can cause decreased blood flow to the aorta, which is the main artery of the human body. Hence, a fast and accurate diagnosis of the aortic valve type is important for the immediate treatment of possible diseases. The most immediate way to detect the type of aortic valve is by an echocardiogram. In some occasions, the noisy nature of ultrasound makes it difficult for doctors to diagnose.

This study aims to distinguish the aortic valve into bicuspid (abnormal) and tricuspid (normal), from echocardiographic data, in order to facilitate specialists during the examination of patients. Aortic valve classification is achieved using deep convolutional neural networks and specifically the well-known 2D network, VGG16, which is extended to 3D. Various techniques, such as data augmentation and transfer learning, are used to address the limitation of the small amount of available data. The proposed architecture achieves an accuracy of 93.82% up to 98.64%, which makes it capable of being used to assist cardiologists during the diagnosis.

Περίληψη

Η καρδιά είναι από τα βασικότερα όργανα του ανθρωπίνου σώματος, καθώς είναι υπεύθυνη για την κυκλοφορία του αίματος μέσα σε αυτό. Πολλές φόρες, όμως, διάφορες καρδιαγγειακές παθήσεις προκαλούν προβλήματα στην λειτουργεία της και χρήζουν άμεσης αντιμετώπισης. Οι παθήσεις αυτές είτε προκαλούνται από τον τρόπο ζωής, είτε υπάρχουν υπό την μορφή ανωμαλιών εκ γενετής και προκαλούν προβλήματα αργότερα στη ζωή του. Μία τέτοια ανωμαλία είναι η δίπτυχη αορτική βαλβίδα την οποία εμφανίζει περίπου το 1% με 2% του παγκόσμιου πληθυσμού. Αυτή δύναται να προκαλέσει διάφορες άλλες καρδιαγγειακές παθήσεις όπως, για παράδειγμα στένωση της αορτικής βαλβίδας η οποία μπορεί να προκαλέσει μείωση της ροής του αίματος προς την κυριότερη αρτηρία του ανθρωπίνου σώματος, την αορτή. Γίνεται αντιληπτό ότι είναι σημαντική η σωστή διάγνωση του τύπου της αορτικής βαλβίδας για την άμεση αντιμετώπιση πιθανών νοσημάτων. Ο πιο άμεσος τρόπος για την ανίχνευση του είδους της αορτικής βαλβίδας, είναι το υπερηχογράφημα καρδιάς. Συχνά, όμως, η θορυβώδες φύση του υπερηχογραφήματος δυσκολεύει την διάγνωση από τους γιατρούς.

Στην μελέτη αυτή γίνεται προσπάθεια για την διάκριση της αορτικής βαλβίδας σε δίπτυχη (μη-φυσιολογική) και τρίπτυχη (φυσιολογική), από δεδομένα υπερήχου καρδιάς, με σκοπό την διευκόλυνση των ειδικών κατά την διάρκεια της εξέτασης των ασθενών. Η διάκριση της αορτικής βαλβίδας επιτυγχάνεται με χρήση συνελικτικών νευρωνικών δικτύων και πιο συγκεκριμένα μέσω του γνωστού 2D δικτύου, VGG16, το οποίο επεκτείνεται σε 3D. Διάφορες τεχνικές επαύξησης δεδομένων και μεταφοράς γνώσης αντιμετωπίζουν το περιορισμό που εισάγει ο μικρός αριθμός των διαθέσιμων δεδομένων. Η προτεινόμενη αρχιτεκτονική επιτυγχάνει ακρίβεια από 93.82% έως και 98.64%, γεγονός που την καθιστά ικανή να χρησιμοποιηθεί για την υποβοήθηση της διάγνωσης από τους ειδικούς.

Contents

List of Tables

List of Figures

To my parents…

Chapter 1 Introduction

Considering that the heart is a vital organ of the human body there is a lot of research focused on diagnosing Cardiovascular Diseases (CVD) with the use of various Artificial Intelligence (AI) algorithms. Machine Learning (ML), a subset of AI, is the most utilized tool for the detection, segmentation, and classification of CVD. It not only facilitates the tasks performed by medical specialists via providing targeted, real-time indications during the examination of the patient, but also it can achieve high performance in such processes. In order to develop an efficient neural network (NN), an enormous amount of data is needed for both training and testing. For heart-related issues, data can be acquired from electro-cardiograms (ECG), magnetic resonance imaging (MRI), heart computed tomography (CT) scans and echocardiograms. The echocardiogram is a simple, non-invasive, inexpensive method, with a short period of acquisition of the results; thus, it is primarily used in detecting various CVD.

1.1 Related Work

There are a lot of studies in literature that address the classification of the Aortic Valve (AV) from a medical perspective [1]–[3], using statistical analysis on manually extracted features from echocardiograms, such as the diameter of the aorta and the number of raphes. Furthermore, Sadron et al. [4] discuss the benefits of 3D transthoracic echocardiograms (TTE) in children for determining the configuration of the AV, comparing 2D and 3D techniques.

Numerous studies deployed deep learning (DL) algorithms to deal with the problem of CVD detection. Most of them are concentrated on identifying the echocardiographic view [5], [6] and [7]. Howard et al. [7] proved that recent state-of-the-art networks can halve the classification error of the different standard views. Nizar et al. [8] focused on detecting the valve in an image or video accentuating the role of inference speed in real-time applications. Gong et al. [9] implements a novel deep convolutional Generative Adversarial Network (GAN) for Fetal congenital Heart Disease (FHD) recognization.

Additionally, many studies explored the significance of using ML in cardiovascular imaging and especially in echocardiography. Seetharam et al. [10] pointed out limitations of ML and Liu et al. [11] provided a detailed review of DL architectures and the tasks that can be accomplished on ultrasound (US) data.

Various medical tasks can be automated with ML algorithms. Zhang et al. [12] proposed a representative example of fully automated procedures. This work proposed a pipeline for the analysis of echocardiograms which provides echocardiographic view classification, disease detection such as Hypertrophic Cardiomyopathy (HCM), Pulmonary Arterial Hypertension (PAH), and cardiac Amyloidosis and finally information regarding the structure and function of the heart. It is common sense that this automation cannot replace human specialists but only help them with their diagnosis.

1.2 Motivation

Patients with bicuspid aortic valve (BAV) might develop CVDs and basically aortic valve stenosis, which is a life-threatening disease, as the valve may contract and the blood flow begins to decrease. As seen in the literature above, there is not any study -in our knowledge- that tries to identify whether the patient has a bicuspid aortic valve or a normal tricuspid one using deep learning techniques on video or image data. Thus, the primary objective of this study is to customize the well-known VGG16 network architecture, developed by Karen Simonyan & Andrew Zisserman [13], in order to efficiently classify the shape of the aortic valve from echocardiograms. The main challenge is that the echocardiograms, sometimes, are difficult to read and the sonographers might struggle to correctly classify the aortic valve configuration. A major limitation is the small dataset size, which is outfaced with transfer learning and case-realistic augmentation.

1.3 Thesis outline

The study, additionally, includes an illustrative description of the theoretical background, that is needed, to grasp the problem, the developed approach and the corresponding results. Specifically, in **chapter 2** the medical and technical background are presented, providing information about the BAV, Convolutional Neural Networks' (CNN) various operations and how they learn features from images. In **chapter 3** there is the full description of the dataset as well as the methodology used to create, train and test the 3D architecture and the alternative classification techniques used instead of fully connected layers. The results of the study are summarized in **chapter 4** and finally, in **chapter 5** there is a discussion about the outcome of the study.

Chapter 2 Theoretical Background

2.1 Medical overview

2.1.1 Anatomy and functionality of the heart

Heart is a concave muscle and one of the most important parts of the human body, since it is responsible for circulating the blood in it. This muscle, that has the size of a human fist, pumps blood from tissues through veins, then filters the carbon dioxide in lungs and other substances in kidneys and then it pushes oxygen-rich blood through arteries back to the tissues [14]. In figure 1, the shape and various parts of the heart are presented. Heart consists of four cavities and four valves which let the blood flow in only one direction. Those cavities include two atria and two ventricles, divided into the left and right part of the muscle. The left atrium is connected to the left ventricle through the mitral valve and the right atrium is connected to the right through the tricuspid valve. The right atrium receives blood from all parts of the body through the veins, promotes it to the right ventricle and from there to the pulmonary circulation for oxygenation. Then the blood is pushed from the lungs to the left atrium and from there to the left ventricle. With the muscle's contraction oxygenated blood is transferred to the whole body through the aorta and large arteries.

Unfortunately, the function of the heart is altered either due to extrinsic factors such lifestyle, either from congenital anomalies that cause various problems later in the life of the patient. Common cardiovascular diseases are:

- Heart attacks
- Heart valve diseases
- Vascular disease
- Abnormal heart rhythms
- Aortic stenosis

Those diseases are, sometimes, treatable and the restoration of the normal functionality of the heart is possible.

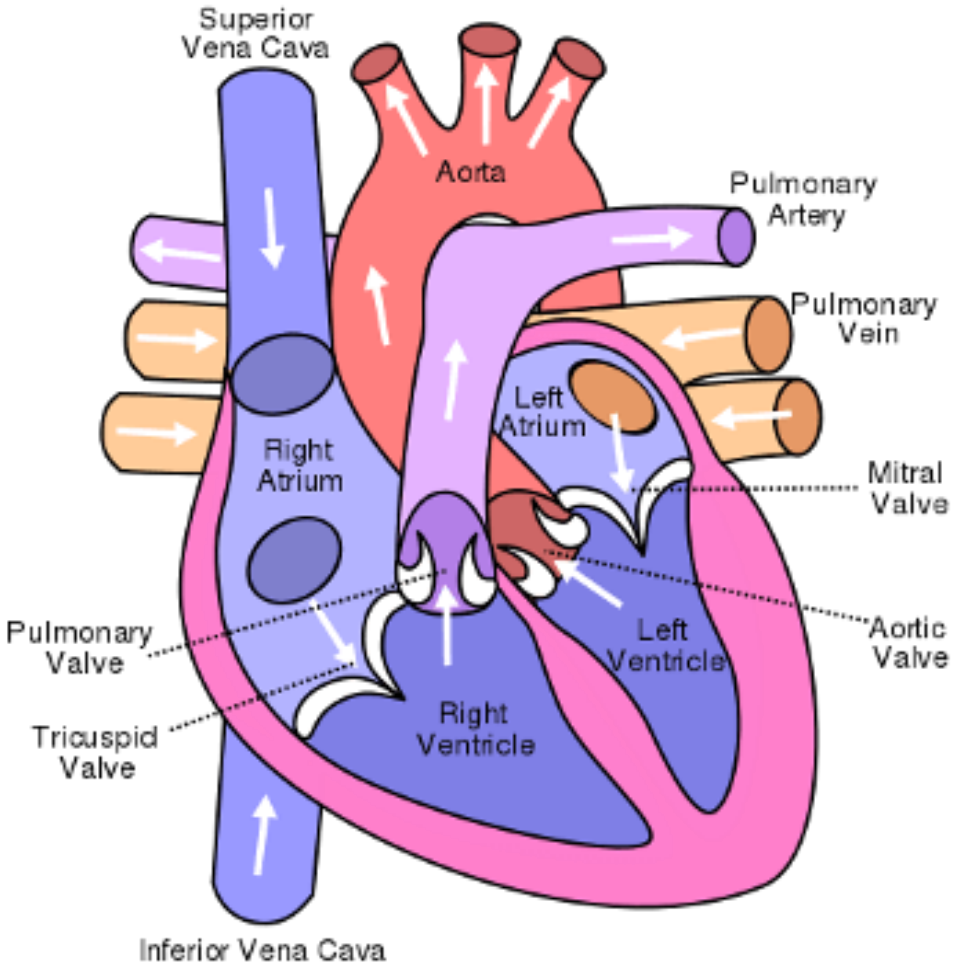

Figure 1 Anatomy of the heart. (https://pacificmedicalacls.com/images/Image-1-Diagram-of-the-human-heart.png)

2.1.2 The aortic valve

Aorta is the main artery of the human body, since the oxygen-rich blood is funneled through it to the rest of the body. It is connected with the left ventricle via the aortic valve, which prevents backward blood flow from the aorta to the ventricle. Aortic valve has three leaflets which seal the valve during the closed state and let the blood flow in the open state. However, some people have an altered shape of valve which has a missing leaflet. In this congenital and abnormal shape the valve is called bicuspid, while the normal configuration is known as tricuspid aortic valve.

The different configurations of the aortic valve are summarized in the next figure:

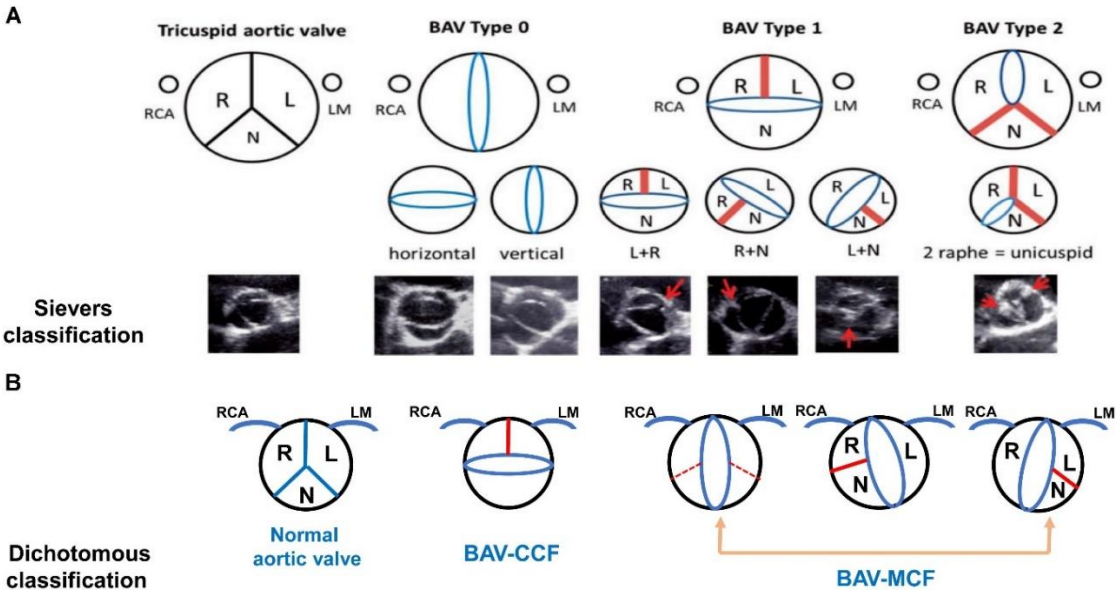

Figure 2 Configurations of the aortic valve. [2]

Bicuspid aortic valve may cause a reduction of the blood flow to the aorta, hence cause aortic stenosis. This underlines the need for early diagnosis and treatment. Finally, in order to better understand the shape of aortic valve the following figures represent the two usual configurations:

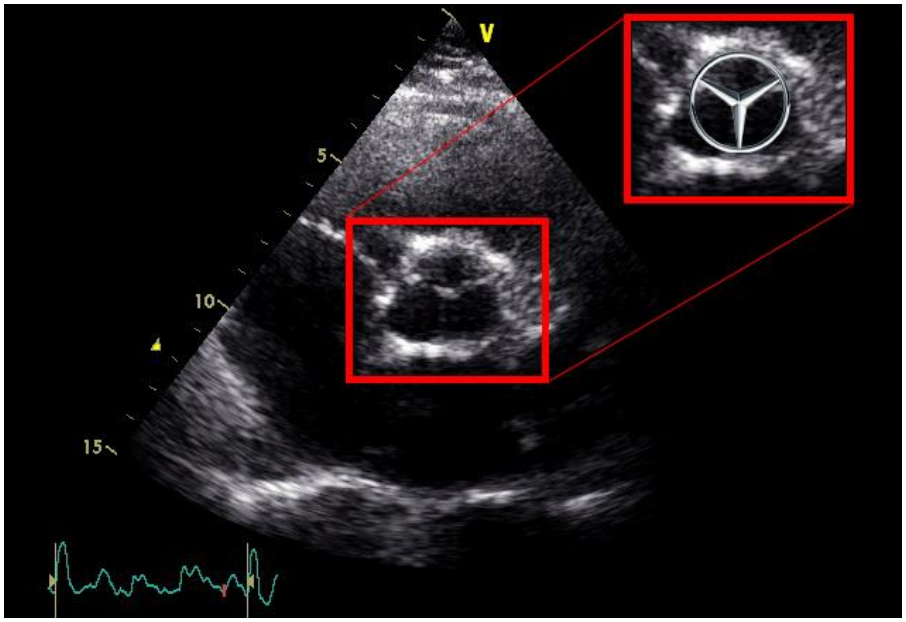

Figure 3 Tricuspid valve as shown in an echocardiogram. Usually it is interpreted as inverted Mercedes sign.

