POLITECNICO DI TORINO

Master's Degree in Biomedical Engineering

Master's Degree Thesis

Integration of Deep Convolutional Networks and Active Shape Models for automatic prostate segmentation

Supervisors

Prof. Filippo Molinari

Ing. Massimo Salvi

Ing. Bruno De Santi

Candidate

Bianca Stefania Pop

March 2021

Abstract

Prostate cancer is the most common cancer in the world for what concerns the male population. Among the different tests to diagnose it, MRI proves to have the greatest accuracy in recognizing the prostate and tissue irregularities within it.
The diagnosis of a pathology affecting this organ is carried out using CAD systems that use automatic segmentation algorithms as a support to recognize at first the organ in its entirety or a region in which it is most likely to be found within the image.
The use of automatic algorithms for prostate segmentation allows to bypass the huge workload on technicians as the data available for processing is enormous and it needs rapid and less operator-dependent techniques. The latest approaches proposed use deep neural networks that rely on the large acquired capacity of computers in terms of GPU and CPU to automatically extract deep features from large datasets and to easily classify the input images on the basis of the repetition of recurring patterns. The anatomy of the prostate will be outlined focalizing on its characteristics in MRI images to better understand the major difficulties in its segmentation, such as the less defined edges at the base and the apex, the small effective area in the whole MRI volume, the appearance of the organ similar to the one of all the tissues surrounding it, the large variability in shape among different patients and other variable characteristics that depend on the scanning protocols and noise corrupting the acquired images.
We propose a fully automatic hybrid approach which involves an initial segmentation of the prostate volumes using a custom made deep 3D network, followed by a refining using an Active Shape Model.
The 3D convolutional neural network is used as it allows for greater spatial coherence, as opposed to 2D networks which are trained on single slices, losing the information deriving from the slices' connections within the volume.
The CNN alone, however, has some limitations, such as the low precision of the segmentation at the edges of the prostate and the possibility that the resulting shapes are not realistic due to the characteristics of the MRI images.
The active shape model is used as it is based on image characteristics different from those used by the neural network and by its nature, it should guarantee a more accurate segmentation even in correspondence with the organ contours.
In addition to these two models, a comparison with a 2D CNN model is proposed to

evaluate the differences and strengths of a model rather than the other.
Out proposed method has been tested on a set of 15 prostatic volumes in T2 MRI using DSC and 95% HD as performance metrics. We achieve a DSC of $84.5\% \pm 4.8\%$ and a 95% HD of 9.55 ± 3. Compared to the 3D CNN alone, the numerical results are improved and if a visual inspection is performed, it can be noted that the resulting shapes are closer to the real one. The use of a hybrid model allows to improve the performance of the CNN-based segmentation method by adding knowledge about the shape of the organ, even in those cases in which sufficient discriminative features for precise segmentation are not present. The proposed 2D network yields better results both for DSC and 95% HD, but this is only due to the fact that we have used a previously trained network and for this reason, more parameters are available for training. However, the 3D model demonstrates higher spatial coherence between neighboring slices, generating a three-dimensional output without holes and inconsistencies.
The performance of our method can be improved by optimizing the network parameters, the algorithm that allows to transform the output of the CNN into the points of the statistical model and the parameters of this latter.

Contents

Chapter 1

Introduction

1.1 Medical imaging

The term medical imaging refers to all the non-invasive techniques and processes involved in the creation of images and representations of the internal and external structure of the human body [19].
The final scope of these methodologies is to provide instruments to help specialists in diagnostic tasks and treatment of patients.

The inside of the body can be studied and classified into different organs, eliciting the anatomy of a healthy subject that can be useful in creating a database for detecting abnormalities.
It also offers the possibility to observe the physiology of a specific organ or tissue, that is the behaviour in time.
By observing the signals that can be obtained with specific methodologies, different properties of living tissues can be deduced with a specificity that depends on the instruments used in the acquisition phase, on the specialist experience and on the techniques available for post-processing.

From the traditional scans, 2-D static output are obtained on film. By combining successive scans a 3-D model can be produced and volume rendering techniques can be used to create 3-D images that the physician can use for later processing.

1.2 Quantitative imaging

The information generated by various medical imaging techniques can be utilized and explored in different ways.
The traditional clinical practice is known as qualitative imaging and it expects the radi-

ologists or other physicians with adequate training to visually inspect and interpret the images produced with the acquisition techniques.
Human eye recognizes patterns, but it is not optimized for complex quantitative measurements. For this reason many parameters contained in medical images are ignored in radiological reports, thus worsening clinical research.

Modern imaging techniques allow to obtain more objective results, which do not depend on visual inspection alone and therefore are more accurate in monitoring the progress of a disease and its response to certain treatments.
These techniques are called quantitative imaging and are useful in providing functional information.
The name 'quantitative' derives from the fact that they offer the possibility to extract quantifiable features from medical images, that can be used to assess the degree of change in the health status relative to a normal condition [26] [21].

Quantitative imaging includes different steps:

- Development of acquisition protocols.
- Standardization and optimization of protocols.
- Data analysis.
- Display methods.

The features obtained from imaging datasets using these methodologies should be more accurate than the traditional subjective methods that suffer from inter- and intra- observer variability.
The features that render the quantitative imaging a useful tool in modern radiology are:

- Accuracy: quantifies how close the measurement is to a correct answer.
- Precision: allows discriminating an error caused by the measurement from a biological change.
  Two important characteristics of the measurements must be the repeatability and the reproducibility, the measure must give the same value under the same conditions.
- Clinical validity: the results must have an impact, improving the outcomes.

In order to perform QI, the following tools should be taken into account.

Image Acquisition
Different acquisition modalities can be used to exploit quantitative imaging.

An important aspect to be considered is the possibility to obtain volumetric datasets that are easier to interpret in the real-world space and permit to obtain more accurate morphologies and quantitative characteristics of tissues.

We give a brief description of the main acquisition modalities used in this field.

1. Ultrasound Imaging
   Ultrasonography is a diagnostic imaging technique, also used for therapy. The purpose of this technique is to create an image of the internal structures of the human body, specifically of the soft tissues, to determine the presence or absence of irregularities or dysfunctions.
   It consists in sending pulses through the tissues at high frequencies using a probe and recording the echo that generates from them in the form of an image, whose values depend on the reflection properties of the different tissues, which affect the return time of the wave and the strength of the received signal.

2. Computed Tomography
   CT is a medical imaging technique that allows visualizing the internal structure of the human body without the necessity of using invasive technique such as cutting.
   It uses an X-ray generator that moves around the object that has to be scanned and detectors that collect the X-rays coming out of it.
   The data obtained with a scan are processed to obtain cross-sectional images that can be stacked together volume rendering algorithms to have a clearer view of the three-dimensional structures.

3. Magnetic Resonance Imaging
   MRI is a technique used to represent the anatomy of the human body and the physiological processes that take place within it.
   The MRI scanner creates a strong magnetic field around the patient's body at a resonance frequency that excites atoms which emit a radio frequency signal, measured by a receiving coil.
   The different tissues are discriminated according to the time it takes for the atoms to return to the equilibrium state. These times are determined by the relaxation processes that occur by setting specific parameters for the pulses and gradients that are applied.

4. Nuclear Medicine
   Nuclear medicine imaging is a technique used to record a radiation that is emitted from inside the human body.
   The process by which this occurs begins with the administration of radiopharmaceuticals to the patient. Detectors placed outside the human body, called gamma

cameras, capture the emitted radiation and create an image that allows recognizing the processes that take place inside the human body.

Quality Assurance

In order to respect the QI metrics of reproducibility and repeatability, it is important to have standardized protocols [26] for the acquisition techniques, in terms of both hardware and software components.

The instruments with which the acquisition is performed have an impact on the quantitative metrics that are further calculated; for this reason, calibration is an important step that has to be performed at the beginning of the procedure.

For what concerns MRI, for example, a crucial step is the coil selection, its positioning and the choice of the sequence parameters that has to be used in the different modalities.

Structure size definition

Evaluation tasks in quantitative imaging require the comparison of the results obtained with a specific algorithm with the real dimensions of the structures in exam.

The measure mostly used in these types of tasks is the volume, that can be estimated in different ways, more or less accurate, depending on the approximations used when assessing the shape.

A list of different software can be used to estimate the volume starting from the edges of structures in the image. Many of these algorithms have been automated during the last years,bringing the image segmentation to a new level of precision.

The volume of a structure can discriminate between normal and pathological conditions, but it is not the only measure used for this scope.

The distribution of voxel values within the volume can determine a huge number of features that reflect particular behaviors of the tissue, thus the texture of an image and the grey levels that can be obtained using simply algorithms could be beneficial in diagnosis purposes.

Quantitative imaging is beneficial for patient care as it extrapolates information that goes beyond visual aspects [21].

This interpretation, however, requires additional processing and an adequate education of the clinicians which can be lacking in case of insufficient experience with the specific tool.

1.3 Automatic Algorithms

As previously discussed, the volume estimation of structures in medical images is an important step in diagnosis. Many image segmentation algorithms have been proposed,

starting from manual segmentation methods, passing through semi-automatic and reaching the techniques mainly used these days, based on a fully automatic approach.
The main reason that has made this automation necessary is that across the years a lot of medical data have been produced, with huge dimensions compared to data available in the past.
To manage this problem and get useful information for the intended purposes, computers have been largely used to facilitate processing analysis. Anatomical regions of interest can be delineated in an automated way, facilitating the workload of the physician, thus leading to a faster diagnosis, a more precise localization of a pathology or lesion as in the case of computer-integrated surgery and a possible more accurate treatment planning.

Methods for image segmentation

The term segmentation refers to the recognition in an image of specific regions consisting of pixels with similar characteristics, representing a homogeneous area in terms of intensity, texture or other correlated image features.
The segmentation can be performed both on 2D or 3D images, the latter incorporating more spatial information, and for this reason being more computationally complex and heavier.

We present a list of the main techniques used for this scope, many of which can be used in conjunction for solving different problems.
A first coarse distinction separates the segmentation methods into 8 categories [3], some of which are not completely automated but can be used inside automatic algorithms to obtain a starting point for other techniques or to refine the segmentation in the middle or at the final step.

1. Thresholding
   It is based on a binary partitioning of the image intensities.
   The threshold is defined as the value corresponding to valleys in the intensities histogram. All pixels with intensity above the threshold are grouped together into one class, and all the other pixels into another class.
   It works well and it is simple when the intensities or other features used to make the distinction have separable values. It can not be applied to multi-channel images and it does not provide spatial characteristics; it is sensitive to noise and inhomogeneities in the pixel intensities that corrupt the histogram.

2. Region Growing
   It consists in manually placing a seed point on the object which has to be segmented, from which all connected pixels are extracted, on the basis of a predefined criterium (such as intensity information). It is sensitive to noise that can mislead

the connectivity between adjacent pixels.

3. Classifiers
   This category gathers methods based on pattern recognition for which the segmentation is derived from data used for training having known labels.
   Classifiers are non-iterative methods, efficient and not time consuming; they can be applied to multi-channel images. They do not provide spatial modelling. Training sets are manually obtained (time consuming).

4. Clustering
   It permits to segment the image without the need for training labelled data. The pixels in an image are partitioned based on particular properties learned from the image itself.
   Each class is defined as a set of pixels having similar properties. It is a fast algorithm, but initial segmentation and parameters are required (operator-based). It does not provide spatial modelling, it is sensitive to noise and intensity inhomogeneities.

5. Artificial Neural Networks
   ANNs are networks made of connected processing units that singularly perform simple computations.
   The features of the input image are learnt passing through the network and adapting the weights assigned to the connection between the units (or nodes). Unlike the other techniques, it provides spatial information.

6. Deformable models
   These models are based on closed parametric curves or surfaces used to define the boundaries of a structure inside the image. These curves deform under internal and external forces that are respectively based on the characteristics of the curve itself and on features computed on the image such as the intensity gradients.
   The adaption of these curves to reach the object is called relaxation and it is an iterative process. The technique is robust to noise and not well defined edges. It requires manual interaction for the initial model and parameters.

7. Atlas-Guided Approaches
   They provide an atlas with information on the anatomy of the region which requires segmenting. The atlas is used as a reference frame for segmenting new images.

8. Markov Random Field Models
   They are statistical models to represent interaction between neighboring pixels.
   The scope of these models is not exactly the segmentation of objects in an image, but they can be used within segmentation algorithms to cope with the problem of separated regions belonging to the same object. They are robust to noise and can

be used to model both segmentation classes and intensity inhomogeneities. They require the selection of the spatial interactions parameters and can be computationally heavy.

Many of the algorithms presented above are not used alone because they require manual interaction, which slows down the process.
For this reason, machine learning has been introduced in the medical field, to fully automate the procedure and to ensure high precision and repeatability.
The first operation that a machine learning algorithm performs is the recognition of image features that are thought to be important for the specific purpose [18].
The best combination of these features is chosen and from these, a new image can be classified or segmented.
Ordinarily, an ML classifier in the computer vision field follows these steps:

1. Structures of interest in an image are segmented with a traditional segmentation technique such as thresholding, active contour etc.

2. The segmentation obtained in the previous step is processed in order to extract particular features (e.g. contrast, shape etc.) with a feature extractor.

3. Sets of features are input to an ML classifier together with known class labels for training.

4. The trained model determines the belonging class for examples that have not yet been seen from the algorithm.

The idea behind this revolutionary technique, is that computers are taught to solve problems by learning from experience in the form of training data, and are optimized using specific algorithms, to output accurate predictions also on unseen elements.
The ability to use their experience on new data is called generalization and it is first tested in the training phase on an set made of examples not used for training called validation set. The results of the predictions obtained from the validation set, are used to further tune the model, in order to finally evaluate it in the final phase, on a test set made of completely new data.

The model can use the input data in different ways during training.
The three main types of ML algorithms are presented below [18], each representing a specific type of approach regarding the use of input data.

- Reinforcement learning
  The first 'stage' of this classifier is built using labelled data. In a second phase, unlabelled data is given to the classifier, which tries to better characterize it in

order to improve the classification.
The model learns from error while optimizing the objective function.

- Unsupervised learning
  The algorithm is trained with unlabelled data, from which it has to recognize specific patterns that repeat into the image, defining the characteristics of different classes. Examples of this type of algorithms are K-means, mean shift, hierarchical clustering, MRF, ISODATA, fuzzy c-means.

- Supervised learning
  The algorithm is fed with annotated data and tries to define some characteristics of the pixels belonging to the labelled region, in order to create accurate predictions on new, previously unseen datasets.
  Example of this type of algorithms are SVM, decision trees, linear regression, logistic regression, naive Bayes, k-Nearest Neighbor, Random Forest, Neural Networks.

Some of the techniques cited above, have already been discussed when introducing the main types of segmentation methods.
In the following we focus on the latest solutions and deepen the neural networks, which at the moment are the state of the art in medical image segmentation.

1. Support Vector Machine (SVM)
   It transforms input data to produce a plane or support vector that separates the two classes.
   This wide plane can separate the points of observation at different degrees, by setting specific parameters the plane can be optimized for the best classification. This algorithm provides basis functions to map a certain distribution of points to other dimensions with nonlinear relationships, in order to render the classification among not linearly separable possible (figure 1.1).

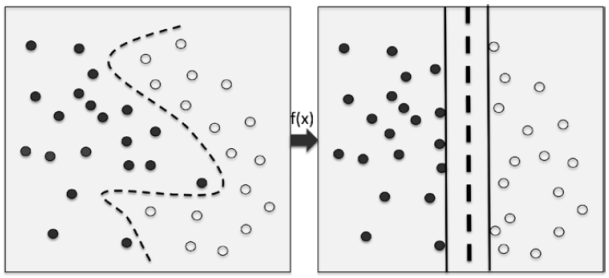


Figure 1.1: Example of mapping from the original space in which data are not linearly separable to a new dimension where classification can be performed using a hyperplane. [18]

2. K-NN

   Objects are represented as position vectors in a multi-dimensional space.

   An input vector made of the chosen features for a single example, is classified into the most common class among the k-nearest neighbors (figure 1.2).

   In order to determine how close an example is to another a similarity measure is used, typically the Euclidean Distance.