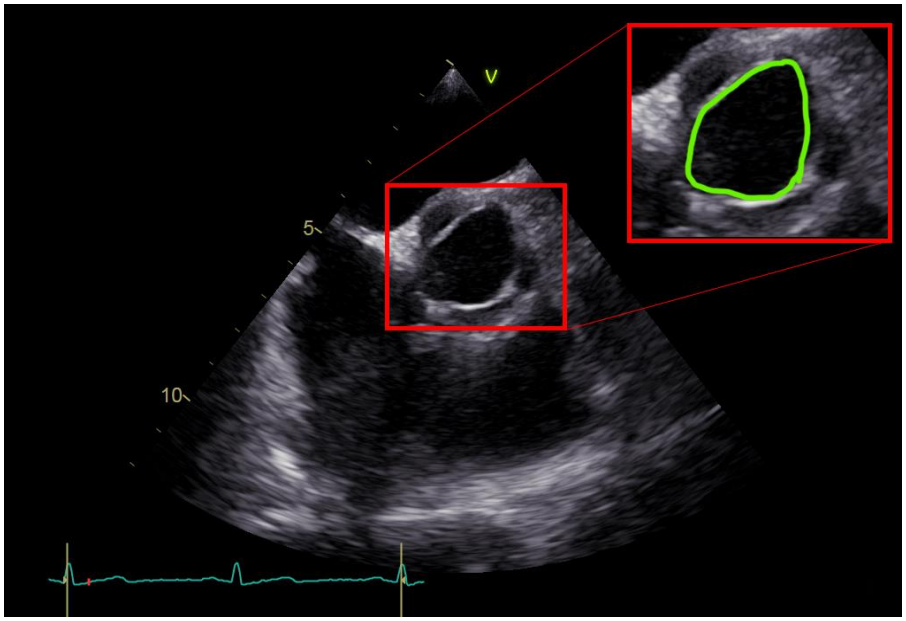

Figure 4 Bicuspid valve as shown in an echocardiogram. Usually it is interpreted as an open fish mouth.

There are a lot of medical tests used to diagnose a bicuspid valve, such as MRI and CT scans, but ultrasound tests are more accessible, since they visualize in real-time the heart's structure. On top of that echocardiograms are cheaper than the other methods and most important it does not emit ionizing radiation upon the patient. Thus, cardiologists use echocardiograms as the primary noninvasive examination method.

2.2 Artificial Neural Networks

In this era of information there is a need for creating more complex algorithms, in order to solve complicated problems. In 1943 Warren McCulloch created the first algorithm that can learn, named artificial neural networks (ANN). This attempt was inspired from the structure and functionality of biological neurons. An example of the simplistic architecture of ANNs is shown in figure 5. The common pipeline for solving a problem utilizing a neural network consists of four stages. First stage is data acquisition, in which all the data that is going to be used to train and test the network should be gathered and preprocessed, in order to become trainable. Second stage is the training of the network, where the weights of the network constantly change in order to learn. Next stage is the evaluation of the network with metrics calculated upon the

test data. Lastly, the trained network is deployed in order to solve the problem it was trained for.

Due to the spread of machine learning in various scientific fields, the simple architectures of artificial neural networks became insufficient for problem solving. New types of neural networks where developed to cope with the challenges that were introduced by the community that embraced machine learning algorithms. Well known examples are Recurrent Neural Networks (RNN) which can model time series efficiently and Convolutional Neural Networks (CNN) that can be trained to perform various operations on images.

The amount of applications that required image processing via this novel problem solving technique increased, because more valuable information could be extracted from images. Common need were object detection, image segmentation and classification based on the context of the image. The main drawback of ANNs is that they cannot handle efficiently the enormous input size of an image and hence a lot of computational resources are needed for training. Contrastingly, CNNs need both less resources and time to train due to their structure and functionality. They have filters that convolve with the image and then the output is propagated to the next layers.

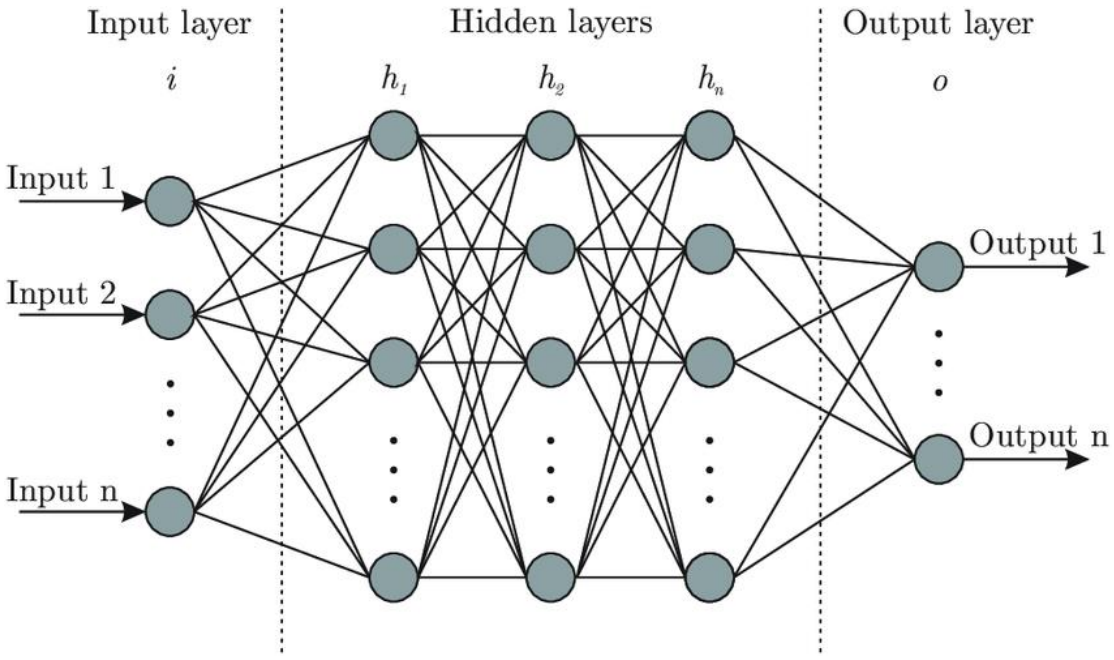

Figure 5 Typical structure of a simple artificial neural network. (https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o_fig1_321259051)

A typical structure of a CNN is shown in the next figure:

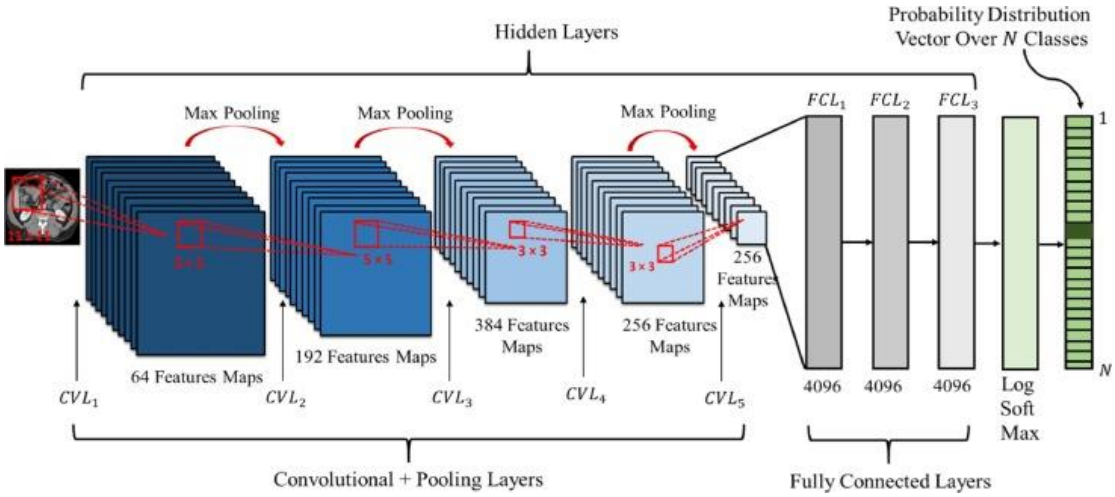

Figure 6 Typical structure of a convolutional neural network. (https://www.sciencedirect.com/science/article/abs/pii/S0925231217308445)

The architecture of a CNN enables it to recognize the images' context and analyze the correlation of the input along both x and y axes. This type of networks offered promising results in all image manipulation tasks, but soon their weakness appeared. They could not learn enough features in order to perform well in all kinds of applications. Shallow CNN architectures are not adequate for classification tasks with complex input, since the amount of features they can learn is limited from the architecture itself. This complexity, usually, is found in medical images where the input is complex and contains a lot of useful information. Specialists need more reliable Computer Aided Diagnosis (CAD) tools, in order to diagnose fatal diseases earlier and with greater precision. Thus, a new type of neural network appeared and was named Deep Learning, due to its multilayer architectures.

2.3 Deep learning

Deep neural network have stacked multiple layers, so they can extract more features than shallower architectures. This enables them to be trained on more complex datasets, achieving great performance. In deep neural networks, as in in CNNs, the procedure of feature selection is automated and optimal, since convolutional layers' weights are tuned in each epoch, with respect to the input data. On the contrary optimality of extracted features is in doubt, because there is no concrete mathematical foundation that is able to prove it. Hence, the deep

network's optimality in constrained for a custom application. Furthermore, deep neural networks can model more abstract notions on input, due to the multilayer architecture. Each layer learn unique features. In the first layers simple features like edges are extracted. In the next layers, those edges are combined and form shapes, therefore while going deeper features become more advanced such shapes and textures. In the next few pages we will present the fundamental principles of deep neural networks.

2.3.1 Convolution

Convolution is the most fundamental principle of deep learning. Convolutional layers contain small fixed sized filters that convolve with the image and extract features. A large image contains petite regions with valuable information which is extracted via convolution. While the kernel moves towards the 2 directions of the image small convolutional kernels trace pixel wise point-to-point influence that the human eye cannot see. The mathematical formulation of convolution is:

$$y\left[i,j\right] = \sum_{n=0}^{N} \sum_{m=0}^{M} h\left[n,m\right] \cdot x\left[i-n, j-m\right]$$

where x is the N-by-M input image, h the filter used and y the resulted image. The dimensions of the resulted image are calculated using the following equations:

$$out_W = \frac{W - F_W + 2 \cdot P}{S_W} + 1$$

$$out_H = \frac{H - F_H + 2 \cdot P}{S_H} + 1$$

where out_W is the output width and out_H the output height. (WxH) is the size of the input image and (F_WxF_H) is the convolutional kernel's size. S_W and S_H is the stride used in each dimension and P is the amount of extra dimensions with 0's used for padding outside the border of the image. The term stride refers the amount of pixels the kernel moves in each direction and the term padding refers to the extra zeros placed around the border of the image, so that the

output convolution has the same dimensions as the input image. If the resulted image do not have the same dimensions then it is called valid padding. To determine the extra dimensions needed for the padding of the input, in order to have the same size as the output we use the following formula:

$$P = \frac{K-1}{2}$$

where K is the kernel's size. An example of valid padding is displayed in the following figure:

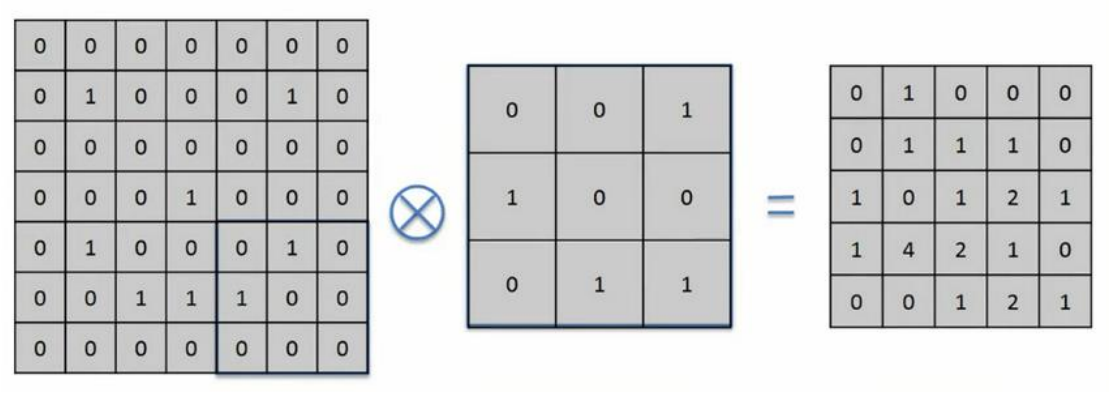

Figure 7 Valid padding. (https://ieeexplore.ieee.org/document/8596839)

It is understood that no extra zeros were placed near the border of the input image, hence the output has smaller dimensions. Next an example of same padding is presented:

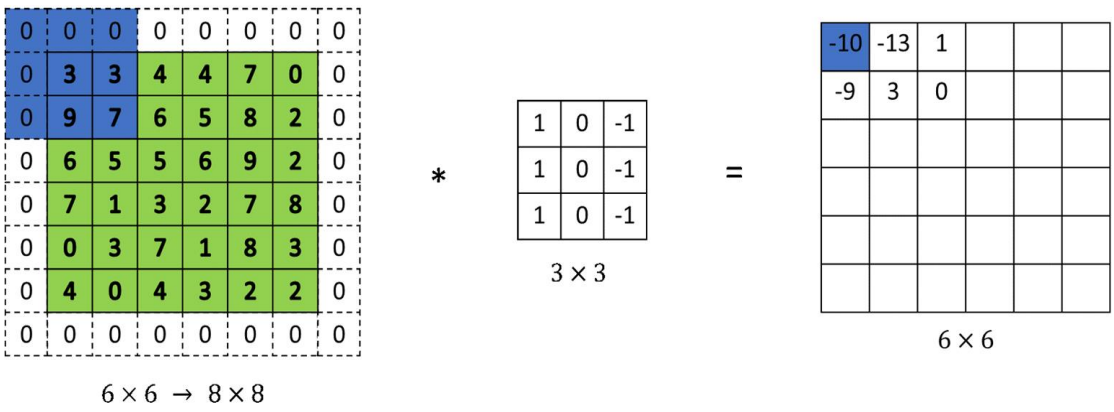

Figure 8 Same padding. (https://medium.com/analytics-vidhya/understanding-cnns-68da06af1dfb)

In the second example there were placed two extra columns and two rows, filled with zeros near the border of the image, so the output image has the same size. This description of convolution applies for 2D data samples. However, convolution can be extended to 3D. Instead of using 2D kernels that move towards the 2 dimensions of the image, 3D filters are applied, which move in 3 dimensions of the 3D input. In this manner the layers learn spatio-temporal features and specifically the correlations between the 3 axes, where the third axis is time.

2.3.2 Activation functions

Activation functions control whether a neuron will be activated by the current input or not. The output of a neuron must be equal to a value greater than the activation threshold so as the neuron can be activated and propagate its value to the next neuron. A common example of activation function is the binary step activation, where the neuron is activated only if the input value is greater than the selected threshold.

$$BinaryStep = 1, output > threshold$$

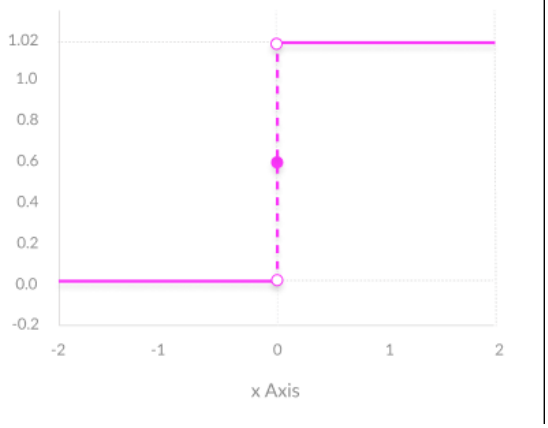

Figure 9 Binary step activation function.[1]

Besides this step function there is the linear activation, where the output of a neuron is a linear function of the provided input as it is shown in figure 10. In

[1] All figures for activation functions where from: https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/

general linear functions cannot learn complex features. First, they do not allow backpropagation, since there is a constant derivative and has no correlation with the input data. As a result the network cannot learn which weight causes better predictions. Secondly, there is no point to stack multiple layers together since one is a linear combination of its predecessors, since this transforms a network into a regression model. Nevertheless, linear activation can be used for more than two classes, in contrast with the binary step.

$$Out = c \cdot input$$

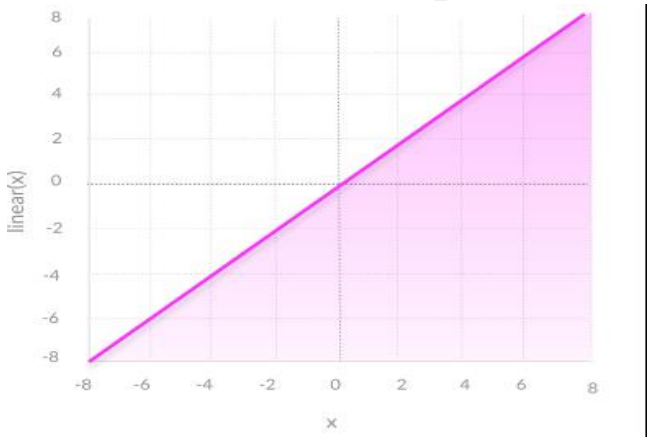

Figure 10 Linear activation function.

Conversely, non-linear activation functions allow the network to create more complex mappings between the network's endpoints and can handle the major drawbacks of the linear as well. Their derivatives are related to the input, so they allow backpropagation. More complex features can be learnt, because multiple layers can be stacked together. The most common activation functions are presented below:

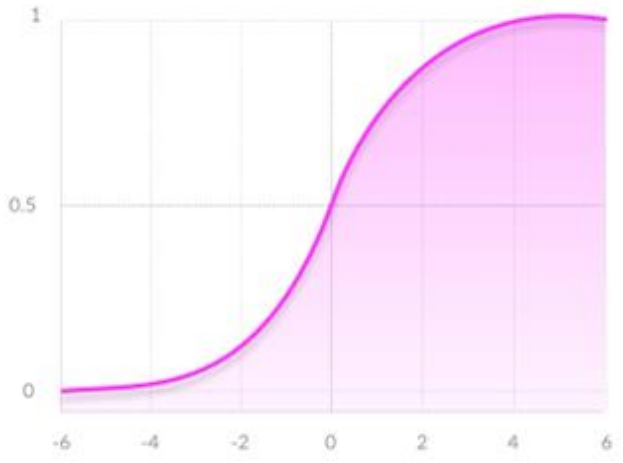

Figure 11 Sigmoid (logistic) activation function.