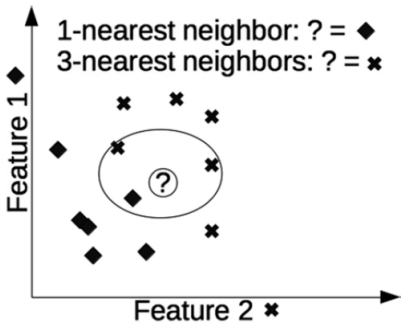


Figure 1.2: The unlabelled example in the image can be classified into two different classes depending on the choice for k. [18]

3. Decision Trees

   Among all the machine learning methods, they are the only one that guarantee a human-readable result on how to classify a given example. The algorithm is based on the search for the best number of decision points that produce a relatively simple decision tree, ensuring the most accurate result possible.

4. Naive Bayes algorithm

   It states that the probability of an event is a function of related events.

   In machine learning-related classification problems, the final probability that an element belongs to a given class is given by the probability chain of all the input features together. All the features are independent of each other, which does not allow this algorithm to faithfully model reality, but it guarantees good acquiring of estimates of performance.

5. Neural Networks

   As previously described, neural networks are a connected set of logical units that process information in order to output a prediction. The output of the network is compared with the given label and an error function (loss function) is computed. This error can be reduced changing the weights of the network, based on the computation of a search function, which defines the direction and magnitude of the change required to reduce the error (iterative process).

Chapter 2

Background and related works

2.1 Introduction to deep neural networks

2.1.1 Deep learning overview

In computer vision, an important aspect to be taken into account is the automation of the algorithm, that permits to obtain results that do not rely on expert crafted features. Traditional machine learning algorithms, as previously discussed, make use of features that sometimes are manually selected by the operator, that render the procedure less reliable, due to the high variation of data among different subjects [10].

Deep learning has been introduced in computer vision field to cope with this problem. Features computed from objects segmented with traditional methods are no more used as input information to discriminate structures in the image. They are substituted by pixel values, providing a technique that does not depend on the operator.
Deep learning avoids possible errors that depend on the feature calculation and selection, offering more accurate results relative to common classifiers. It is also called end-to-end machine learning, because the process on which it is based maps from raw images directly to the final classification.

Why deep learning has advanced so far these last years?

As we can observe in figure 2.1 , during the last decades computational power has grown and continues to grow both in terms of CPU and GPU following a certain distribution.
The general distribution of both CPU and GPU power can theoretically be described by the Moore's law, which states that the number of transistors in a dense integrated circuit doubles about every two years.
A more recent law, proposed by J. Huang, questions Moore's law, arguing that it is valid only for the CPU, while improvements in the GPU occur at a much higher rate.

This theory has been scientifically proved. A synergy between hardware, software and artificial intelligence has made it possible, giving rise to new technologies and components capable of processing large amounts of data in reasonable time [17].

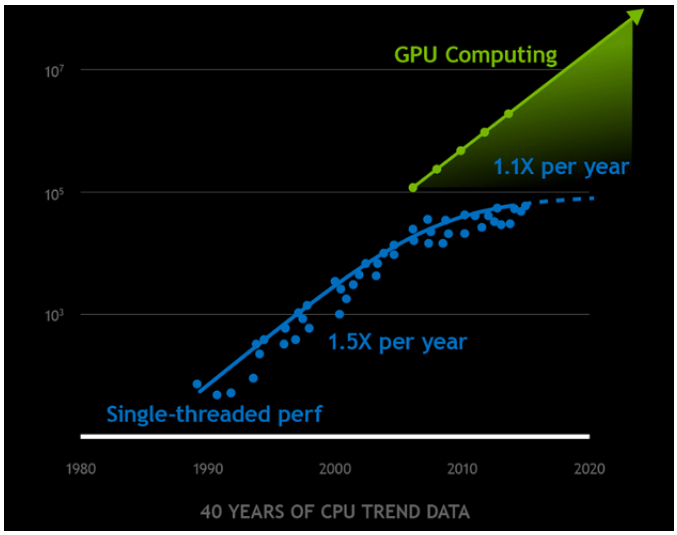


Figure 2.1: Comparison between Moore's law and Huang's law predictions on CPU and GPU growth. According to the Huang's law, silicon chips will double their performance every two year, at a rate much higher respect to CPU growth. [17]

The idea at the base of deep learning is to enlarge the basic architecture of neural networks, adding a variable number of hidden neurons and layers that further process the raw information extracting high-level features and semantic interpretation from the input data.
These architectures rely on the possibility to transfer algebraic computations and convolutions to the GPU [24][20], obtaining fast results even with large datasets, compared to the native architectures of ANNs.

Artificial Neural Networks' main components

A generic artificial neural network [10] is an ensemble of interconnected nodes called artificial neurons, that process the information that is input at the first layer by means of non-linear functions of the sum of the inputs at each node. These connections have weights that are adjusted while the learning proceeds and that determine the strength of the specific connection. Neurons are organized in layers, each of which performs a specific transformation.
Training of the network is performed by passing labelled examples through the network and computing the difference between the processed output and the desired output (label). This difference corresponds to the error that is used to correct the weights of the connections in order to make the predicted value more similar to the desired one.

The first and simplest feedforward neural network is based on perceptron [10], an algorithm for supervised learning of a binary classifier called a threshold function that maps an input to an output value.
A single-layer perceptron (figure 2.2) can be seen as an artificial neuron that uses a linear activation function to transform an input vector (or multiple input vectors) into an output prevision.

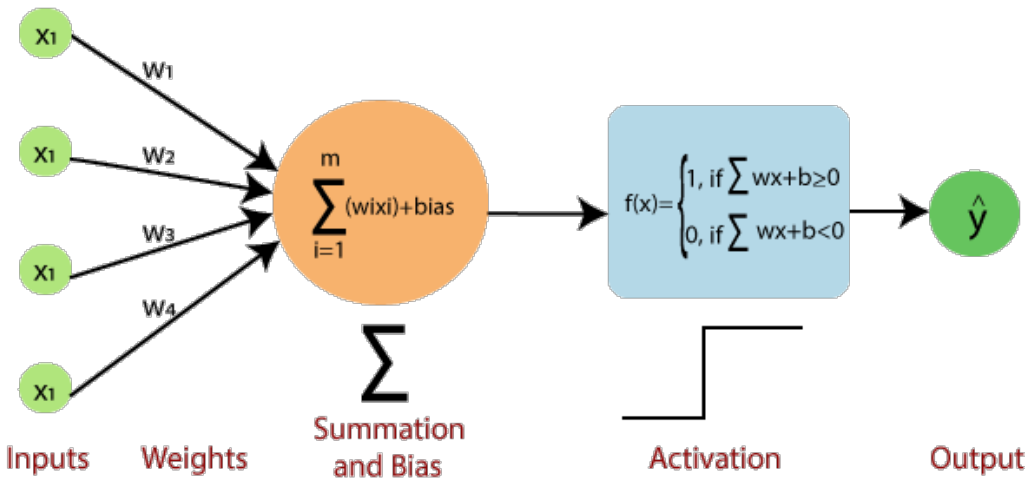


Figure 2.2: Single-layer perceptron architecture: inputs are weighted and passed into a node that performs a summation and adds a bias term. The output from this node is then passed through an activation function to obtain the final output.

The function ($f(x)$) that transforms the linear combination of input values into the output value for the single-layer case is called 'Heaviside Step Function' and has the following form:

$$f(x) = \begin{cases} 1 \text{ if } w \cdot x + b > 0 \\ 0 \text{ otherwise} \end{cases}$$

where:

- w is a vector representing real-valued weights.
- $w \cdot x$ is the dot product $\sum_{i=1}^{m} w_i x_i$ calculated over m inputs.
- b is the bias term, used to shift the decision boundary from the origin.

As we can observe, this value can be 0 or 1 whether x represents a positive or a negative instance (if the classification is between two classes).
The algorithm on which the single-layer perceptron is based, converges only if the two classes are linearly separable.

Modern Neural Networks

The multilayer perceptron (figure 2.3) is a class of feedforward ANNs of at least three layers of nodes (input, hidden and output layer), introduced to deal with data that can not be handled with the previous model.
The multilayer perceptron can also be referred as a 'neural network' and it represents the basic model from which many architectures have been developed to handle different types of problems.

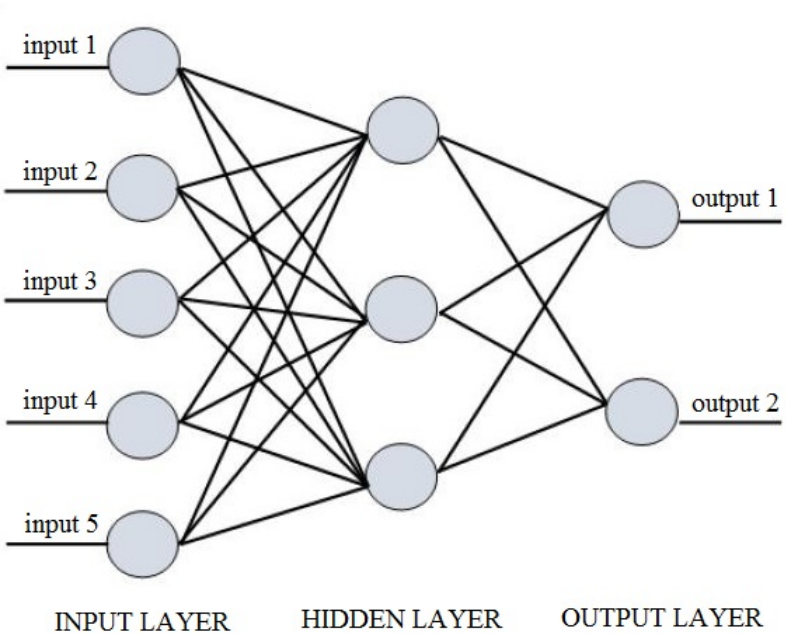


Figure 2.3: Multilayer perceptron architecture: despite the single-layer perceptron, inputs are passed into a variable number of hidden units in the hidden layer that process non-linear activation functions to compute the final output.

From the hidden layer on, some neurons in this architecture can process a non-linear activation function, which allows dealing also with non-linear data distributions.
These non-linear activation functions are represented in figure 2.4 with the corresponding formulas.

1. Logistic activation function (sigmoid)
   It exists in the range [0 1] and it describes the probability of an output value given an input. Values out of the range are saturated to 0 and 1 respectively.

2. Hyperbolic tangent activation function
   Similar to the sigmoid, but zero-centered (range [-1 1]).

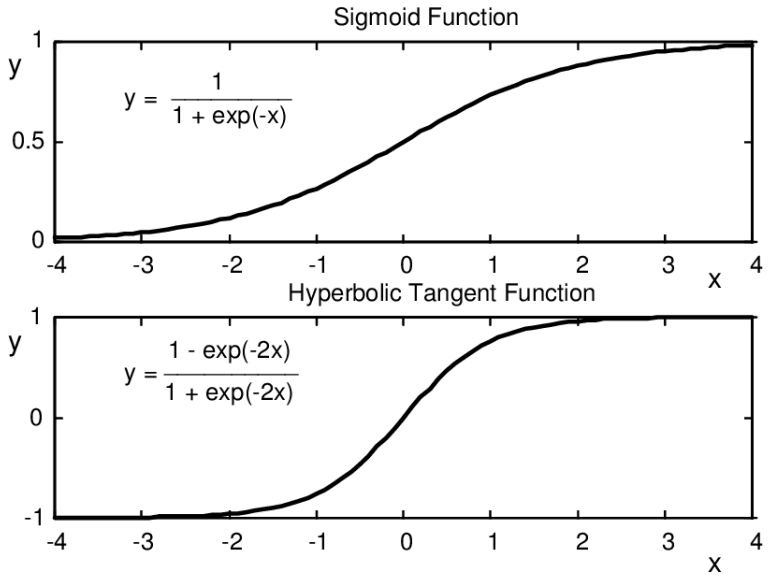


Figure 2.4: Comparison between sigmoid and hyperbolic tangent activation functions

Network training

The supervised learning technique used by the multilayer perceptron for training is called backpropagation and consists in adjusting the weights of the network based on corrections that minimize the error in the entire output using a technique called gradient descent.

In order to evaluate how well the algorithm learns to model the training data, it is necessary to define an objective function that has to be optimized.
In these applications, this function must be minimized, so it is called loss function and it computes the magnitude of the difference between the output predicted by the network and the input data.
The loss error refers to a single training example; when computing the error for the entire training set, we must define a cost function, which corresponds to the average of the loss function over the whole input data.
Many types of loss functions exist and can be used to train a neural network. A first rough classification allows to divide these functions into two main categories, namely the classification loss functions, which refer to discrete outputs, and regression loss functions, dealing instead with continuous value outputs.
Each time a different set of weights is tested during the training process, the value of the cost function is calculated and minimized using gradient descent.
To understand the behavior of this algorithm, suppose we define a cost function with a quadratic behavior [12], known as mean squared error (MSE) and expressed as follows:

$$C(w,b) \equiv \frac{1}{2n} \sum_{x} ||y(x) - a||^2$$

where:

- w: corresponds to the set of weights in the network.
- b: all the bias terms used in the network.
- n: the total training inputs.
- a: the outputs of the network given x as input.

More generally, we can define any $C(v)$ function that depends on an arbitrary number of variables, for example v_1 and v_2.
The derivative of this function with respect to these two variables is then:

$$\Delta C \approx \frac{\partial C}{\partial v_1}\Delta v_1 + \frac{\partial C}{\partial v_1}\Delta v_2$$

where the vector of partial derivatives is called the gradient of C and relates changes in v to changes in c, and can be expressed as:

$$\nabla C \equiv \left(\frac{\partial C}{\partial v_1}, \frac{\partial C}{\partial v_2}\right)^T$$

We can express the derivative of C as a function of the gradient:

$$\Delta C \approx \nabla C \cdot \Delta v$$

We can choose:

$$\Delta v = -\eta \nabla C$$

with small and positive η, in such a way as to have $\Delta C \leq 0$, or rather the derivative of the cost function always decreasing.
The value of v will be iteratively updated until the global minimum point is reached. In essence, the gradient descent algorithm repeatedly calculates the gradient of the function C and then shifts the values in the opposite direction.
This rule can also be applied if there are many more variables on which the function to be minimized depends.
If we substitute the values of v with the weights and biases terms, we have to do the following updates at each step:

$$w_k \to w'_k = w_k - \eta \frac{\partial C}{\partial w_k}$$

$$b_l \to b'_l = b_l - \eta \frac{\partial C}{\partial b_l}$$

Backpropagation is the algorithm used to compute the gradient of the loss function with respect to the weights and bias terms in the network. Is is called this way because the error vectors are calculated starting from the final layer.
The complete algorithm involves the following steps:

1. Input
   Set the activation a^1 corresponding to the input layer.

2. Feedforward
   For each layer, starting from the second to the last one, compute

$$z^l = w^l a^{l-1} + b^l$$

$$a^l = \sigma(z^l)$$

3. Output error
   It is the error corresponding to the last layer, obtained as:

$$\delta^L = \nabla_a C \odot \sigma'(z^L)$$

   where

   - $\nabla_a C$ represents a vector whose components are the partial derivatives of C with respect to the activation of neurons in the different layers.
   - $\odot$ is the elementwise product of the two vectors (Hadamard product).
   - $\sigma'(z^L)$ represents the rate at which the activation function changes at z^L with z^L being the weighted input to the neurons in layer L.

4. Backpropagate the error
   Starting from $l = L - 1$ compute the error related to each layer, until $l = 2$.

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$$

   where

   - $(w^{l+1})^T$ represents the transpose of the weight matrix in the $(l+1)^{th}$ layer.
   - δ^{l+1} is the error previously computed, relative to a layer subsequent to the one under consideration.

5. Output
   Obtain the gradient of the cost function with respect to the weights and the biases as

$$\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j$$

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l$$

Problem of overfitting

When the number of weights and biases of the network is too high, it can happen that the network learns patterns in the data that do not refer to real values, but to noise and random fluctuations.
This phenomenon is known as overfitting and it leads the network to model train data almost perfectly, but to fail when new data is presented to it, even if belonging to the same domain as the training data.
The performance of the network during the validation and inference phases deteriorates because random noise is different in each example presented to the network and the true function describing the data distribution can not be recognized during learning.