The sigmoid function, shown in figure 11, has output values between 0 and 1. The main drawback of this function is that it causes the elimination of the back-propagated gradient, also known as the Vanishing Gradient Problem. In order to minimize the vanishing of the gradient, relu (rectified linear unit) was introduced. It has a computationally feasible cost, allowing fast convergence and as well it allows the backwards propagation of the derivatives. The challenge of using this function is the Dying Relu Problem, where derivatives approximate zero when there is negative or close to zero values and decelerates backpropagation.

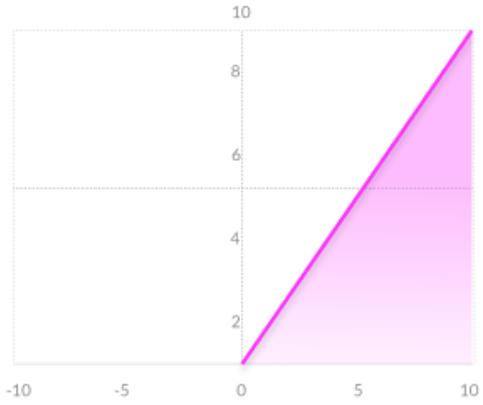

Figure 12 Relu activation function.

Finally, to deal with the dying relu problem, leaky relu was introduced. It restraints Dying ReLU Problem, but output may not be consistent for negative values of the input.

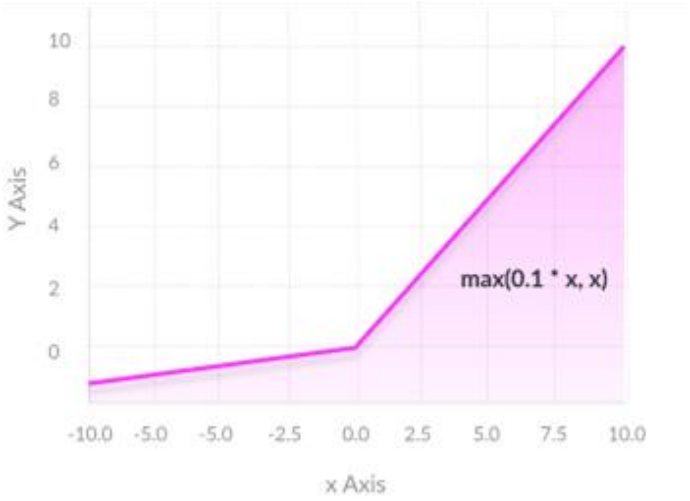

Figure 13 Leaky relu activation function.

Selection of activation function must take into consideration the number of classes, since not all functions are suitable for multiclass problems, the values generated and finally whether they should propagate the output of the neuron to the next or make a prediction, like the softmax activation function does and hence position them properly.

2.3.3 Pooling layers

Another important part of a deep learning network is the dimensionality reduction of the data propagated to the next layer. This down sampling happens in pooling layers (e.g. max, min, sum, etc.) and enables the network to learn advanced features (shapes, textures, details) on deeper layers, since the influence of the bigger image parts starts eroding. These layers result to optimal feature selection by the network. The pooling operations consists of selecting a value from a pooling layer's receptive field and transfer it to the next layer, while all the other values are skipped. The selected value can be the largest in this small region, so it is called max pooling and is the most commonly used. Respectively, it is called min pooling when the smallest value is selected. In addition, there are some cases that sum or average pooling can be used. This means that the result is either the sum of the values of the receptive field, either their average.

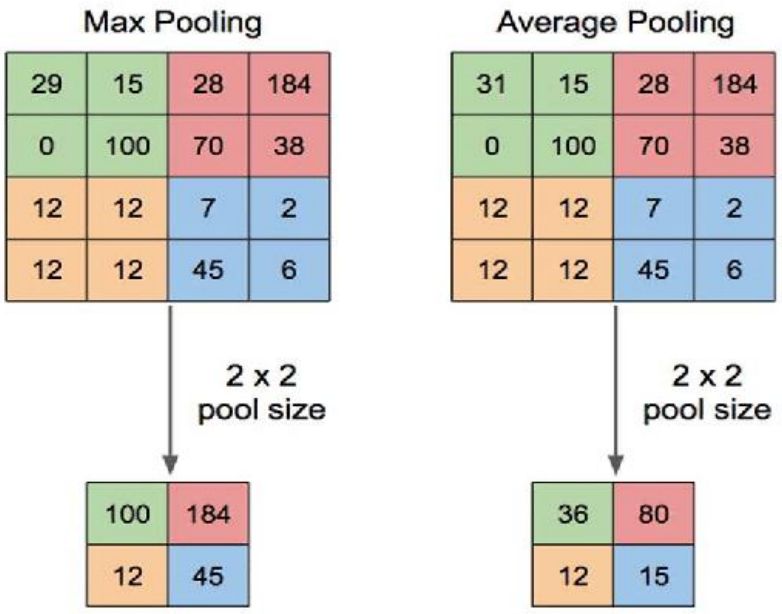

Figure 14 Max and average pooling operations.
(https://www.cs.cmu.edu/~16311/current/schedule/ppp/CNNs.pdf)

2.3.4 Basic network architecture and operation

Every deep learning network layer contains at least one convolutional layer followed by an activation function and then a max pooling layer. Feature extractions occurs in the first layer and feature selection in the last. This layer sequence is repeated several times so it forms a deep, multilayer architecture. Finally, there is a flatten layer which creates an 1D vector followed by dense layers which contain nodes that are connected with all the nodes from the previous and next dense layers, but not between the same layer. This part of the network is called Fully Connected layer and forms the classifier that is trained for deciding the class that the provided sample belongs to. The figure below shows the common order of layers in deep learning architecture:

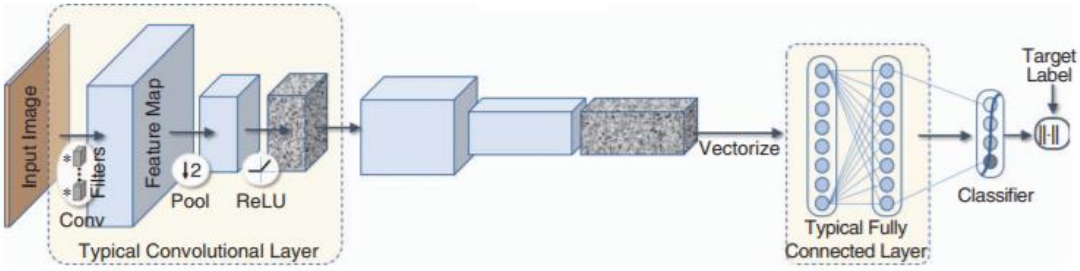

Figure 15 Common layer order in deep learning architectures. (https://www.researchgate.net/figure/A-basic-CNN-architecture-with-a-convolution-pooling-activation-along-with-a-fully_fig3_323694671)

In order to operate a neural network, there are some stages that must be completed. The first stage is the forward pass where the network uses the existent weights and process the input in each layer. While passing the input forward, the network generates an output that is the prediction for the given input. The output of the forward pass may have great deviation from the real output. This is the reason why backpropagation of the error is occurring. When the output of the forward pass is generated, a loss function is used to calculate the model error between the prediction and the ground truth. The whole training process is based on optimizing the selected loss function. Then the gradient of the network's output loss with respect to all weights is calculated and used in order to recalculate the network's weights.

After calculating the gradients, the backpropagation stage begins, in which the calculated gradients are propagated from the last to the first layer of the network. The propagated deviations are then multiplied by a learning rate and

used in recalculating the weights for each layer. A simple interpretation of how the backpropagation happens in a neuron is presented in the following figure:

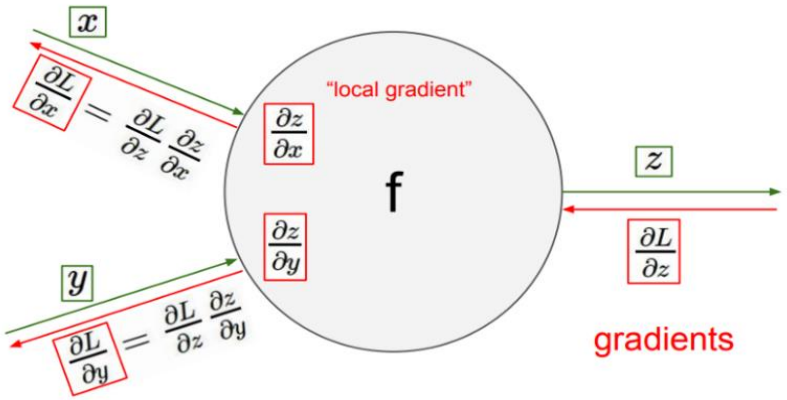

Figure 16 Gradient backpropagation. (https://slideplayer.com/slide/14518448/ - slide 36)

A more detailed interpretation of the training procedure is provided by the figure below:

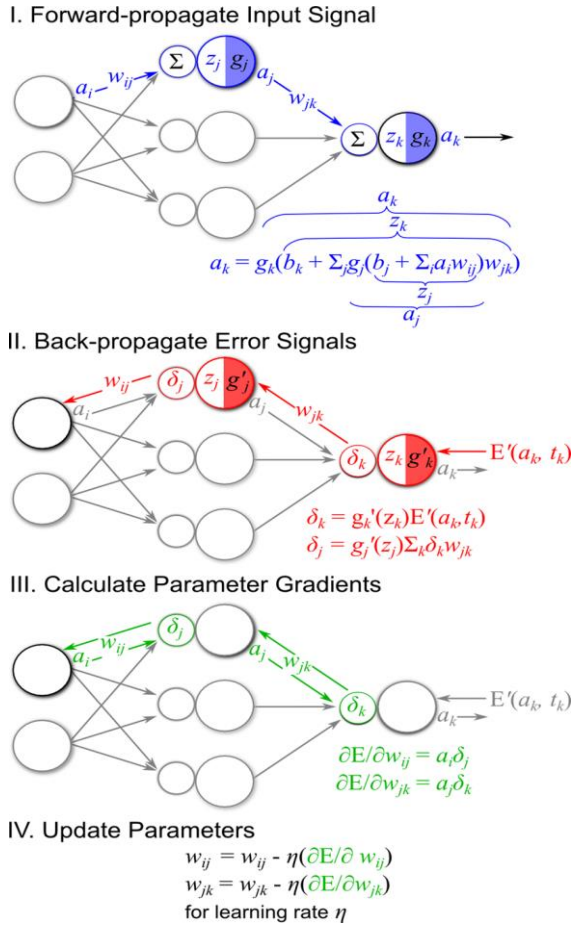

Figure 17 Detailed interpretation of training process in neural networks. (https://en.proft.me/2016/06/15/getting-started-deep-learning-r/)

At this point we should underline that training a neural network is a computationally heavy and time-consuming task as it requires a lot of resources, to perform the numerous calculations that are needed. The algorithm used during training has an important role as well as the selection of different network hyper parameters. Despite the heavy computational load deep learning has become a trend and is widely used, since it has high performance in classification and segmentation tasks of general images as well as more complicated input as medical images and videos.

2.3.5 Performance evaluation of a network

Another important aspect for the development of a successful model with high accuracy is evaluating it. Testing the network with unseen data must be performed in order to calculate its real performance. The most common evaluation method is calculating the model's accuracy, by dividing the amount of all the correct prediction by the total amount of predictions made by the network, using the same test set. Other common metrics, which were used in this thesis, are:

- $Sensitivity = Recall = \frac{TP}{TP+FN}$
- $Specificity = \frac{TN}{TN+FP}$
- $Precision = \frac{TP}{TP+FP}$
- $F_1\ score = \frac{2xTP}{2xTP+FP+FN}$, the harmonic mean of precision and sensitivity.

Finally, the AUROC (Area Under the Receiver Operating Characteristics) curve was used in order to further evaluate the model's performance. The ROC curve is a graph that shows the model's performance at various classification thresholds, by plotting the true positive rate, which is defined as the recall and the false positive rate that equals $\frac{FP}{FP+TN}$. AUC (Area Under Curve) measures the area under the ROC curve, providing a complete measurement across all thresholds used for classification. Last, is the confusion matrix, also known as the error matrix. In the y-axis it displays the predicted labels and in the x-axis the actual labels, while each cell contains the amount of samples of the predicted class that where classified as the corresponding actual class.

Chapter 3 Methodology

3.1 Available dataset

The selected dataset has a crucial role in every deep learning study. The quality of the data is contingent on the technical specifications of the machine used to capture them, the nature of the tests and the experience of the specialist performing them. Although, high quality helps the training procedure of the network, it is not always sufficient [15] for achieving high performance. Three necessary conditions that must be satisfied, on top of data quality, are the dataset size, the deep learning network architecture and the tuning of model's hyper parameters. Large train dataset will help preventing the network from overfitting and increase generalization. Furthermore, testing the model on a larger test set increases the precision of any performance metric calculated on it. Furthermore, the network's architecture (filter and layer size and quantity, type of each layer and its location) and the selection of the parameters (learning rate, optimization technique, etc.) can influence the learning procedure as well the amount of features learnt by it. More filters in a convolutional neural network means that more features can be learnt, but it requires more training data. This shows the importance of the dataset while deploying a deep learning model. This also applies in medical applications where extreme caution is needed for accurately diagnosing diseases, such as in the current study.

In this thesis, the provided dataset consists of a total of 67 echocardiograms and 100 images from open and closed state showing both normal and abnormal aortic valves. Provided files have three color channels Red, Green and Blue (RGB). Those echocardiograms were captured from patients at the Naval Hospital of Athens by an experienced sonographer and were provided confidentially to us for the purposes of the study. No personal or health-related information were available to preserve the anonymity of the patients. The received echocardiograms and images were captured from the Parasternal Short Axis (PSAX) view, for at least one cardiac cycle, containing the electrocardiogram (ECG) waveform and various ultrasound indicators which are not used in the present study. The GE Versana Active with 3Sc-RS probe was used to capture the echocardiograms. The aortic valve types that were interpreted in both the images and the videos are Tricuspid, Bicuspid and

Raphe. As mentioned in section 2.1.2, tricuspid is the normal and both bicuspid (type 0) and raphe (types 1 and 2) from figure 2 are the abnormal configurations of the aortic valve. The number of available videos for each type are shown in Table 1.

	Number of cases	Type
Normal aortic valve	30	Tricuspid
Abnormal aortic valve	9	Bicuspid
	28	Raphe

Table 1 Amount of provided videos.

Similarly, the number of available images are depicted in Table 2.

	Number of cases	Total	Type
Normal aortic valve	60 open/closed state	120	Tricuspid
Abnormal aortic valve	11 open/closed state	22	Bicuspid
	29 open/closed state	58	Raphe

Table 2 Amount of provided images.

The amount of available data is relatively small; however, in such applications, larger dataset is important in order to evaluate model's performance and increase precision of metrics. Hence, a case-realistic augmentation schema is

developed and described in section 3.3. Finally, the class imbalance does not seem to be significant for the two class classification problem (normal vs abnormal), since there are 30 normal versus 37 abnormal videos and 120 normal versus 80 abnormal images. In contrast, there is a significant class imbalance between the three class cases.

3.2 Data preprocessing

A vital step, before training a neural network, is preprocessing the data. This should happen for two main reasons. First of all, the data must fulfill the network's constrains, like the input size in a convolutional layer, else the training procedure cannot begin. Secondly, the data dimensions might be exhausting for the available computational resources. Thus, data preprocessing preserves the feasibility of the development and implementation of the network. In this study, we used some common preprocessing methods such as cropping, resizing and converting to grayscale in order to reduce the size of the data and comply with the network's architecture input dimensions.

It is worth mentioning that not all videos had the same amount of frames, varying from 15 to 482 frames. There were three videos (1 from bicuspid type and 2 from tricuspid) with only a single frame; thus, they were excluded from the analysis. From the single (raphe) case with 482 frames we managed to produce one extra video, splitting the initial video into two parts by looking the ECG waveform for capturing a cardiac cycle. In each video, every frame was striped and stored -in the correct order- in a folder named after the video's title. Then, we thoroughly selected only 40 frames from each video to cover one cardiac cycle in all available cases. The selection did not happen randomly, conversely it was based on the electrocardiogram provided within the video. Consequently, all videos with less than 40 frames were interpolated to increase the frame rate and acquire the extra frames that were needed. Lastly, each preprocessing method was applied directly to each consecutive frame of every video, as well as each image.

The first preprocessing method of the images was the RGB to grayscale conversion where the ITU-R BT.601-2 [16] luma[2] transform was used, described by the following equation:

$$Luminance = \frac{R \times 299 + G \times 587 + B \times 114}{1000}$$

In this way, the initial three channels (RGB) were replaced by the luminance channel, since color in the given images does not contain any resourceful information. The conversion to grayscale was not applied in provided images, because they were used to train the VGG16 2D architecture, which accepts the three color channels, as well.

Cropping was the second method applied on the data, because it can drastically remove all the unwanted indicators from video frames and images. For the bicuspid and raphe cases, center-cropping was applied, while the tricuspid cases needed a custom square cropping area starting in the (68, 5) point and ending in the (570, 386) point of both frames and images. This happened because data for the abnormal class had centered the echocardiograms in both images and frames, while echocardiograms were not centered in the data from the normal cases. Finally, all the video frames were resized to a 256x256 fixed size and all images' size was set to 224x224 so they can be used to train the 2D VGG16.

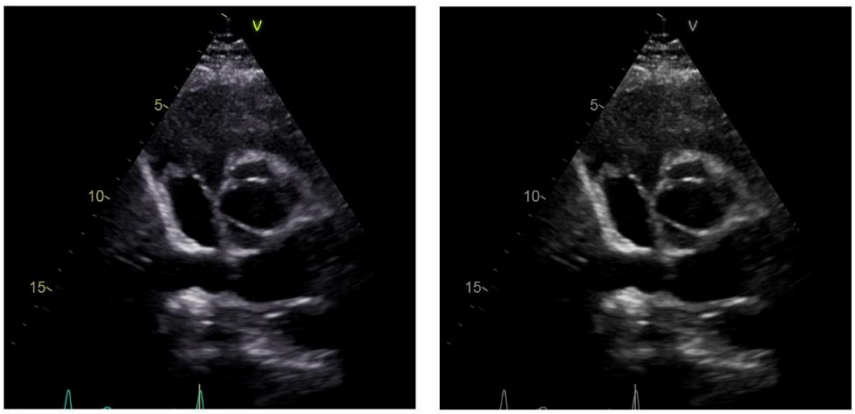

Figure 18 (left) RGB image, (right) grayscale using ITU-R BT.601-2 luma transform.

[2] Luminance transformation for compressed images.

Figure 19 Picture above is the original raphe frame, while the picture below is the same image but cropped.

3.2.1 Video interpolation techniques

As mentioned previously, during the preprocessing of videos there was the need to interpolate those with less than 40 frames. Interpolation is a technique that generates new data from existing frames in order to extend the frame rate of the video. This technique can be applied using different methods and tools. In our approach we used linear and motion interpolation, on 15 bicuspid videos out of 28. For linear interpolation the "FFmpeg" [17] library was used, providing mediocre results, since the output videos were blurry, making them almost impossible to investigate. Similarly, for motion interpolation the tool "butterflow" [18] was used, which implements the methodology introduced by G. Farneb*ä*ck [19], which is based on polynomial expansion. After executing both methods, we compared the final results and discarded the one with the worst optical performance. The motion interpolation performed optimally; thus, it was selected as the default interpolation method. Observing the two images bellow, our choice is justified.

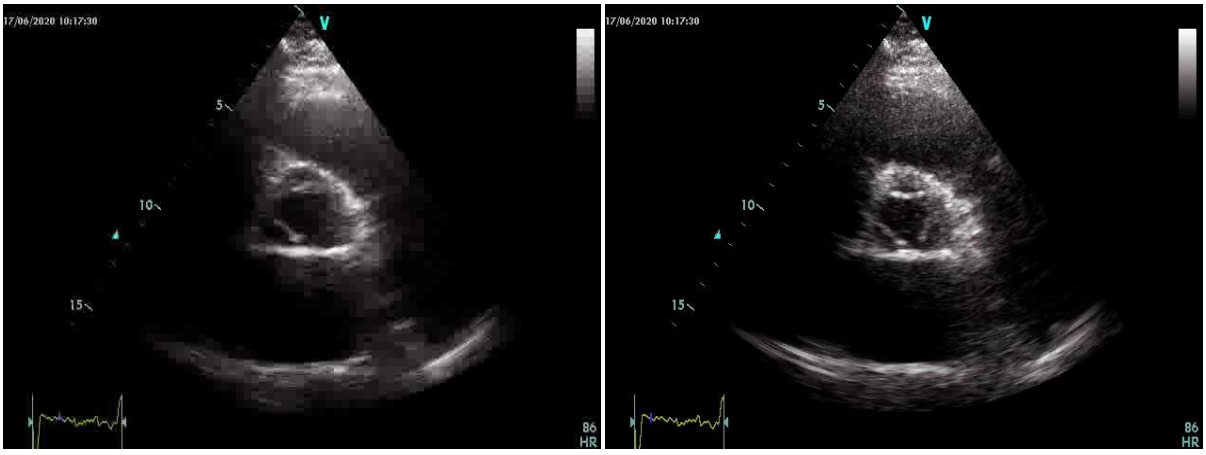

Figure 20 (left image) Linear interpolated frame, (right image) motion interpolated frame. Above frames are not from the same video file, but it is clear which method performed better, since the left image is blurry and illegible, while the right one is more clean.

3.3 Data augmentation

After the preprocessing step, dataset consists of 8 bicuspid cases, 28 tricuspid cases, and 29 raphe cases each one contains 40 carefully selected grayscale frames from the corresponding videos. The few samples designates that there

is a need for an increase in dataset size. To deal with the small dataset size, we implemented five augmentation techniques to not only increase the provided dataset size, but also to create a more robust model for the classification of the aortic valve. The proposed techniques presented below, try to simulate case-realistic distortions that may occur when capturing videos or images on ultrasound data.

3.3.1 Additive noise

Noise is the most common distortion in every signal. Ultrasound images are not always clear and the specialists may have difficulties deciding in which class the configuration of the aortic valve belongs during the test, depending on the level of their experience. Hence, additive noise tries to simulate this confusing situation on top of the already naturally noisy echocardiograms. Moreover, some videos were clear, so we added the noise to create noisy copies of them, in order to create a more robust model. To apply this technique, we create a new image of the same size as the original frame, filled with zeros. Then, for each pixel of that new image a value between 0 and 45 is assigned, out of a discrete uniform distribution. Finally, the generated image is added to the original with respect to "uint8" type. If any pixel's value is exceeding the minimum or maximum of the range: [0, 255], it is trimmed to the nearest legal value.

Figure 21 Original image (left), original image with additive noise (right)

3.3.2 Horizontal flip

Original image is flipped horizontally only, which does not alter the content, in contrast with vertical flipped echocardiograms, that are not a common case in real time echocardiography.

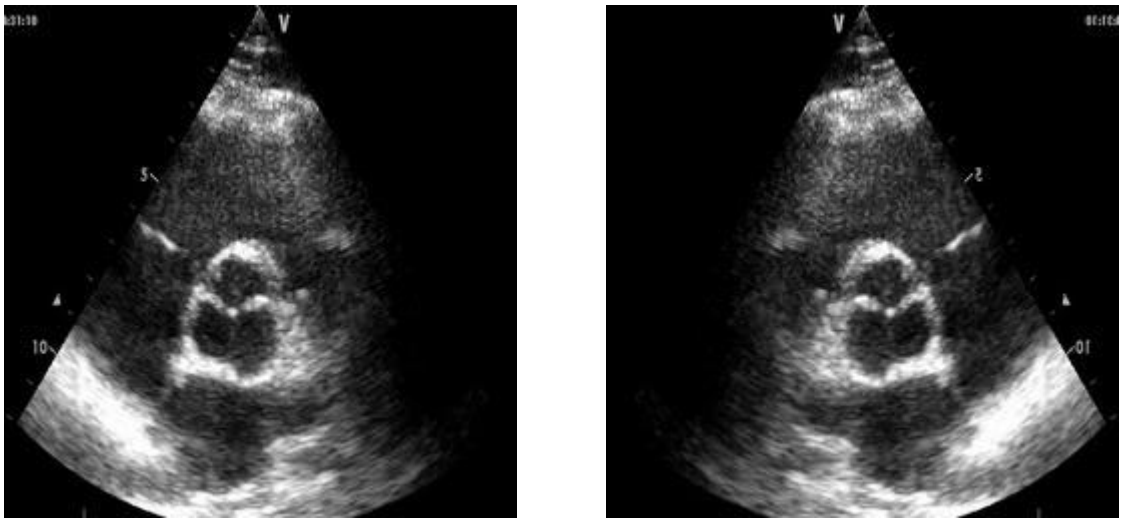

Figure 22 Original image (left), horizontally flipped image (right)

3.3.3 Jittering

Jittering is a technique that randomly increases or decreases intensity levels in pixels by introducing small variations in the original image. This method is usually applied by adding or subtracting small values in range [1, 4] as [20] proposes. On our data this had no effect on frames, since they had relatively small and large values, which were near 0 and 255 respectively. In this study, the contrast of the dataset, can be altered by arbitrarily multiplying by 1.25, which means 25% increase in intensity, or by 0.75, which translates into 25% decrease of the pixels' values. This contrast transformation can be expressed using the following equation:

$$jittered\ image\ = \begin{cases} original\ image \times 1.25, & 50\%\ \text{propability to be applied} \\ original\ image \times 0.75, & 50\%\ \text{propability to be applied} \end{cases}$$

Figure 23 Original image (left), jittered image (right). At the lower left part of the jittered image we can observe an increase in intensity of pixels.

3.3.4 Translation

Image translation is a transformation that slightly moves the objects of the image, by shifting its location within the image's boundaries. Interpreted objects in translated images have different position compared with the initial image. The transformation matrix describing translation is:

$$T_M = \begin{bmatrix} 1 & 0 & t_X \\ 0 & 1 & t_Y \end{bmatrix},$$

where t_X and t_Y are the amount of shift to be applied in x and y axes respectively. To randomize the shifts, two extra parameters "p" and "range" were introduced, representing a uniform probability and the maximum translation range. The values of t_X and t_Y were calculated using the formulas:

$$t_X = range \times p - \frac{range}{2},\ t_Y = range \times p - \frac{range}{2}$$

The selected range is a scale of 5 pixels, due to the fact that the echocardiogram did not disappear from frames. Probabilities were generated from a uniform distribution.

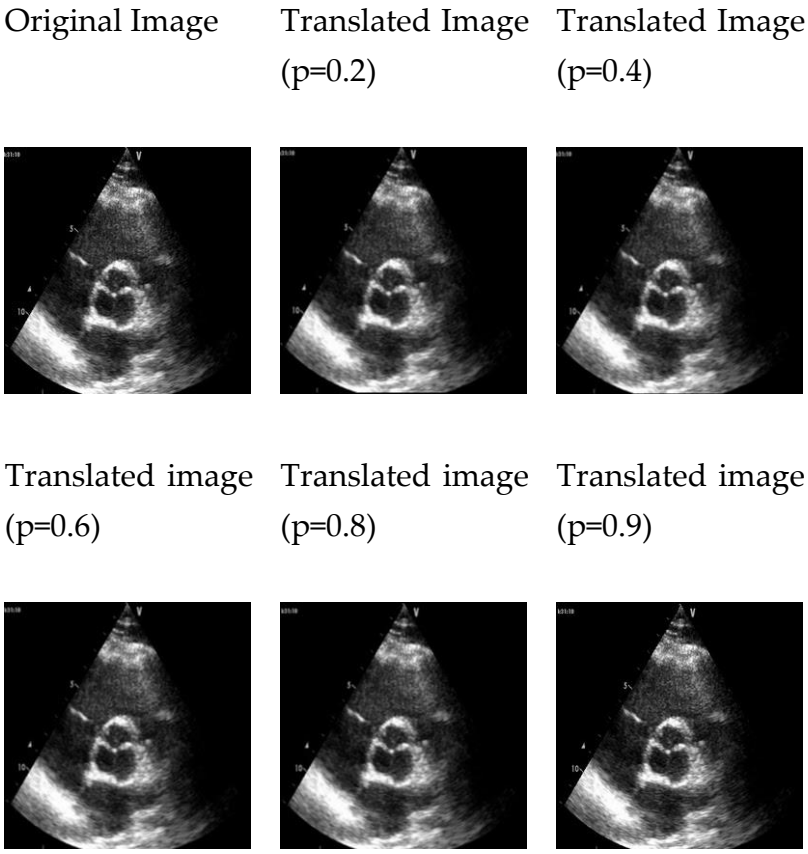

Figure 24 Original and translated images with different values of p.

With translation we can simulate the cropping error that may occur during preprocessing, especially in the tricuspid case, that original images were not centered. The term "cropping error" refers to the main region of echocardiogram appearing in the cropped image, due to variations in image sizes among all cases. The output of translation may seem to be the same, since the echocardiograms inside were moved in a small range, in order to be entirely in the images' borders.

3.3.5 Shearing

The last augmentation technique is shearing which applies an affine transform on the source image. Affine transformations are also called collinear, since all parallel lines of the original image are still parallel in the resulting image. The shearing technique is performed to partly simulate the movement of the sonographer's wrist and the aforementioned cropping error. For using this method, two transformation matrices must be initialized containing three

points from input image and their corresponding position in the output image. The selected points from input image are:

$$P_1^{In} = (5,5), \qquad P_2^{In} = (20,5), \qquad P_3^{In} = (5,20)$$

Variables "range" and "p" were used again to randomize the selection of the position of the corresponding points in the output image, which they can be computed with the following equations:

$$P_1^{Out} = (pt1,5),\ P_2^{Out} = (pt2,pt2),\ P_3^{Out} = (5,pt2),$$

with $pt1 = 5 + range \times p - \frac{range}{2}$ and $pt2 = 20 + range \times p - \frac{range}{2}$

The probability range was set to [0.3, 0.6] uniformly generated, as well as the value of range and other constants were selected in a manner that would not alter the output dramatically.

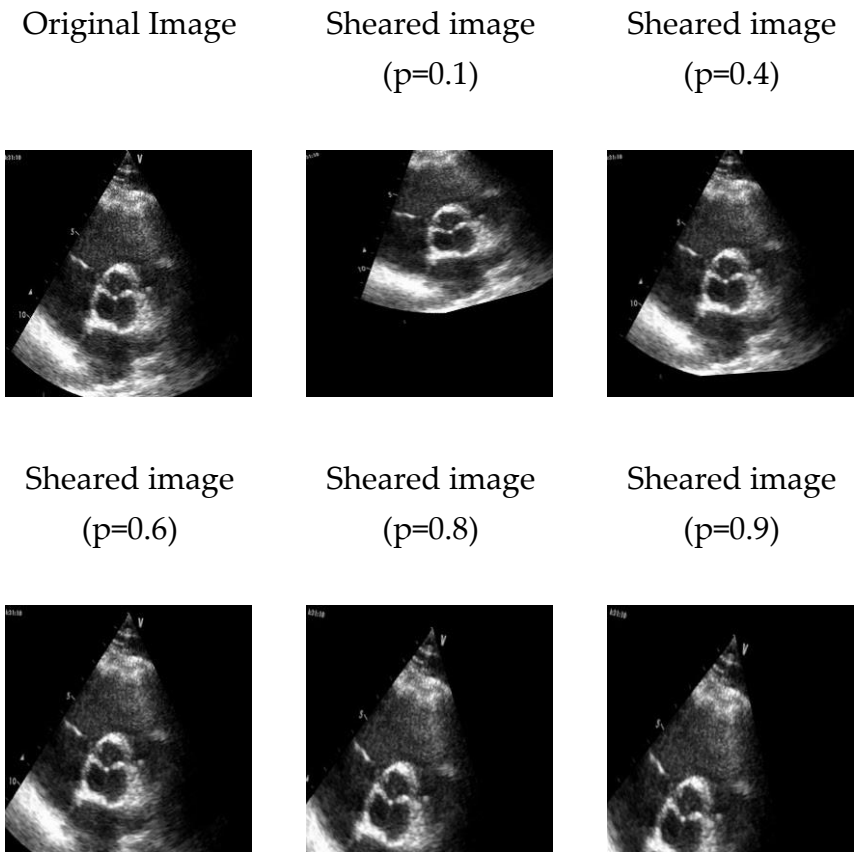

Figure 25 Original and sheared images for different values of p.

The tables below summarize the increase of the dataset's size:

	Tricuspid		Bicuspid		Raphe		Total	
	Videos	Images	Videos	Images	Videos	Images	Videos	Images
Initial size	28	60	8	11	29	29	65	100
Augmented size	476	1020	136	187	493	493	1105	1700

Table 3 Size of dataset before and after augmentation.

	Normal		Abnormal	
	Videos	Images	Videos	Images
Initial size	28	60	37	10
Augmented size	476	1020	629	680

Table 4 Comparison of the size of the two classes

In the proposed augmentation schema, five techniques were implemented for increasing dataset size. Additionally, with the first three techniques mentioned and the last two which have probabilities as arguments, the resulted dataset size was 17 times larger than the initial. Hence, a more robust model can be developed, capable to achieve higher performance.

3.4 Implementation of 3D VGG16 network architecture

This section contains an extended description of the main implementation of current thesis, which is focused on developing deep convolutional neural networks for classifying the configuration of the aortic valve. The selected architecture is derived from the VGG16 network which Karen Simonyan and Andrew Zisserman [13] introduced in 2015. Although, there are more modern neural network architectures such ResNeXt-50 [21], Inception-v4 [22] and Xception [23], we stick to VGG16 since there is a wide range of applications that utilize this network and noise in input images does not drastically affect its performance [15].

In order to use VGG16 for video classification we had to expand it from 2D network to 3D. Thus, we replaced all 2D convolutions with 3D, as well as all 2D Max pooling operations were extended to 3D. Despite those changes, the network was impossible to be trained, due to the computational resources that were needed to be allocated in order to carry out the calculations. To deal with resource limitation, we reduced the amount of parameters to be trained.