Two simple ways of reducing the effect of overfitting can be increasing the amount of training data or reducing the complexity of the network.
These two possibilities, however, are sometimes impractical, because obtaining new data can be difficult and decreasing the size of the network can lead to lower performance.
There are methods, called regularization techniques, whose purpose is to reduce overfitting, keeping constant the size of the network and the number of examples presented during training.
A regularization technique [14] foresees that terms are added to the loss function in order to obtain an effect on the overall training trend.

There are several types of regularization, they differ according to the target to which they are applied, to which function is used, to the hyperparameters introduced to scale the degree of regularization on the basis of the problem being faced.
We present below three of the main techniques, with a brief explanation of how they work.

1. L2 Regularization - Weight Decay
   It consists in adding a regularization term to the cost function, which corresponds to the sum of the squares of the weights of the whole network.
   For clarity, the regularized version of the cost function mentioned above becomes:

   $$C = \frac{1}{2n}\sum_x ||y - a^L||^2 + \frac{\lambda}{2n}\sum_w w^2$$

   where the regularization parameter λ is scaled by the size of the training set.
   With adding this term the network is more pushed to learn small weights (if high *lambda*) or simply to minimize the original cost function (if low λ).

2. L1 Regularization
   The term added to the cost function is the sum of the absolute values of the network weights, again scaled by λ.

$$C = \frac{1}{2n}\sum_{x}||y - a^L||^2 + \frac{\lambda}{n}\sum_{w}\|w|$$

   This technique allows shrinking the weights much more, keeping only the most important connections in the network and deleting all the others.

3. Dropout
   It is different from the two previous techniques, as it does not act on the loss function, but directly modifies the network, randomly and temporarily choosing neurons that will not be taken into account during training, being restored only at the end of the process with their original weights.
   The procedure is repeated, deleting a different subset of neurons each time.
   The overall effect that this technique has on overfitting is to mediate the effect of different networks, trying to eliminate the effects deriving from noise.

Deep architectures

Deep Neural Networks, as anticipated in the introduction to the chapter, are an extension of basic Artificial Neural Networks.
Figure 2.5 shows the architecture of a typical Deep Neural Network. As we can observe, it is made of the same components of a shallow NN, but it has multiple hidden layers between the input and the output.

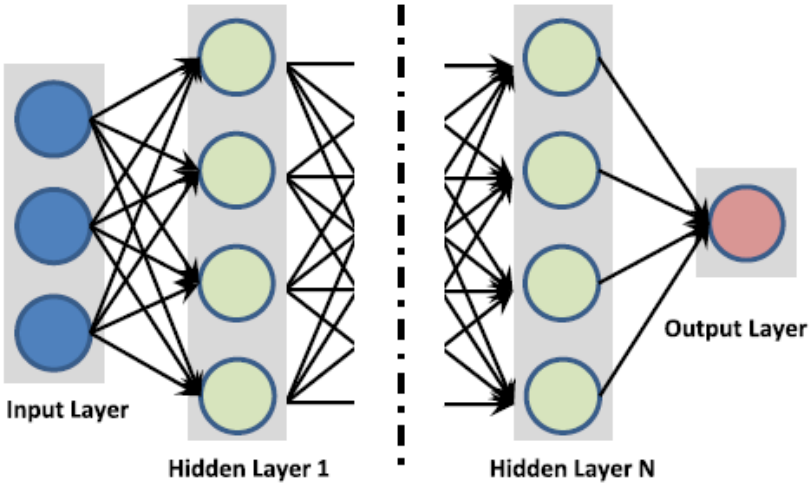


Figure 2.5: Generic Deep Neural Network architecture made of many hidden layers that process the input information to obtain the final prediction. [20]

To cope with the high computational effort that training these deeper networks requires, two additional activation functions have gained popularity:

- Rectified Linear Unit (ReLU)
  This function performs the *'max'* operator between 0 and the input. If the output of the transformation is less than zero, the neuron will be deactivated. This allows to activate neurons in different instants of time, rendering the process of learning faster respect to the one using the sigmoid or the tanh function.

- Leaky ReLU
  It is an improved version of the ReLU function which solves the problem of the null gradient for $x < 0$ (that would deactivate all neurons in that region) by making the negative component of x a small linear function of the positive one.

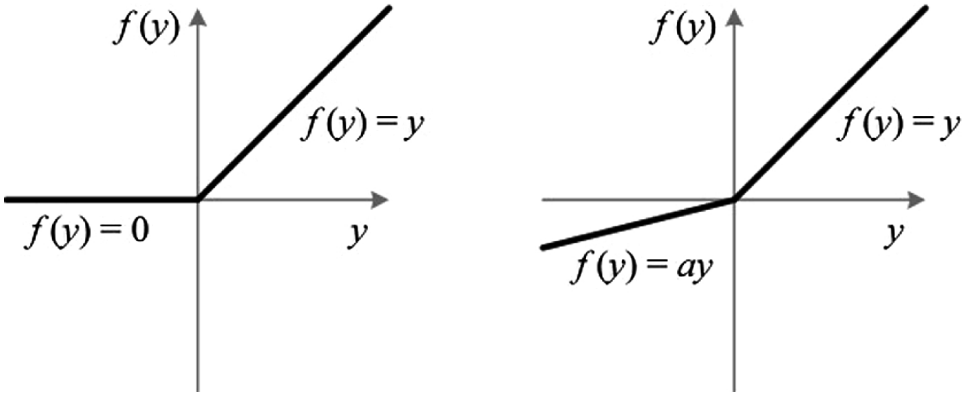


Figure 2.6: Difference between the ReLU (left) and Leaky ReLU (right) activation functions.

Many types of Deep Neural Networks exist, depending on the task they have to achieve [20]. Deep Autoencoders, for example, do not deal with the classification of the input examples, but simply with their recreation in the output. Other networks model sequence data (RNNs) or probabilistic relationships between the variables involved (RBMs).

In this study we will deal with image data and for this reason we will use a particular type of networks called Convolutional Neural Networks, specific for this type of problem. Instead of using predefined kernels, a CNN employs convolution filters to perform the convolution operation multiple times on the input image to learn locally connected neurons as they were hypothetical kernels created on the data.

The architecture of a basic CNN is shown in figure 2.7; the network details, along with an in-depth description of all the parts constituting the model are presented in the following paragraph.

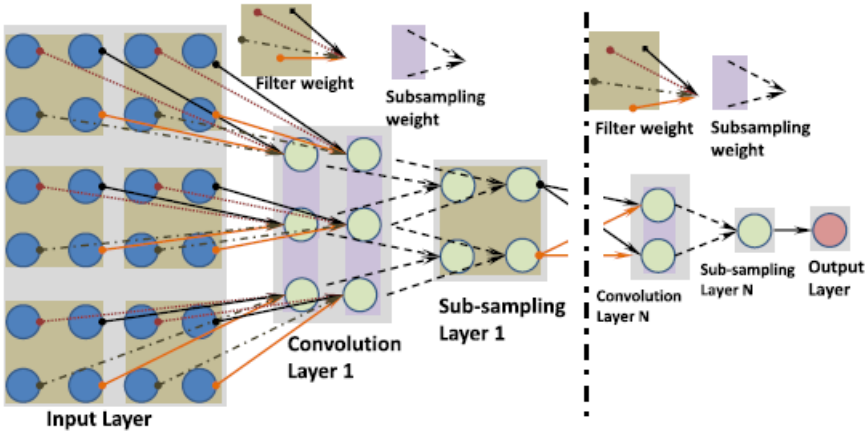


Figure 2.7: Convolutional Neural Network architecture: hidden filters are used to transform input data into 3D output probability maps that represent neurons' activations. [20]

2.1.2 Convolutional Neural Networks

Introduction

The idea behind convolutional networks is to revisit the architecture of a standard network (such as the multilayer perceptron) by removing the fully connected layers, in order to mitigate the high computation and the overfitting effect that derives from them [20]. The resulting effect can be considered a sort of regularization, by means of which complex data patterns are mixed together and resized, guaranteeing a hierarchical view of the different patterns present in the image.