The resulting architecture consists of 5 convolutional blocks, each having two or three 3D convolutional layers with Relu as the activation function and the last convolutional layer is followed by a 3D Max pooling operation. In the first convolution block the convolutional layers have 32 3D convolution kernels. The next block consists of two convolutional layers with 64 filters. Next three blocks have three convolution layers with 128, 256, 256 filters correspondingly. Lastly, there are two dense layers with 2048 nodes each and a Dense layer with only two nodes with softmax activation, forming a fully connected prediction network. Batch normalization between all convolutional layers and dropout with 50% rate were included between dense layers, for preventing network from overfitting. The figure 26 presents our 3D-expanded VGG16 architecture:

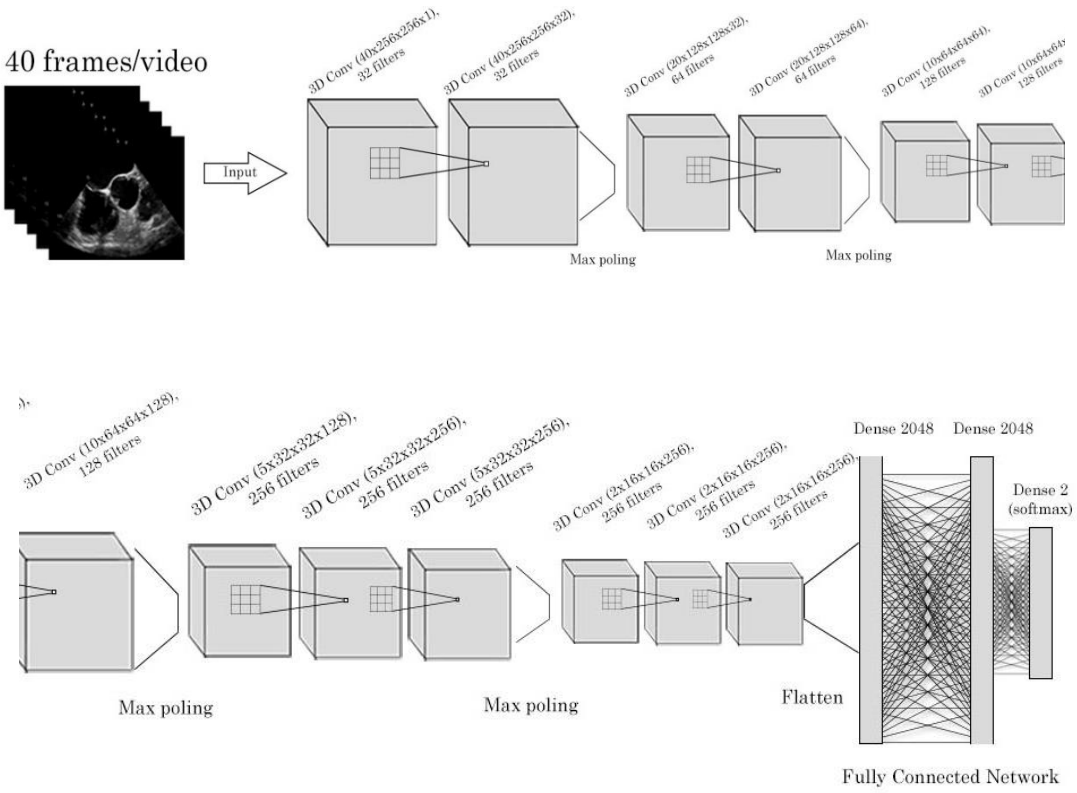

Figure 26 3D architecture implemented for the purposes of the study.

3.4.1 Replacing Fully Connected layers with an SVM clasifier

To further extend the VGG16 architecture, we replaced the Fully Connected part of the network with an SVM classifier with linear kernel. E. Trivizakis et al. [24] proposed to replace the softmax layer with the SVM. We incorporated this idea to our architecture, but instead of replacing the softmax activated layer only, we completely removed the fully connected layer and placed the SVM. The convolutional blocks (feature extraction part of the network) are responsible for feature extraction and the SVM accepts those features as input and classify samples. The feature extraction from convolution blocks and the training of the SVM executed separately, since they could not be embedded in a single network, due to the bottleneck that the feature extraction part introduces. Convolutional kernels were initialized using the corresponding trained weights from previous network (figure 28).

The altered architecture is presented in the figure below:

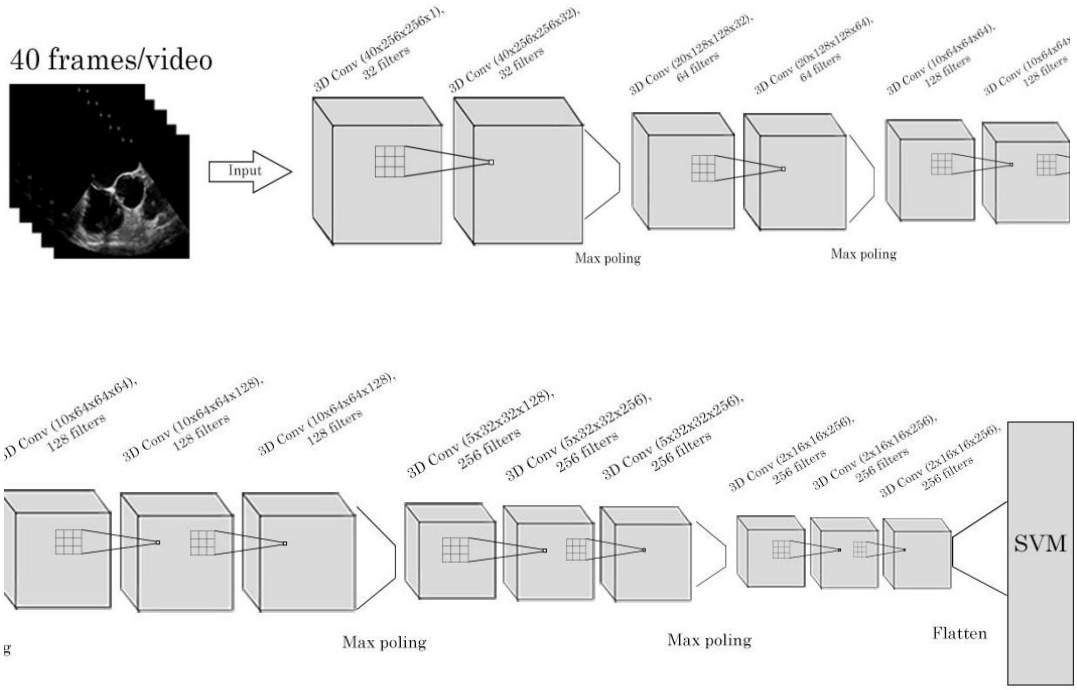

Figure 27 Altered 3D architecture with an SVM replacing the fully connected network.

All 3D architectures, including a detailed description of the input and output of each layer are presented in Appendix A.

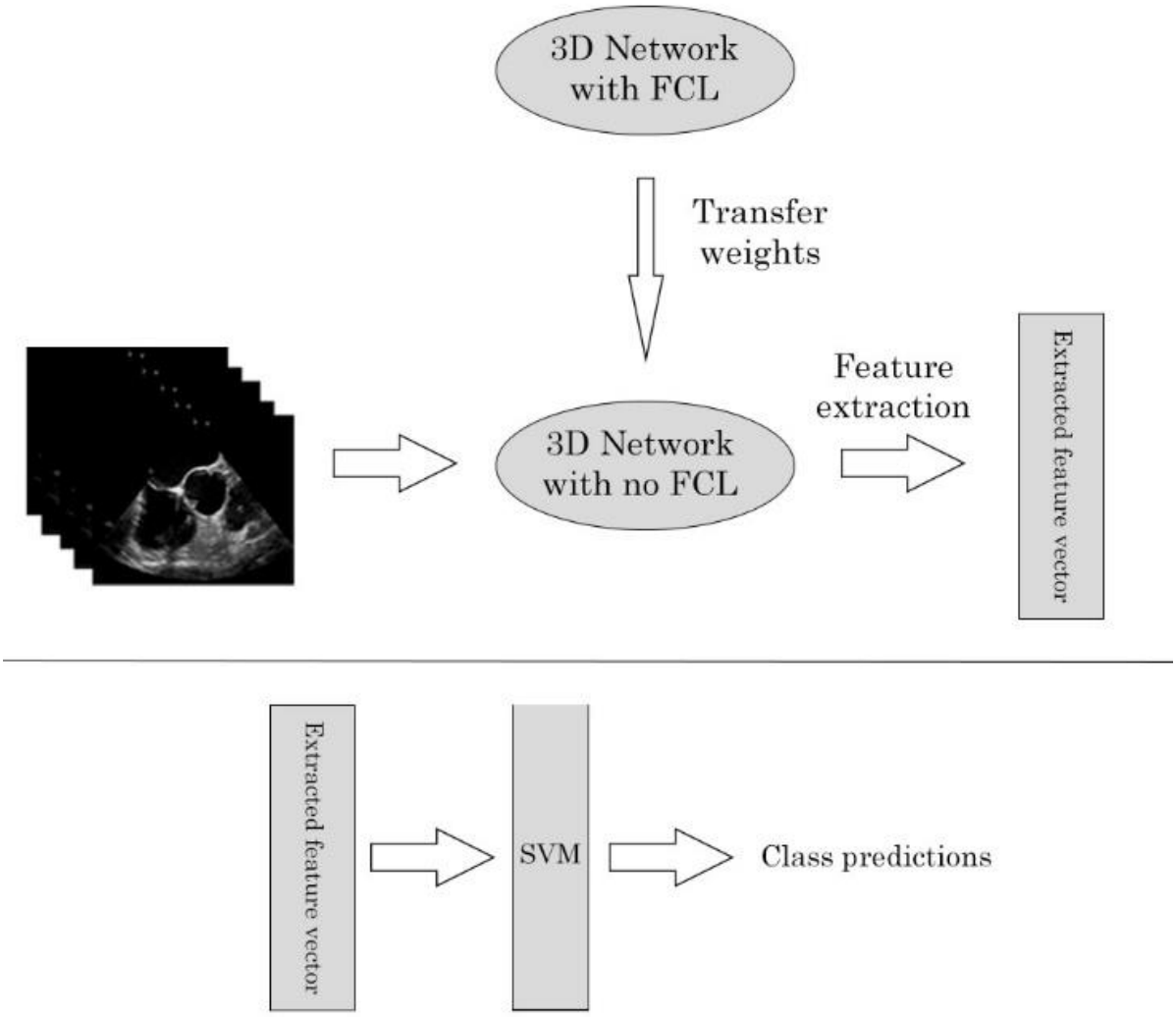

Figure 28 Weights from the trained 3D network were used to extract features from videos. The extracted features then were used to train and test the SVM classifier. For the training of the SVM were used the same data samples as for the training of the 3D network.

3.4.2 Extention of 2D filters to 3D

For implementing transfer learning, 3D weights from a similar architecture must be acquired. This acquisition is not possible, since there is no other 3D network following the exact same architecture. Thus, we had to expand the weights from various 2D networks to 3D to be fitted in our implementation. In order to perform the expansion, we extracted every single filter from each convolutional layer and then we stacked it three times in order to form a 3x3x3

cube. This cube was the 3D filter that was embedded in the same position in the corresponding layer of our 3D network.

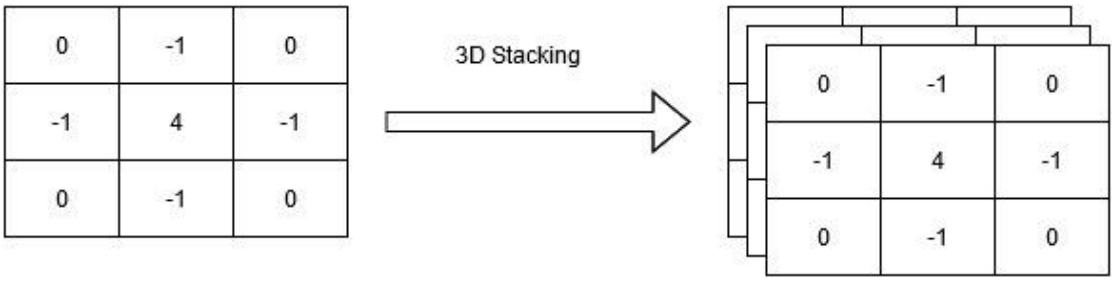

Figure 29 2D to 3D expansion of the trained weights. This figure represents the expansion of the filters in the first convolutional layer. A 2D 3x3 filter is stacked in order to form a 3D 3x3x3 filter.

The first layer in our architecture has 32 initial filters, in contrast with the original VGG16, which has 64. During expansion we ensure that only the first 32 filters of the first layer of the 2D network are expanded, in order to accelerate the procedure. This process is repeated for every convolutional layer, using the first half of its filters.

3.5 Experiments

After implementing the main network and the SVM expansion on it, we started experimenting with the available dataset. Firstly, we split videos and images to fixed train and test sets, containing 80% and 20% of the available data correspondingly. We also used 20% of the train set for validation during the training of the network with the augmented data. All experiments were executed in the Google Colab[3] platform, using GPU backend for accelerating the training procedures. Training was extremely time-consuming, due to the complexity of operations in a 3D Network. Convolutional kernels were initialized by drawing weights from a Uniform distribution within the range [-a, a] where $a = \sqrt{\frac{6}{I}}$, and I the number of incoming neurons from the previous layer. For every network, we chose Adam optimizer with an adaptable learning

[3] https://colab.research.google.com

rate starting of 10^{-4}, with 50 epochs to minimize cross-entropy loss function and a batch size of 1, due to memory resource limitations. All hyper parameters tuned after multiple trials that were executed on the data before augmentation. The conducted experiments are:

1. Classification of the aortic valve in 2 classes (normal/abnormal) using video data and the 3D network.
2. Classification of the aortic valve in 2 classes (normal/abnormal) using video data and transfer learning in the 3D network.
3. The two experiments aforementioned were repeated using the SVM extended 3D network.
4. Classification of the aortic valve in 3 classes (normal/abnormal) using video data and the 3D network.
5. Classification of aortic valve in 2 classes using images (2D network), with and without transfer learning.
6. The 3D network were re-trained for the 2 class classification problem, using the weights of the 2D network trained on our data.

The first experiment used the video data to classify the configuration of the aortic valve in 2 classes, "Tricuspid" (normal) and "Bicuspid" (abnormal) with both bicuspid and raphe cases included. In the second experiment we used the extension method described in section 3.4.2, to extend the 2D VGG16 weights, which were trained in ImageNet dataset, to 3D. After expanding the weights, we trained the whole network on our dataset again, in contrast with conventional transfer learning, in which only the fully connected layers are trained. In this manner, we accomplished to fix any mismatch that may have occurred in weight expansion and achieve slightly faster convergence during training. Then, experiments 1 and 2 were repeated using an SVM instead of fully connected layers test the SVM-extended network as well. Finally, we wanted to inspect whether the 2-class or the 3-class classifier can achieve better classification performance.

Next, we trained the 2D network using available images, for comparing its performance against the 3D network. In order to clarify, which architecture can achieve higher accuracy we trained the conventional 2D VGG16 network for 100 epochs, instead of 50, using Adam optimizer, with an adjustable learning rate of 10^{-4}, to minimize cross-entropy loss and a batch size of 4. K. Simonyan

and A. Zisserman [13] stated that the training finished at 74 epochs. In our case, the network could not converge with a larger learning rate, hence we executed training for more epochs. This reduction of epochs needed for convergence occurs due to the normalization that the deep network architecture introduces. In addition, the experiment repeated using transfer learning, with the 2D trained weights on ImageNet. During the specific experiment, we used the data showing only the open state of the aortic valve, as the doctor captured them, as well as equal number of frames from the videos that interpret the aortic valve, just before or after it is fully opened. Lastly, in the final experiment, we expanded the 2D weights from previously trained network, on the images with fully open state and compared its performance with the expansion of the 2D weights trained on ImageNet.

Chapter 4 Results

4.1 Normal and abnormal aortic valve classification from video data

In the first two experiments, we successfully accomplished to build a 3D network that can achieve high accuracy while trained on augmented data with and without transfer learning. The evaluation of the performance is summarized in the table below:

3D model trained on augmented data	Without Transfer Learning	With Transfer Learning (Run 1)	With Transfer Learning (Run 2)
Accuracy (%)	93.27	91.47	97.75
Error rate (%)	6.73	8.53	2.25
AUROC (%)	92.18	93.49	97.39
Sensitivity	1	1	1
Specificity	0.8437	0.8020	0.9479
Precision	0.8943	0.8698	0.9621
F1-score	0.944	0.93	0.9806

Table 5 Performance of 3D network trained with augmented data, using random initialization of weights and transfer learning.

Abnormal is defined as the "positive" class with the label "1" and normal as "negative" class with the label "0". Hence, if a video is classified as positive, it means that the patient has a bicuspid aortic valve.

The high performance of the proposed network is confirmed with the high values of sensitivity and specificity, indicating that the network identifies properly the greatest proportion of both positive and negative data samples.

Accuracy and loss graphs for both train and validation phase of the first experiment are shown in the next figures:

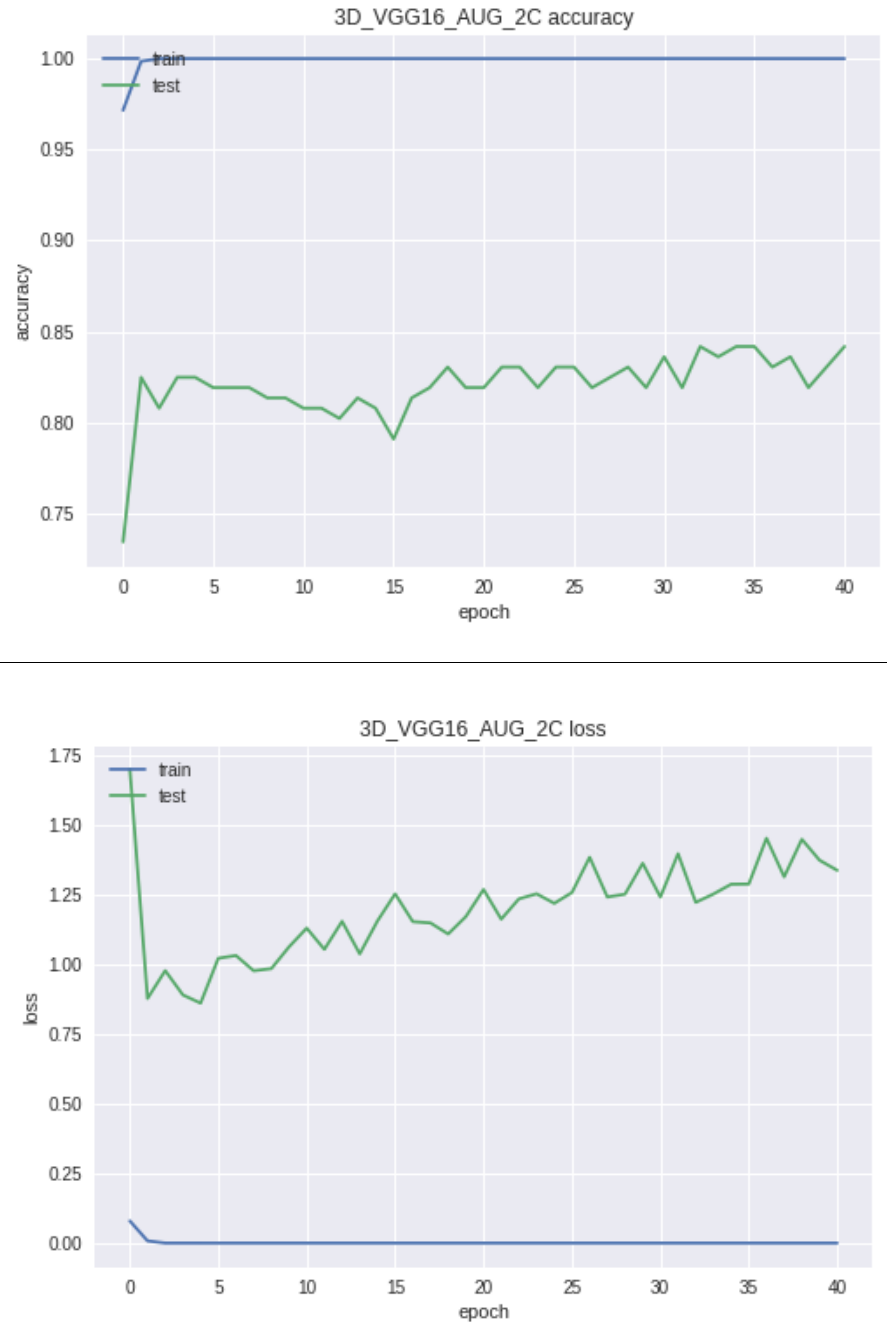

Figure 30 Train and validation phase accuracy/loss for the 3D model without transfer learning.

We can observe that validation accuracy is lower than the train accuracy, due to the small size (176 samples) of validation data, but we observe that the network starts to converge. The confusion matrices are presented as heatmaps:

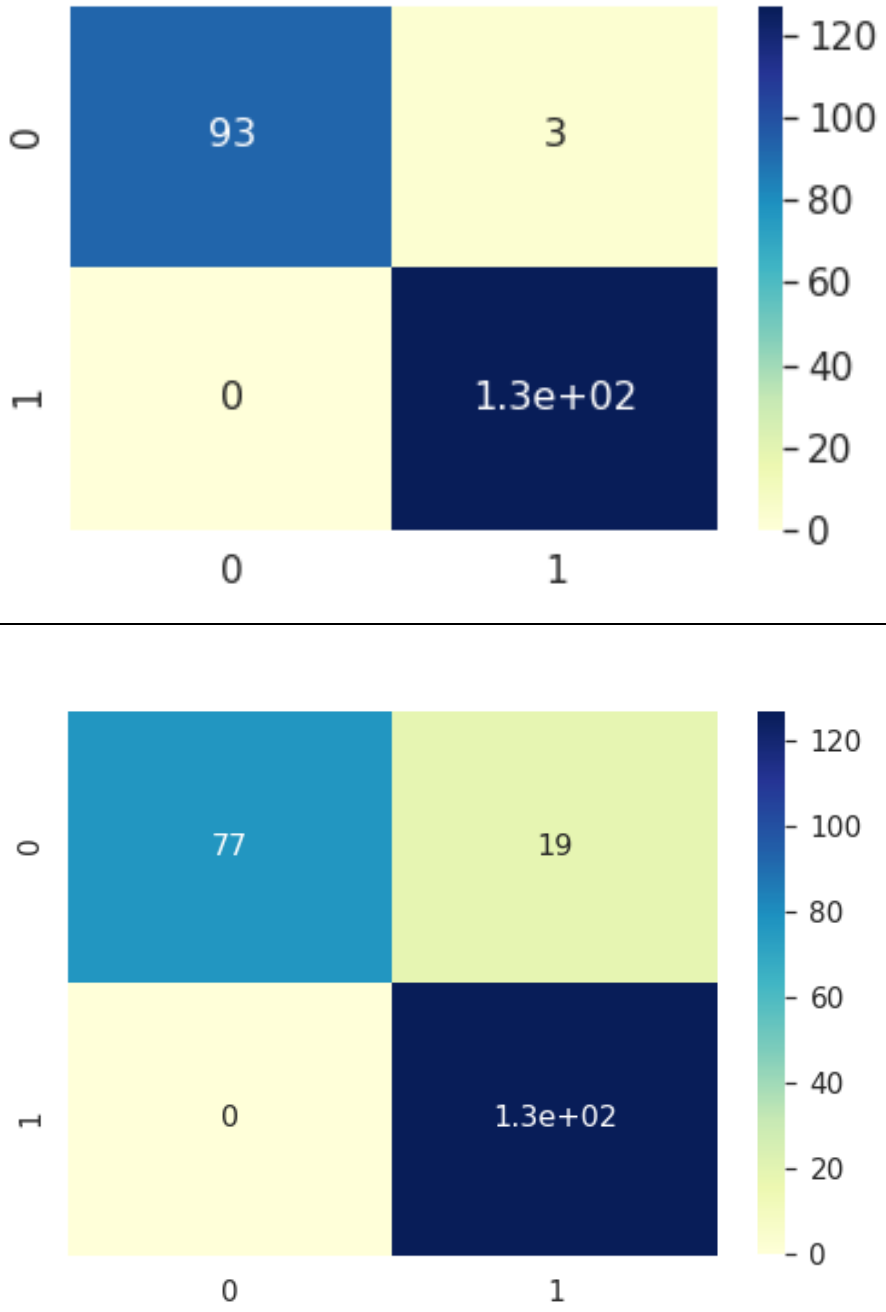

Figure 31 Confusion matrix of 3D model without transfer learning (upper part) and with transfer learning from the first run (below).

The slight drop of the accuracy of the network with transfer learning in the first run is explained considering the random order of the data samples that were used to train the network. Moreover, the lower performance of the first run occurs, because the initial weights were from ImageNet, which contains images with non-medical subject and they had not adjusted correctly, during retraining. In the second run, we can observe an increase in all the computed metrics and the average accuracy reaching 94.61%. For each run the model was trained from scratch. More precise results could have been provided using a 10-fold cross validation method, but the available resources forbid such an operation. The Receiver Operation Characteristic curve of the first run of the 3D network with transfer learning is:

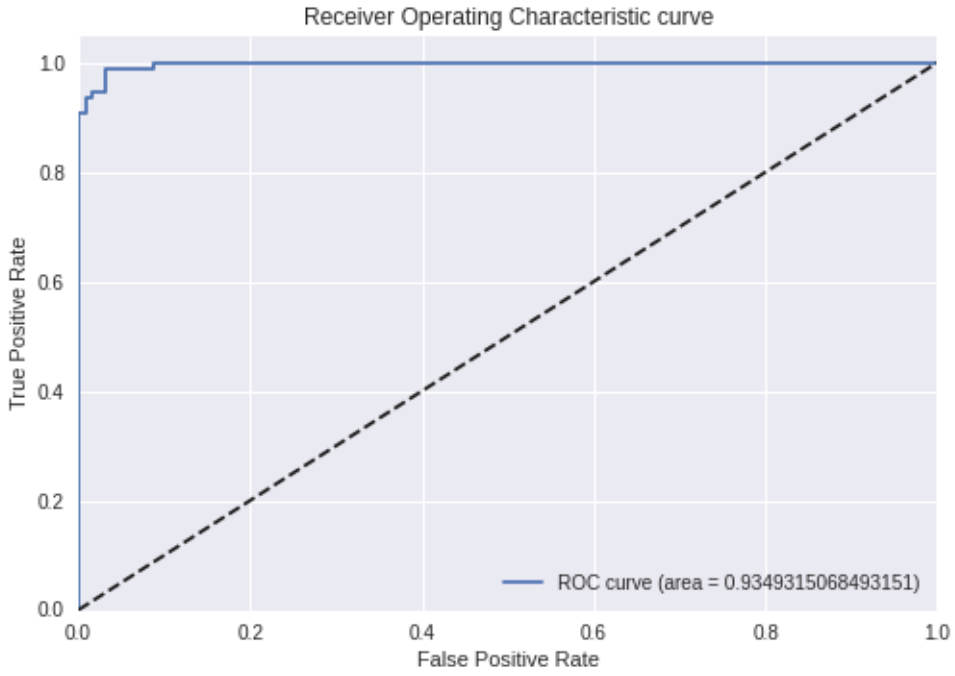

Figure 32 ROC curve of the first run.

The ROC curve is near the ideal curve, since it passes near the left upper corner and is steep.

4.1.1 SVM performance as a classifier

After receiving the first evaluation of our expanded architecture, we executed the network with the SVM classifier. After copying the weights from the first run of the second experiment to the feature extraction part of the network, we

created the feature vector for each sample (for both train and test set) and then we trained the SVM with videos' features from the train set. Next, we tested our SVM expanded architecture and received the following results:

3D model with SVM trained on augmented data	Without Transfer Learning	With Transfer Learning
Accuracy (%)	97.28	98.64
Error rate (%)	2.72	1.36
AUROC (%)	96.87	98.43
Sensitivity	1	1
Specificity	0.9375	0.9687
Presision	0.9541	0.9765
F1-score	0.9766	0.9881

Table 6 Evaluation of 3D network with SVM with and without the use of transfer learning.

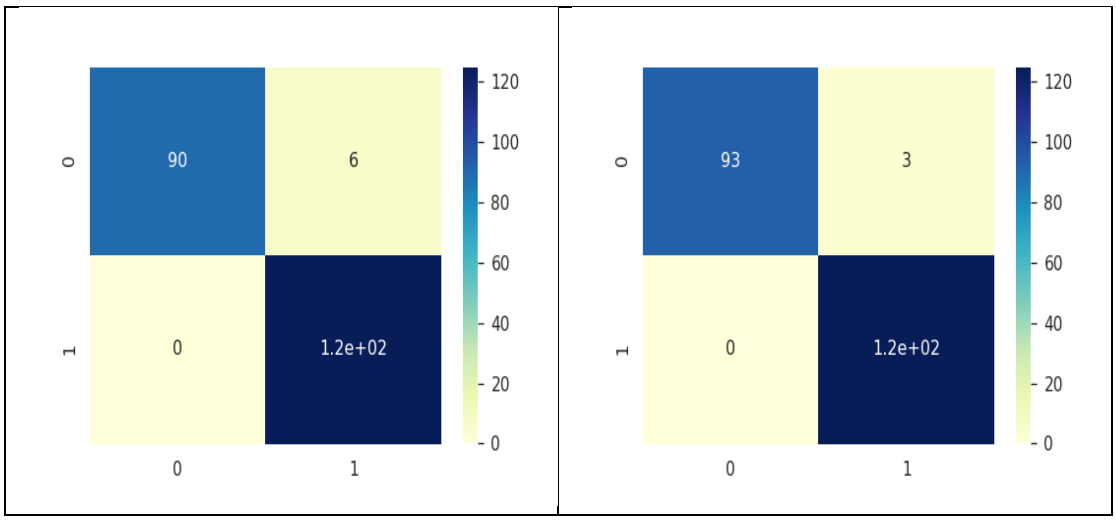

Table 7 SVM expanded network without transfer learning (left) and with transfer learning (right).

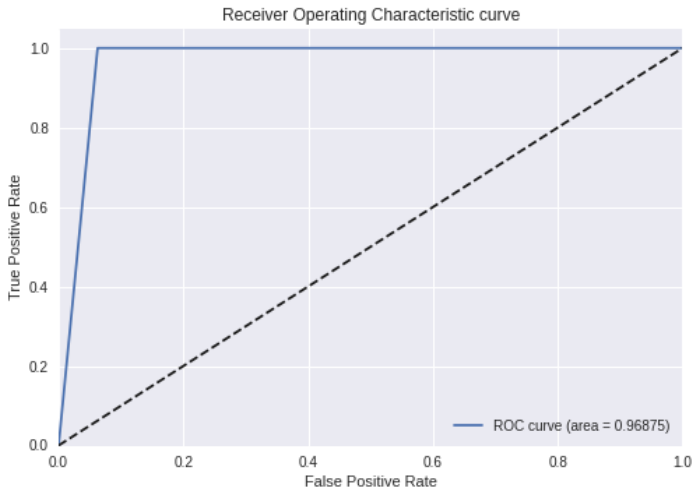

Figure 33 ROC curve of SVM expanded network without transfer learning.

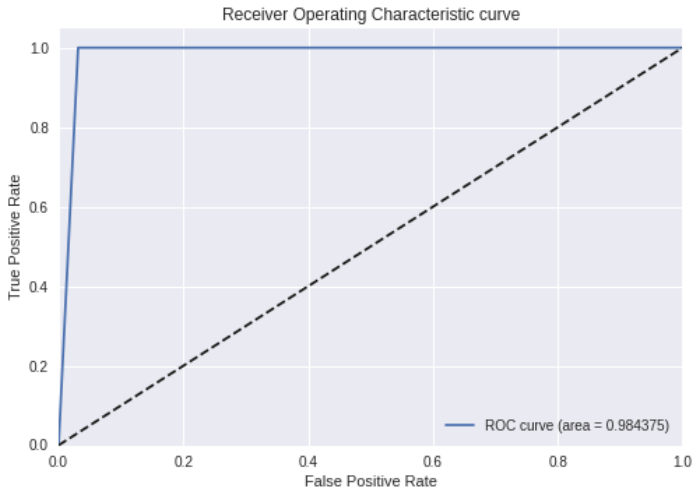

Figure 34 ROC curve of SVM expanded network with transfer learning.

The results showed that the use SVM instead of fully connected layers can only benefit the network's performance. SVM classifier outperforms fully connected layers, since it needs less training and predicting time, while achieving higher accuracy. Area Under Curve metrics indicate that the classifier is capable of distinguishing the two classes more precisely than the conventional fully

connected layers. Finally, observing ROC curves make it clear that the network trained with transfer learning and the SVM is more capable than the one with no transfer learning, since the ROC curve is steeper. In the SVM experiments the normal cases are classified more accurately compared with the use of fully connected layers.

4.2 Tricuspid, Bicuspid and Raphe classification

Next, we investigated further the performance of our proposed architecture, by splitting the data in all three available classes rather than two which was used in the previous experiments. For that reason, we replaced the last dense layer with another that had 3 nodes with softmax activation. All weights were re-initialized from a uniform distribution as before, while all other hyper parameters remained constant. Finally, we set a class "0" to be tricuspid, "1" as bicuspid and class "2" as raphe.

4.2.1 Expanded network with no transfer learning for distinguishing 3 classes

In our first 3 class classification experiment, the network achieved an overall accuracy of 74.44% while it managed to recognize almost all tricuspid samples correctly. This translate into the network is biased to the tricuspid case. The confusion matrix is presented below:

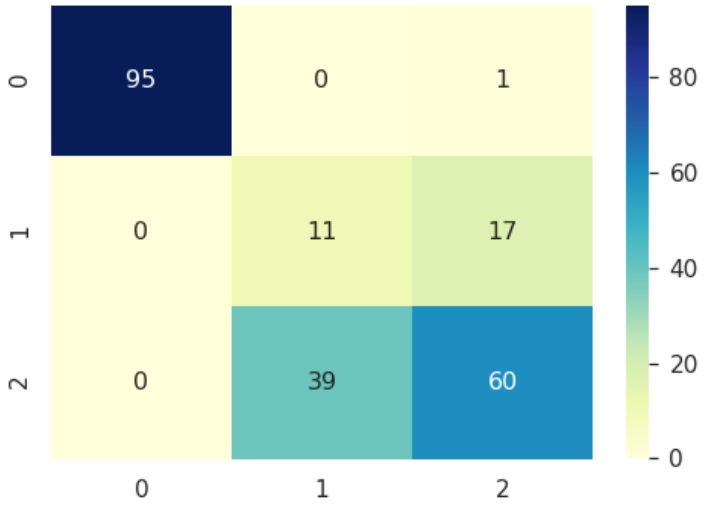

Figure 35 Confusion matrix of 3D network for 3 class classification. Label "0" is tricuspid, "1" is bicuspid and "2" is raphe.

Looking at the train and validation accuracy and loss graphs, we observe that the network converged faster, since there are small ripples after 15 epochs in all the waveforms, near final accuracy and loss accordingly as shown in the following graphs.

Figure 36 Train and validation accuracy/loss for the 3D model in 3 class classification.

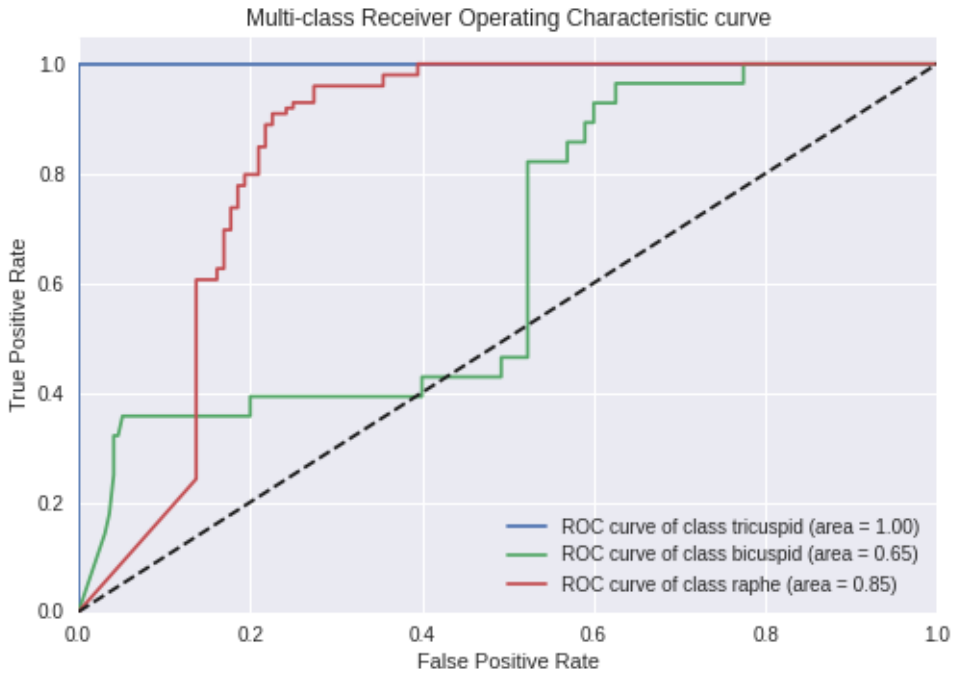

Figure 37 ROC curve of 3D network trained from scratch.