The spatial structure of the data is important in those tasks specialized in image recognition.
Traditional artificial neural networks are not optimized to handle image recognition tasks as they do not take into account the spatial structure of the data.
A CNN network is inspired by the biological visual process [28], in the sense that neurons are organized within the network in such a way that each of them responds to a stimulus coming from a restricted region, called the receptive field, just like in the visual cortex.
The entire visual field is given by the partial overlap of the receptive fields of different neurons.
The spatial hierarchies of the features are learned from the network in an adaptive way, starting from low resolutions, up to high level patterns.
Initially the network creates a representation of small parts of the input image and subsequently, starting from these, it recreates a representation or larger areas.

In the following, the functioning of each different block within a CNN will be explained in detail, making the potential of this method clearer.

Architecture and main features

The network architecture is based on a stacking of consecutive layers of different types which process the input image, extract particular features and based on these, create an output image [10][28][22][15][9].
The first two types of layers described below, together constitute the part of the network that deals with the feature extraction process, while the third type deals with the final classification [9].

- Convolutional Layer
  This layer allows to extract a map of features from the input image.
  The process through which this transformation takes place, makes use of particular learnable filters called kernels which are convolved with the image (tensor of values).

  The convolution operation is a type of linear operation in which a filter (matrix of numbers) is multiplied by means of a dot product (element wise multiplication) by a region of the input image, the receptive field, and is run along the entire height and width of the image, and extended across the entire depth.

  The filter is composed of a vector of weights and bias terms which are shared by all neurons within the receptive field.
  For clarity, an example of convolution is presented in figure 2.8.

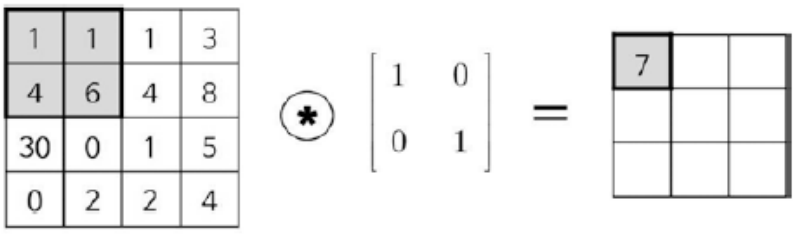


Figure 2.8: Example of convolution operation of a 4 x 4 pixel image with a 2 x 2 filter. The first element in the resulting matrix is obtained convolving the first submatrix of dimensions equal to the ones of the filter by the filter itself. [9]

  As can be observed, the kernel output in a given position of the image corresponds to the sum of the tensor values convolved with the filter inside the receptive field. By applying the convolution to different regions of the image, it is possible to obtain the value of a pixel and of those surrounding it.
  Different filters can be applied to the image to extract different features, whereby a

particular kernel can be said to correspond to a particular feature extractor.

The size of the output is determined by three hyperparameters:

1. Depth: number of neurons in a layer that connect to the same region in the input volume.
2. Stride: number of units of which the filter is moved along the image (example in figure 2.9 with $stride = 1$)
3. Zero padding: performed along the edges of the image (example shown in figure 2.10).

Neurons belonging to the same slice use the same weights and bias, decreasing the number of parameters the network has to learn.

During the training process, the network learns filters that are activated when they detect specific characteristics of the image at specific points.
The activation maps that are output from all filters are stored together after being stacked along the depth dimension to create the output volume from the convolution layer.

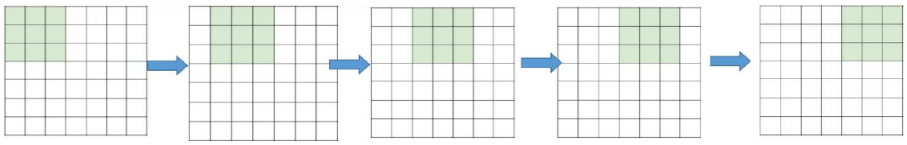


Figure 2.9: Example of the influence of the stride hyperparameter during the convolution operation. In this case the stride has been set to 1, meaning that after each convolution, the filter is moved along the image of 1 pixel at a time. [15]

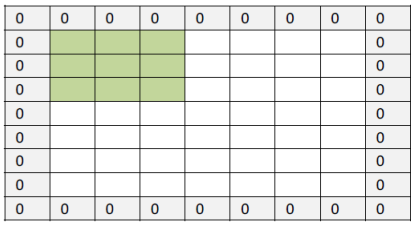


Figure 2.10: Zero padding operation performed on the image by adding zeros on the edges [15].)

- Pooling Layer
  The purpose of this layer is to reduce the size of the volume by combining together the outputs of a specific region of the image into a single neuron in the following layer.

A set of non-overlapping windows is passed over the image and a value is calculated for each of them. The size of the output volume depends on the size of the patch chosen to select the neighboring pixels.

The pooling operation can be assimilated to a non-linear downsampling and can be of various types. The main types used in these applications are shown in figure 2.11.

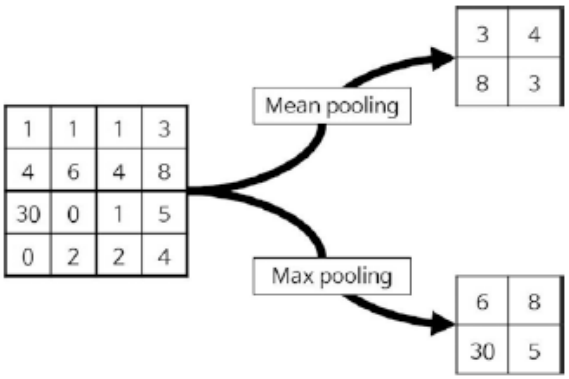


Figure 2.11: Example of pooling operation with resulting matrices for mean and max cases.

1. Max-pooling: outputs the maximum value within the window.
2. Average-pooling (Mean pooling): outputs the average value within the window.

By resizing the volume, the number of parameters decreases and consequently also the memory and computation required to train the network.
Having few parameters also contribute to mitigate the problem of overfitting.

- Fully Connected Layer
  This layer represents the part of the network that deals with the classification.
  The feature maps extracted using the other two types of layers are transformed into a one-dimensional array and connected to one or more fully connected layers that produce the final output, which generally represents the probability for an input to belong to a given class.
  The nodes that make up the final fully connected layer are as many as the classes to which the volume in question can belong.
  The classification is carried out passing the output of the final fully connected layer into a non-linear activation function (such as the ReLU function).

The training process for a CNN consists in identifying kernels and weights, respectively in the convolution and in the fully connected layers, which allow to minimize the error

between the final output of the network and the ground truth relative to the training set data.

Kernels are learned automatically during the training process, while other parameters, such as the hyperparameters, must be set before starting the training. Those hyperparameters can be for example:

- Number of filters.
- Shape and size of filters.
- Padding.
- Stride.
- Shape of max-pooling.
- Learning rate for back-propagation.

The performance using particular kernels and weights is evaluated using a specific loss function and implementing forward propagation.
Subsequently the learnable parameters are updated through backpropagation and gradient descent.

We have introduced in the previous section the problem of overfitting, which occurs when the network memorizes and learns to recognize noise in the image, becoming less accurate when evaluating data not belonging to the training set.
One way to recognize the overfitting is to monitor the trend of the loss function and of the accuracy both on the training set and on a validation set, which is therefore necessary to fine-tune the hyperparameters.
The methods used by CNNs to reduce the effects of overfitting are:

- Obtain more data available for training.
- Regularization techniques such as dropout or weight decay.
- Batch normalization that consists in adding a layer that normalizes the input of the subsequent layer in order to improve the gradient flow through the network.
- Data augmentation with random transformations (rotation, flipping, translation) to make the network recognize objects that could potentially be in different positions or angles in different images.
- Early stopping: stop training before overfitting.
- Decrease the complexity of the network by limiting its depth or number of hidden units to limit the number of learnable parameters.

2.2 Active shape models

Active shape and appearance models have been developed to face the need to model structures within the human body that have a great variability in terms of size, shape and appearance.
In order to automatize the segmenting task and obtain acceptable results, the model must predict these variations and modify its characteristics on the basis of statistical estimates made on a set of images called a train set.
The models in question are made up of flexible sets of labelled points representing an object, from which the statistics of neighboring points on a certain number of training shapes are analyzed using the Point Distribution Model (PDM) [23][2].
The PDM represents the modes of variation of the shape of a structure,having the average shape calculated on the training set as initial model.
From this average shape, the position, orientation, scale and shape parameters are adjusted to obtain a better fit of the image. Limits are placed on the movements that every point can make, to maintain the shape that can be obtained plausible (Global Shape Constraints).

2.2.1 Traditional approach

Point Distribution Model
PDM is used to represent a class of shapes, modeling it with a parametrization, starting from a set of points that are usually placed on the edges of the structure under analysis or on significant internal points of it [23].

The steps required to develop a model and use it to predict new instances are as follows:

1. Obtaining the training data
   As previously said, the shape of an object is described by a series of points, each of which has a label that represents a specific part of the object or its boundary, so it must be placed in the same way on each example belonging to the training set.
   If the point are not placed accurately on each individual example, the different shapes cannot be aligned and the model will not be able to correctly represent the position of each point. As a result, there will be noise terms that will corrupt the resulting shape.

2. Aligning the set of training shapes
   In order to compare equivalent points belonging to different objects, these must be aligned in the same way with respect to a set of axes that can be defined on the

basis of an object (belonging to the training set) taken as reference. The alignment is performed through scaling, rotations and translations of the original shape.

3. Capturing the statistics of a set of aligned shapes
   Once the shapes have been aligned, the mean shape and how much each individual example differs from it can be computed as follows.
   The n points of the i^{th} shape of the set are represented with a vector x_i:

$$x_i = (x_{i0}, y_{i0}, x_{i1}, y_{i1}, ..., x_{ik}, y_{ik}, ..., x_{in-1}, y_{in-1})^T$$

   where (x_{ij}, y_{ij}) is the j_{th} point of the i_{th} shape.
   The mean shape can be obtained as:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

   The deviation from the mean is calculated for each example as:

$$dx_i = x_i - \bar{x}$$

   PCA (principal component analysis) is applied to each deviation to obtain the modes of variation.
   First, the covariance matrix S of dimensions $2n$ x $2n$ is calculated:

$$S = \frac{1}{N} \sum_{i=1}^{N} dx_i dx_j^T$$

   The unit eigenvectors p_k $(k = 1...2n)$ of S describe the modes of variation of the points, respecting:

$$Sp_k = \lambda_k p_k$$

   where λ_k is the k^{th} eigenvalue of S
   and

$$p_k^T p_k = 1$$

   A large part of the variation can be explained by a small number of modes, $t(< 2n)$; usually the eigenvectors corresponding to the largest eigenvalues describe the most significant modes of variation. The value t can be found choosing the smallest number of modes for which the sum of their variances explain a sufficient large

proportion of λ_T, which is the total variance of all the variables, and

$$\lambda_T = \sum_{k=1}^{2n} \lambda_k$$

Each point l moves along a vector parallel to (dx_{kl}, dy_{kl}), influenced by the k^{th} eigenvector.
It is possible to approximate any shape in the training set using the following formula:

$$x = \bar{x} + Pb$$

where

- $P = (p_1\ p_2\ ...\ p_t)$ is the matrix of the first t eigenvectors.
- $b = (b_1\ b_2\ ...\ b_t)^T$ is a vector of weights corresponding to each eigenvector.
- $b = P^T(x - \bar{x})$ because the eigenvectors are orthogonal, thus $P^T P = I$.

By varying the parameters b within certain limits, it is possible to obtain new examples similar to those in the training set. The limits reflect the distributions of parameters present in the training set and can be chosen to comply with some chosen mathematical laws, for example the Mahalanobis distance from the mean can be set to be less than a predefined value, as follows:

$$D_m^2 = \sum_{k=1}^{t} \left(\frac{b_k^2}{\lambda_k}\right) \leq D_{max}^2$$

In order to better locate an object in a new unseen image, the grey-level appearance of the training set images can be processed [2].
The regions that lay around the landmarks in each image, due to the automatic labelling of corresponding points performed at the beginning, should have similar values of grey-level patterns. For this reason, by examining the statistics of the gray levels in those regions, it is possible to reach a more accurate positioning of the final model.
The simplest way of choosing a region surrounding the landmark is to focus on one-dimensional profiles normal to curves passing through the landmarks.
If the points are defined over the object boundaries, from each of them a profile is extracted, of length n_p pixels, centered at that point.
Bailes and Taylor proposed to sample the derivative of the gray levels on the profile in the image and then normalise it to obtain invariance to uniform scaling of the gray levels. As for the shape model, also in this case the mean profile is calculated together with a covariance matrix of dimensions (n_p x n_p) that describes the statistics of the profiles

corresponding to that point.
Again PCA is applied to obtain a model of the modes of variation in the grey-level profiles. We can then obtain the new grey-level profile, as previously computed for the shape, as:

$$g_{new} = \bar{g} + P_g b_{g(new)}$$

where

- $P_g = (p_{g1}\ p_{g2}\ ...\ p_{gt})$ is the matrix of the first t eigenvectors of S_g.
- $b_g(new)$ is a set of t_g parameters describing the profile model.

The parameters that best fit the model to g are:

$$b_g = P_g^T(g - \bar{g})$$

and the best fit of the model to g , represented in figure 2.12, is then:

$$g_{bestfit} = \bar{g} + P_g b_g$$

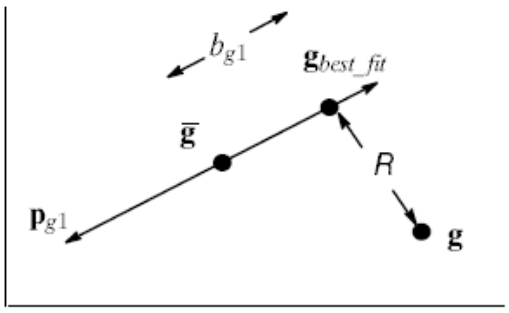


Figure 2.12: Simple model made of a single mode of variation p_{g1} and of the mean profile $\bar{g}$. $g_{bestfit}$ can be obtained projecting the point g on the line described by p_{g1} through the mean. The residual R can be calculated as the difference between the current point g and the point of best fit $g_{bestfit}$. [2]

Since only ($t_g < n_p$) eigenvectors are used in the model, there is a difference between the best fit obtained from the one that can be achieved if all the eigenvectors would have been used. This difference can be expressed as a sum of squares of differences and it is given by:

$$R^2 = (g - g_{bestfit})^T(g - g_{bestfit})$$

We can explicit $\bar{g}$ and b_g and rewrite the sum of squared error as:

$$R^2 = (g - \bar{g})^T (g - \bar{g}) - b_g^T b_g$$

If the eigenvectors are not truncated and $t_g = n_p$ eigenvectors are used, e measure of performance of the model in fitting the profile can be expressed with the Mahalanobis distance:

$$M = \sum_{j=1}^{n_p} \frac{b_{g_j}^2}{\lambda_j}$$

where λ_j is the eigenvalue of the i^{th} eigenvector.
The following equivalence can be used to express the performance measure as a function of R^2:

$$\sum_{j=t_g+1}^{n_p} b_{gj}^2 = R^2$$

If less than n_p eigenvector are used, to calculate b_{gj} for $j > t_g$, we approximate $\lambda_j = 0.5\lambda_{t_g}$ and we can use F as a measure of how well the model performs.

$$F = \sum_{j=1}^{t_g} \frac{b_{gj}^2}{\lambda_j} + \frac{2R^2}{\lambda_{t_g}}$$

F decreases approaching zero as the fit improves.

In order to recognize the modelled structures in new images an iterative procedure is necessary, made of the following steps:

1. Estimate a plausible initial position, orientation and scale for the model in the new image (figure 2.13).

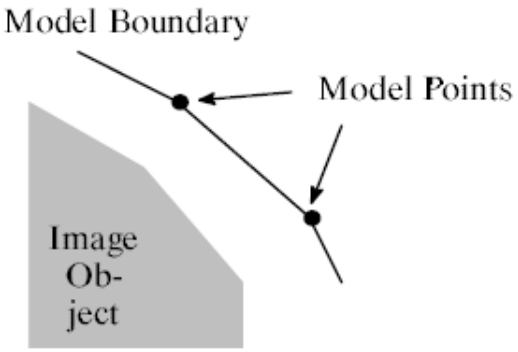


Figure 2.13: Example of initial position of the points of the model respect to the borders of the object to be detected. [2]

2. Choose a set of parameters b for a PDM and define the shape of a model instance x in a model centred co-ordinate frame.