Classifying samples of bicuspid class is the most challenging part, as the class's ROC curve indicates. The other two classes have a higher Area Under Curve, proving that the network can classify samples from those classes with greater ease. In spite the high performance, we conjecture that the result is not reliable enough, in this case, since the bicuspid class has less data than the other two.

4.2.2 Expanded network with transfer learning for distinguishing 3 classes

In addition to the previous experiment, we aimed to explore whether transfer learning will boost the performance of this network. In order to accomplish that, we initialize the weights with the expanded ones that were trained on ImageNet. Transfer learning, indeed, increased the performance with overall accuracy raised up to 83.41%.

Train and validation graphs are presented in the following figure:

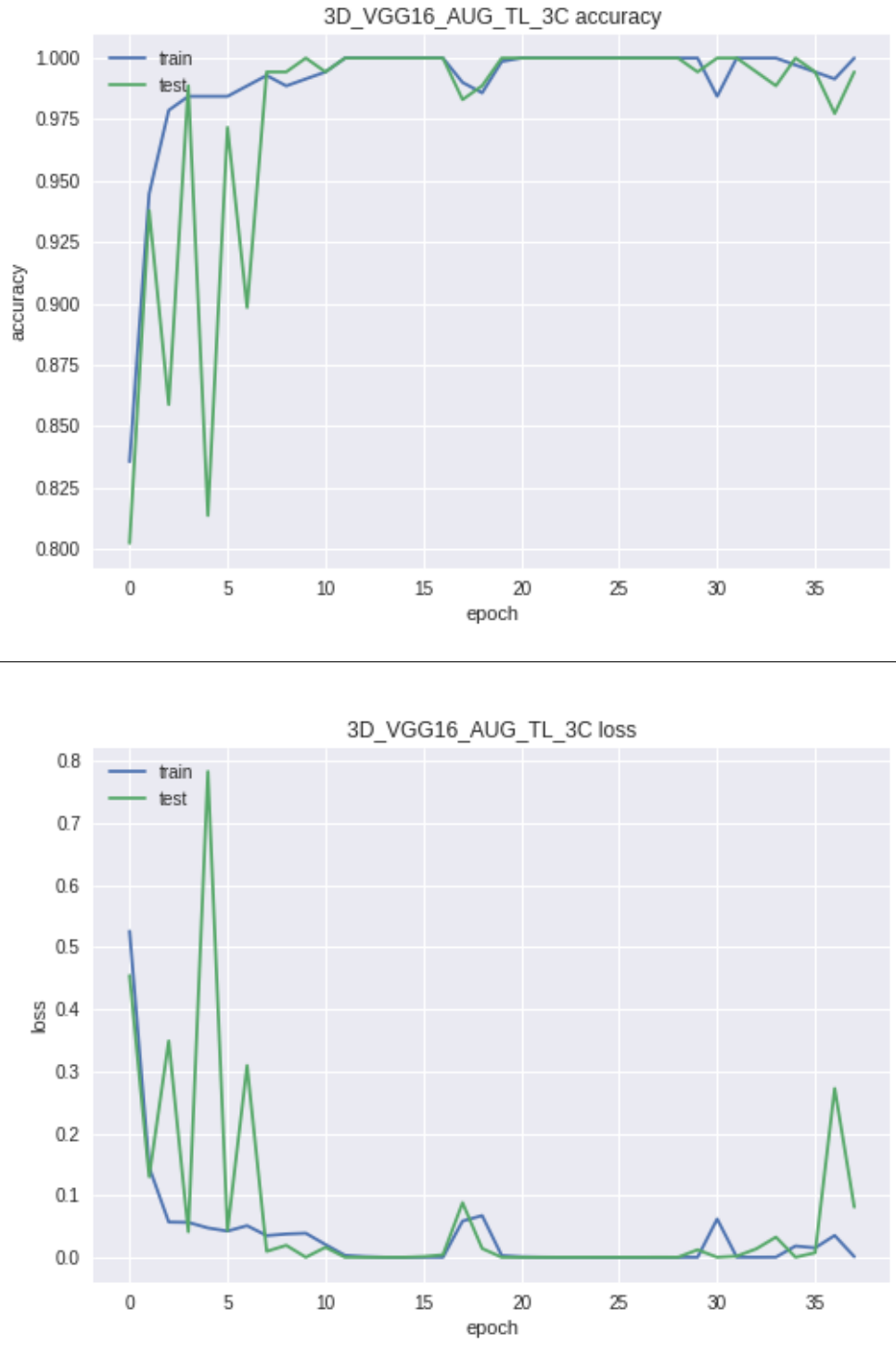

Figure 38 Train and validation accuracy/loss for the 3D model with transfer learning in 3 class classification.

The confusion matrix and ROC curve for this experiment are:

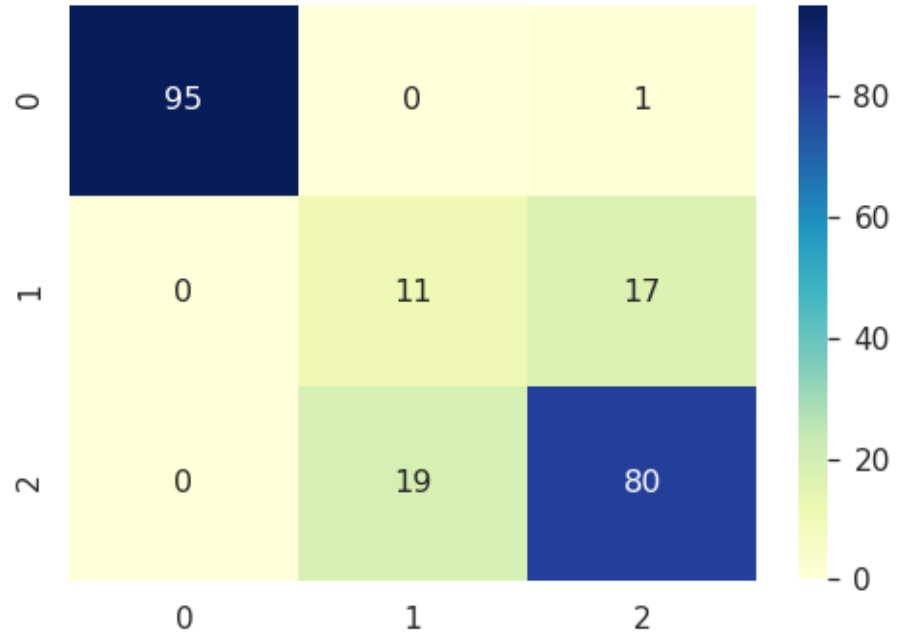

Figure 39 Confusion matrix of 3D network with transfer learning for 3 class classification. Label "0" is tricuspid, "1" is bicuspid and "2" is raphe.

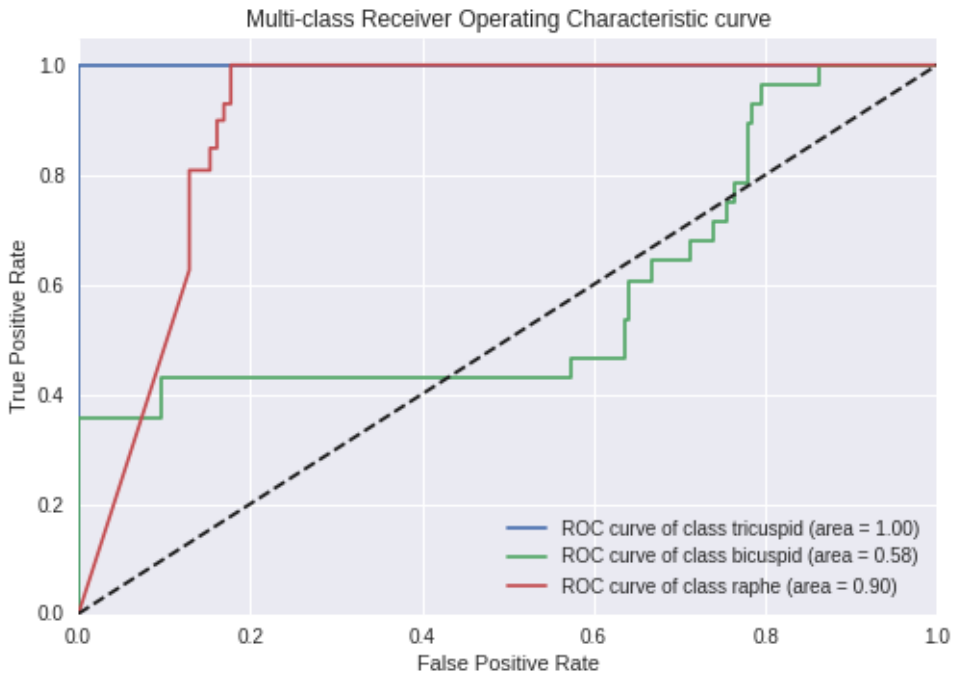

Figure 40 ROC curve of 3D network trained using transfer learning.

The following table summarizes the calculated metrics per class:

		Without transfer learning	With transfer learning
Overall accuracy (%)		74.44	83.414
Per class metrics			
Accuracy (%)	Tricuspid	99.55	99.55
	Bicuspid	74.88	83.85
	Raphe	74.43	83.40
AUROC score (%)	Tricuspid	100.00	100.00
	Bicuspid	65.00	58.00
	Raphe	85.00	90.00
Sensitivity	Tricuspid	0.9895	0.9895
	Bicuspid	0.3928	0.3928
	Raphe	0.6060	0.8080
Specificity	Tricuspid	1	1
	Bicuspid	0.8000	0.9225
	Raphe	0.8500	0.9025
Precision	Tricuspid	1	1
	Bicuspid	0.2200	0.3666
	Raphe	0.7600	0.8163
F1-score	Tricuspid	0.9947	0.9947
	Bicuspid	0.2820	0.3793
	Raphe	0.6779	0.8121

Table 8 Metrics calculated for the specific experiment

In the case of three class classification, we observed that transfer learning increased network performance, especially in the raphe cases where more samples were classified correctly. As mentioned before, the large class imbalance played a critical role in bicuspid samples misclassification, where only 98 used for training out of a total 136, and were not constrained with the presence of transfer learning.

4.3 Normal and abnormal aortic valve classification from images

For this experiment, we used two image sets as mentioned in section 3.5. The first set consists of 100 images interpreting the fully open state as the experienced specialist captured them. For the second set, we thoroughly examined the initial frames of videos and selected only the frames which interprete the aortic valve just before or after it was fully opened. The two equally sized sets were preprocessed and augmented with the same methods used for video frames.

4.3.1 Training with frames extracted from specialist

Acquiring a clear view of the fully opened state of the aortic valve is challenging, since this happens momentarily. Cardiologists are trained to understand immediately the open state, therefore the image selection for this experiment has a great quantity of clear, high quality images. The network trained on those images achieves an accuracy of 93.82%, while transfer learning increases the accuracy to 97.94%. The confusion matrices are presented below:

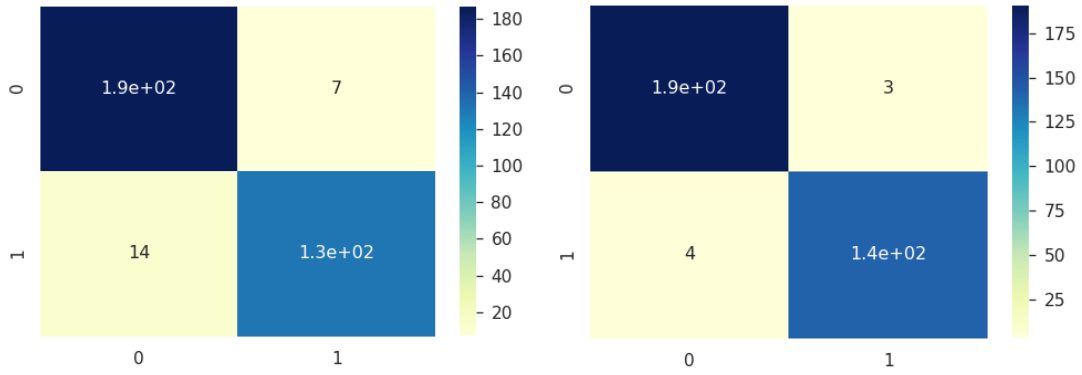

Figure 41 Confusion matrix of 2D network trained on specialists extracted data, without transfer learning (left) and with transfer learning (right)

All calculated metrics are presented in the following table:

2D model trained on cardiologist extracted images	Without Transfer Learning	With Transfer Learning
Accuracy (%)	93.82	97.94
Error rate (%)	6.18	2.06
AUROC (%)	93.99	97.93
Sensitivity	0.9041	0.9726
Specificity	0.9639	0.9845
Presision	0.9496	0.9793
F1-score	0.9263	0.9759

Table 9 Metrics calculated upon 2D network trained on cardiologist extracted images.

From Table 9 it is concluded that all the metrics indicate the high performance of network.

4.3.2 Training with frames extracted using ECG waveform

Next, we initialized 2D VGG16 with weights pre-trained on ImageNet, in order to train the classifier to fit the video frames that we extracted near the open state of the valve, using the ECG waveforms. The metrics that were calculated are presented it the table below:

Accuracy	Error rate	AUROC	Sensitivity	Specificity	Presision
95.21%	4.79%	0.9655	1	0.8648	0.931

Table 10 Metrics calculated upon 2D network trained on video frames near open aortic valve, using ECG waveform.

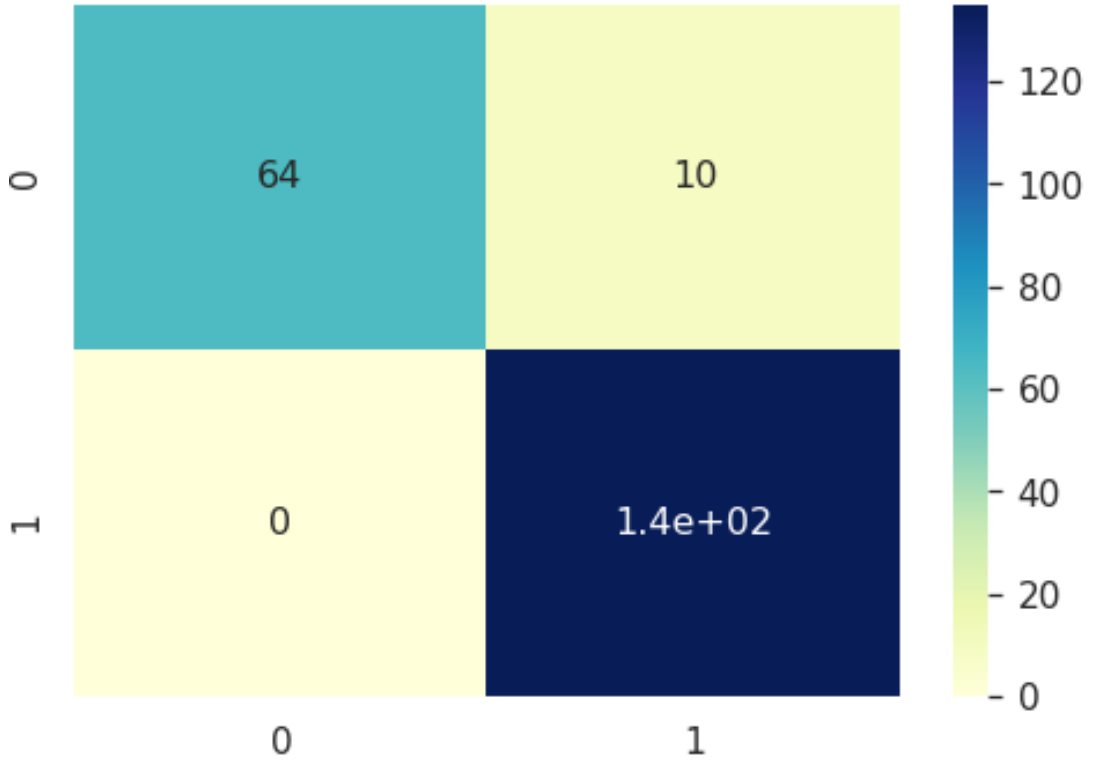

Figure 42 Confusion matrix for 2D network trained on video frames.

Apparently, network trained on fully opened state images achieves higher performance, since it is clear if the valve is tricuspid or abnormal, except some naturally noisy images. It is worth mentioning that training the network with the extracted data, eliminates the misclassification of the abnormal class, as confusion matrices indicate. This depends on the selected frames; hence the 2D network can achieve higher performance when the aortic valve is captured exactly at the open state.

4.3.3 Transfer weights from 2D trained network to 3D

Finally, we tested the 3D network using the weights from the 2D network trained with transfer learning, on images extracted by the specialist. In this manner we accelerate the training of the 3D network. Specifically, we expanded the weights trained on image data, provided by the cardiologist, without the use of transfer learning. As a result the 3D network achieved 97.30% accuracy, reaching the initial performance of training on video with transfer learning.

Metrics calculated upon it, the corresponding confusion matrix and the ROC curve are presented below:

Accuracy	Error rate	AUROC	Sensitivity	Specificity	Presision
97.30 %	2,7	97.77 %	1	0.9375	0.9575

Figure 43 Metrics calculated upon 3D network initialized with weights trained on provided image data.

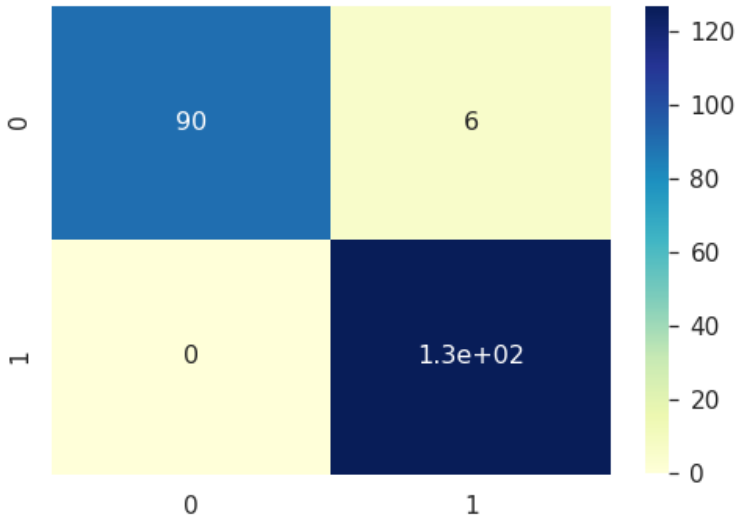

Figure 44 Confusion matrix of the 3D network with transfer weights from 2D network.

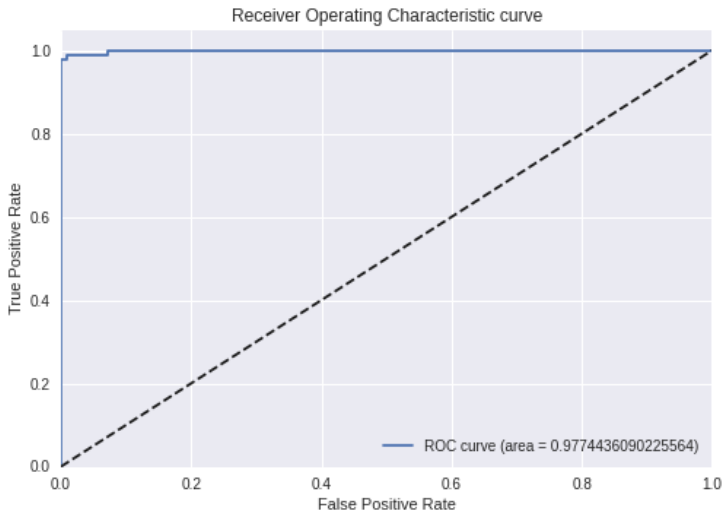

Figure 45 ROC curve of 3D network with transfer weights from 2D network.

All metrics, indicate the high performance of the network, as well the ROC curve is near the optimal.

Chapter 5 Discussion

In this study, we presented our implementation of a deep neural network that can successfully classify the configuration of the aortic valve, introducing a 3D model for video analysis. The proposed 3D expansion of VGG16 network achieved 97.75% accuracy while using the expanded 2D weights trained on ImageNet. We showed that replacing the fully connected layers with an SVM classifier can boost the performance and achieve up to 98.64% accuracy, providing some methodological improvements.

The three class classification experiment needs to be executed with more data, since there is significant class imbalance among the three classes; something that does not occur in the 2 class classification. Nevertheless, the accuracy of 83.41% seems promising and the network should be tested using more data and cross validation. Finally, a new network can be introduced in order to achieve better classification among abnormal configurations of the valve, since our architecture can fully identify the normal cases.

The conventional 2D VGG16 network achieved 97.94% accuracy on cardiologist extracted data and 95.21% accuracy on our set of selected images. This selection happened, because we aimed to prove that the network is capable of separating the configurations of the aortic valve not only from clear cut images, but also from selected frames interpreting the aortic valve near the open state. Thus, there is a comparison between 2D and 3D models on the same application. Our conclusion was that the 2D network needs images interpreting the valve in the fully open state, in order to achieve higher performance.

Transferring the learnt weights from 2D to the 3D network showed that whether the weights were trained from similar dataset, either from images with different subject, transfer learning achieves higher performance and faster convergence. This happens because weight initialization is not random, but already contains common patterns across all image sets. For instance, edges exists in all images from different backgrounds; thus the first layers will learn to detect them equally. Going deeper in the network architecture, we understand that learned shapes and textures have significant differences from dataset to dataset. Hence, this differentiation might reduce the accuracy of transfer learning. This is the main reason we implemented transfer learning by initializing the network with 3D expanded weights and then trained it from

scratch, rather than training only the fully connected layers, as in conventional transfer learning. It is worth mentioning that the use of this technique also requires a satisfactory amount of data, so the last layers can alter the previously learnt patterns in order to make them fit the available data.

5.1 Study limitations

This study has some major limitations which should be taken into consideration. The proposed 3D architecture could not fit in a common GPU since it uses at least 10GB of memory for storing the network's weights and processes them in the training phase. This procedure is extremely time-consuming, since the 3D convolution is more computationally expensive than 2D. High memory usage and long running times were the main reasons that we could not apply cross validation to our network, since the platform used introduces memory and execution time limits for users.

The major limitation is the small dataset size, which in the three class classification experiment, reduced the precision of the metrics calculated upon the network due to the class imbalance. The proposed augmentation schema helped constraining this limitation, but more data would give more precise results. An important fact that should be mentioned is that data samples of the bicuspid class are extremely hard to be acquired. Considering that the current population on earth is around 7.8 billion people and only 1% to 2% [3] of that global population is estimated to have bicuspid aortic valve, nearly 150 million of people have this cardiac anomaly. This translates into difficulties in accessing, gathering and managing those datasets.

5.2 Future work

Despite the limitations, this thesis presents some promising results in classification of the aortic valves configuration. As seen in the literature there is not any study that involves with this task. In order to extend our work, Class Activation Maps (CAM) [25] could be used for the identification of the regions of the input image or video that activates the network the most; in other words the important parts of the input. Therefore, we could demystify how our extended 3D deep learning network makes its classification decisions and

invalidate the “black box” characterization. Moreover, new 3D kernel formation and training techniques can be introduced.

To further extend our work, 3D semantic segmentation seems a challenging next step by extending U-Net [26] to fit 3D data. This will further help cardiologists to diagnose faster since they will be able to reconstruct the valve and observe its function in real time. Another step toward this direction is to replace contraction path with our 3D VGG16 architecture as Pravitasari et al. [27] introduced in the 2D VGG16 network. This will accelerate the training procedure of U-Net, since only the encoding path of the network should be trained.

Finally, training the proposed network is a time consuming process due to the great amount of the required computational resources, as mentioned in the previous section. Hence, there is an imperative need to reduce the execution time as well as the memory used to perform operations. In this effort, richer computational resources must be used in order to be able to test the network with more data and therefore get more precise results. An excellent starting point would be to use Field Programmable Gate Arrays (FPGAs) to accelerate the training procedure. Geng et al. [28] proposes a scalable framework for training convolutional neural network using FPGAs. Thus, not only the execution times will be reduced, but also it will make feasible the creation of a more compact, fast and portable echocardiographic device.

References

[1] H. H. Sievers and C. Schmidtke, "A classification system for the bicuspid aortic valve from 304 surgical specimens," *J. Thorac. Cardiovasc. Surg.*, vol. 133, no. 5, pp. 1226–1233, 2007, doi: 10.1016/j.jtcvs.2007.01.039.

[2] T. Liu *et al.*, "Bicuspid aortic valve: An update in morphology, genetics, biomarker, complications, imaging diagnosis and treatment," *Front. Physiol.*, vol. 10, no. JAN, pp. 1–17, 2019, doi: 10.3389/fphys.2018.01921.

[3] A. Della Corte *et al.*, "The ascending aorta with bicuspid aortic valve: A phenotypic classification with potential prognostic significance," *Eur. J. Cardio-thoracic Surg.*, vol. 46, no. 2, pp. 240–247, 2014, doi: 10.1093/ejcts/ezt621.

[4] M. A. Sadron Blaye-Felice, P. E. Séguéla, B. Arnaudis, Y. Dulac, B. Lepage, and P. Acar, "Usefulness of three-dimensional transthoracic echocardiography for the classification of congenital bicuspid aortic valve in children," *Eur. Heart J. Cardiovasc. Imaging*, vol. 13, no. 12, pp. 1047–1052, 2012, doi: 10.1093/ehjci/jes089.

[5] K. Kusunose, A. Haga, M. Inoue, D. Fukuda, H. Yamada, and M. Sata, "Clinically feasible and accurate view classification of echocardiographic images using deep learning," *Biomolecules*, vol. 10, no. 5, pp. 1–8, 2020, doi: 10.3390/biom10050665.

[6] A. Madani, R. Arnaout, M. Mofrad, and R. Arnaout, "Fast and accurate view classification of echocardiograms using deep learning," *npj Digit. Med.*, vol. 1, no. 1, pp. 1–8, 2018, doi: 10.1038/s41746-017-0013-1.

[7] J. P. Howard *et al.*, "Improving ultrasound video classification: an evaluation of novel deep learning methods in echocardiography," *J. Med. Artif. Intell.*, vol. 3, pp. 4–4, 2020, doi: 10.21037/jmai.2019.10.03.

[8] M. H. bin A. Nizar, C. K. Chan, A. K. M. Yusof, A. Khalil, and K. W. Lai, "Detection of aortic valve from echocardiography in real-time using convolutional neural network," *2018 IEEE EMBS Conf. Biomed. Eng. Sci. IECBES 2018 - Proc.*, pp. 91–95, 2019, doi: 10.1109/IECBES.2018.08626735.

[9] Y. Gong *et al.*, "Fetal congenital heart disease echocardiogram screening based on dgacnn: Adversarial one-class classification combined with video transfer learning," *IEEE Trans. Med. Imaging*, vol. 39, no. 4, pp. 1206–1222, 2020, doi: 10.1109/TMI.2019.2946059.

[10] K. Seetharam, S. Raina, and P. P. Sengupta, "The Role of Artificial Intelligence in Echocardiography," *Curr. Cardiol. Rep.*, vol. 22, no. 9, pp.

1–8, 2020, doi: 10.1007/s11886-020-01329-7.

[11] S. Liu *et al.*, "Deep Learning in Medical Ultrasound Analysis: A Review," *Engineering*, vol. 5, no. 2, pp. 261–275, 2019, doi: 10.1016/j.eng.2018.11.020.

[12] J. Zhang *et al.*, "Fully automated echocardiogram interpretation in clinical practice: Feasibility and diagnostic accuracy," *Circulation*, vol. 138, no. 16, pp. 1623–1635, 2018, doi: 10.1161/CIRCULATIONAHA.118.034338.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[14] A. J. Weinhaus, "Anatomy of the human heart," *Handb. Card. Anatomy, Physiol. Devices, Third Ed.*, pp. 61–88, 2015, doi: 10.1007/978-3-319-19464-6_5.

[15] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," *2016 8th Int. Conf. Qual. Multimed. Exp. QoMEX 2016*, 2016, doi: 10.1109/QoMEX.2016.7498955.

[16] International Telecommunication Union Radiocommunication Sector (ITU-R), "Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect rations," *Recomm. ITU-R BT.601-7*, vol. 7, pp. 2–8, 2011, [Online]. Available: http://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.601-7-201103-I!!PDF-E.pdf.

[17] "FFmpeg." https://ffmpeg.org/.

[18] Dthpham, "Butterflow." 2019, [Online]. Available: https://github.com/dthpham/butterflow.

[19] G. Farneb, "Two-Frame Motion Estimation Based on," *Lect. Notes Comput. Sci.*, vol. 2749, no. 1, pp. 363–370, 2003, doi: 10.1007/3-540-45103-X_50.

[20] Z. Hussain, F. Gimenez, D. Yi, and D. Rubin, "Differential Data Augmentation Techniques for Medical Imaging Classification Tasks," *AMIA ... Annu. Symp. proceedings. AMIA Symp.*, vol. 2017, pp. 979–984, 2017.

[21] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5987–5995, 2017, doi: 10.1109/CVPR.2017.634.

[22] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning,"

31st AAAI Conf. Artif. Intell. AAAI 2017, pp. 4278–4284, 2017.

[23] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1800–1807, 2017, doi: 10.1109/CVPR.2017.195.

[24] E. Trivizakis *et al.*, "Extending 2-D Convolutional Neural Networks to 3-D for Advancing Deep Learning Cancer Classification with Application to MRI Liver Tumor Differentiation," *IEEE J. Biomed. Heal. Informatics*, vol. 23, no. 3, pp. 923–930, 2019, doi: 10.1109/JBHI.2018.2886276.

[25] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, 2020, doi: 10.1007/s11263-019-01228-7.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9351, pp. 234–241, 2015, doi: 10.1007/978-3-319-24574-4_28.

[27] A. A. Pravitasari *et al.*, "UNet-VGG16 with transfer learning for MRI-based brain tumor segmentation," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 18, no. 3, pp. 1310–1318, 2020, doi: 10.12928/TELKOMNIKA.v18i3.14753.

[28] T. Geng *et al.*, "FPDeep: Acceleration and Load Balancing of CNN Training on FPGA Clusters," *Proc. - 26th IEEE Int. Symp. Field-Programmable Cust. Comput. Mach. FCCM 2018*, pp. 81–84, 2018, doi: 10.1109/FCCM.2018.00021.

Appendix A 3D Architectures

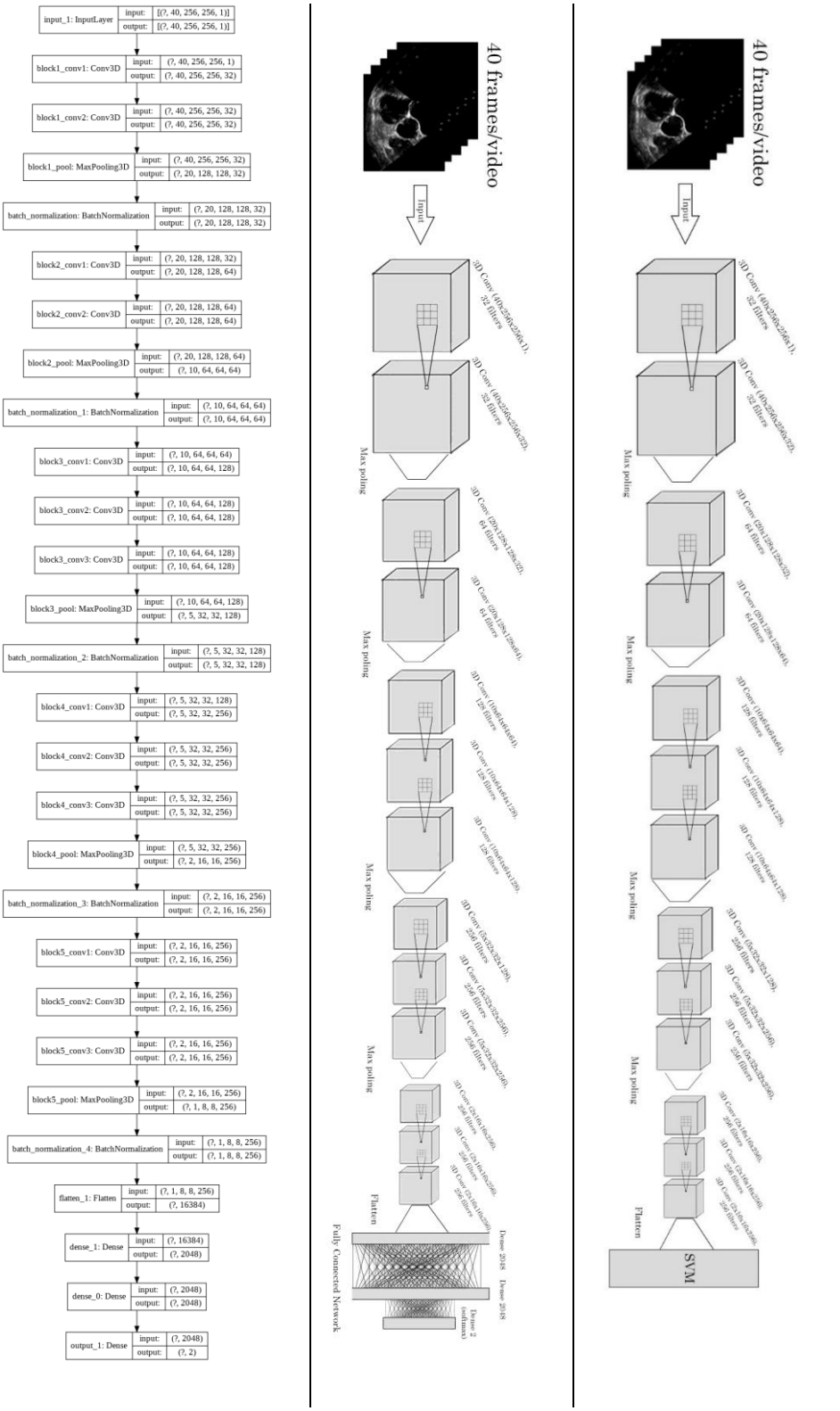

Figure 46 (Left) Detailed 3D architecture, (middle) Simple 3D architecture, (right) 3D architecture with SVM.