3. Create an instance X of the model in the image frame as:

$$X = M(s, \theta)[x] + X_c$$

where

- $X_c = (X_c, Y_c, X_c, Y_c, ..., X_c, Y_c)^T$.
- $M(s, \theta)[]$ is a matrix used to scale x of a rotation by θ and a scaling by s.
- (X_c, Y_c) is the position of the centre of the model in the image frame.

4. Examine the neighbor pixels to each landmark to search for a better location.

5. Calculate the needed displacements.

6. From the displacements obtained in the previous point, obtain the adjustments to pose, scale and parameters of the PDM.

7. Update the model parameters.

This method is iterative and stops only when an accurate shape and pose that matches the structures in an image is obtained (based on the error calculation) or until no significant changes occur.

Movement estimation

For each landmark point of the model placed on a new image, a derivative profile is extracted with a certain length $l > n_p$ centred at that point and normal to the boundary of the model.
The profile model searches for the best point along that line in which the best match is found.
The performance measure F is then calculated along the sampled derivative profile for each point belonging to it and when a minimum is found, the value of distance from the center of the profile (landmark) is chosen (d_{best} is the point of best fit).
The displacement for the point belonging to the model (shown in figure 2.14) is calculated within a range chosen a priori $[\delta]d_{max}$ and corresponds to:

- $|dX| = 0$ if $|d_{best}| \leq \delta$.
- $|dX| = 0.5d_{best}$ if $\delta < |d_{best}| < d_{max}$.
- $|dX| = 0.5d_{max}$ if $d_{max} \leq |d_{best}|$.

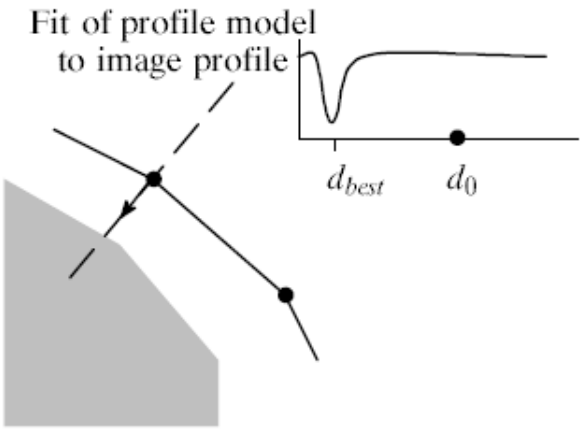


Figure 2.14: A point moves along the normal to the edges of the object in a direction defined by the point of best fit between the profile model and the profile sampled from the image. [2]

Adjustment to the pose and shape parameters

In order to move points from their current locations in the image frame to new locations that better match the image, it is necessary to find the translation, rotation and scaling factors and it can be done using a weighted least squares fit.
The latter adjustments correspond to deformations of the shape of the model in order to refine the final shape to better match the contours of the structures in the image.
The position of the points in the image frame is:

$$X = M(s, \theta)[x] + X_c$$

After the translation, rotation and scaling we have that:

$$M(s(1 + ds), \theta + d\theta)[x + dx] + (X_c + dX_c) = (X + dX)$$

Remembering that:

$$M^{-1}(s, \theta)[] = M(s^{-1}, -\theta)[]$$

we obtain the adjustments in the local co-ordinate frame

$$dx = M((s(1 + ds))^{-1}, -(\theta + d\theta))[M(s, \theta)[x] + dX - dX_c] - x$$

We have to transform dx into model parameter space, so that db can be used to adjust the model points as is allowed by the model.
The global shape constraints are respected by truncating the modes of variation (only $t(< 2n)$ are used).

With simply calculations, we finally obtain:

$$db = P^T dx$$

With all the adjustments obtained in these steps, we can update the pose and shape parameters with the following:

$$X_c \to X_c + w_t dX_c$$

$$Y_c \to Y_c + w_t dY_c$$

$$\theta \to w_\theta d\theta$$

$$s \to s(1 + w_s d_s)$$

$$b \to b + W_b db$$

To avoid unplausible shapes from occurring, the b-parameters should be limited as before discussed. If the constraint are violated during updating, the shape can be rescaled to lie within acceptable limits using

$$b_i \to b_i \frac{D_{max}}{D_m}$$

where

- D_{max} is an arbitrary constraint.
- D_m is the Mahalanobis distance calculated over $i = 1...t$ modes of variation.

The model presented above uses fixed descriptors to deform the set of points on the bases of image properties (such as the normalized first derivative profile).
Other features and descriptors can be used to deform the model [5]. An example of different implementation could be the use of gray levels of the image instead of the derivatives, a topic that will be explored in section 3.5.3 as part of the proposed method.

The Point Distribution Model can be extended to volumetric data. Apart for the space dimensioning, all the steps involved in the algorithm are unchanged respect to the traditional 2D ASM, providing a simply approach to extend the benefits of a simple, dynamic and fast algorithm to medical data.

2.3 Prostate Segmentation

2.3.1 Introduction

Prostate cancer is the most frequent cancer in the male population worldwide.
There are many techniques that allow to carry out the diagnosis, such as screening using the serum prostate specific antigen (PSA) or urological approaches, such as biopsies guided by transrectal ultrasound, however the method that demonstrates higher accuracy is MRI.

The use of MRI in prostate cancer diagnosis has the primary purpose of determining the stage of the disease, and possibly planning the necessary radiation therapy, both in form of external beam radiation, and in the form of a guide for the seed implant for brachytherapy.
In recent years, many studies have been carried our in this regard which have led to the conclusion that MRI is a technique that can accurately characterize the presence of focal lesions in the whole gland, a feature that is very useful for carrying out a possible biopsy [16].

Prostate cancer diagnosis systems make use of automatic segmentation of the prostate and of specific areas of it to determine a region of interest (ROI) in which the tumor is most likely to be found. In order to perform a segmentation, be it manual or automatic, it is important to know the anatomy of this organ.

The prostate can be mainly divided into three parts (shown in figure 2.15).

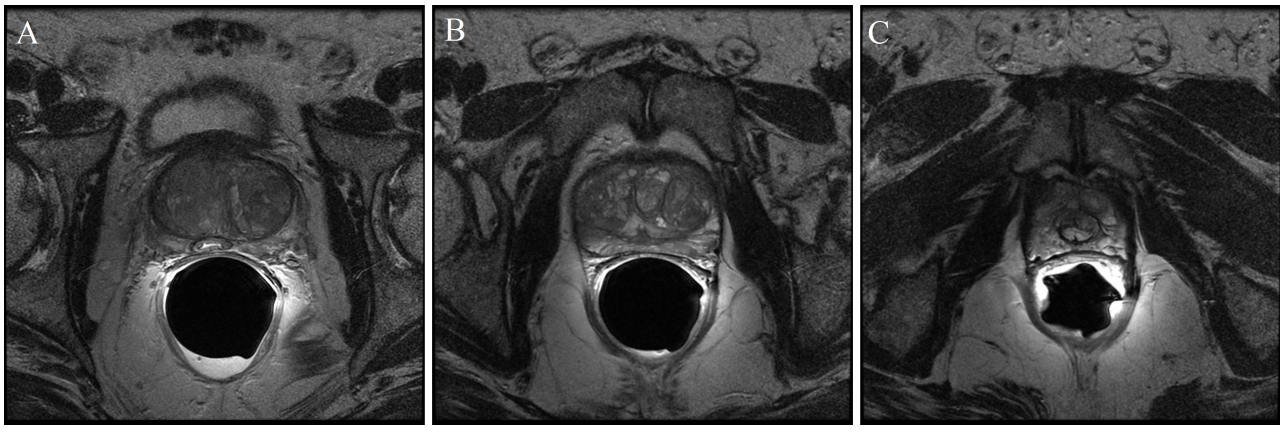


Figure 2.15: Sections of the prostate in axial plane: A) base, B) mid gland, C) apex.

- Base: contains parts of the Central Zone, Peripheral Zone and Transition zone in equal measure.
- Mid gland: contains mostly Peripheral Zone and Transition Zone.

- Apex: contains mostly Peripheral Zone and some Transition Zone.

The histologic zones included in the three anatomical levels in which the gland is divided from superior to inferior (PZ, TZ and CZ) are better detailed in figure 2.16, together with the AS (Anterior fibromuscolar Stroma).

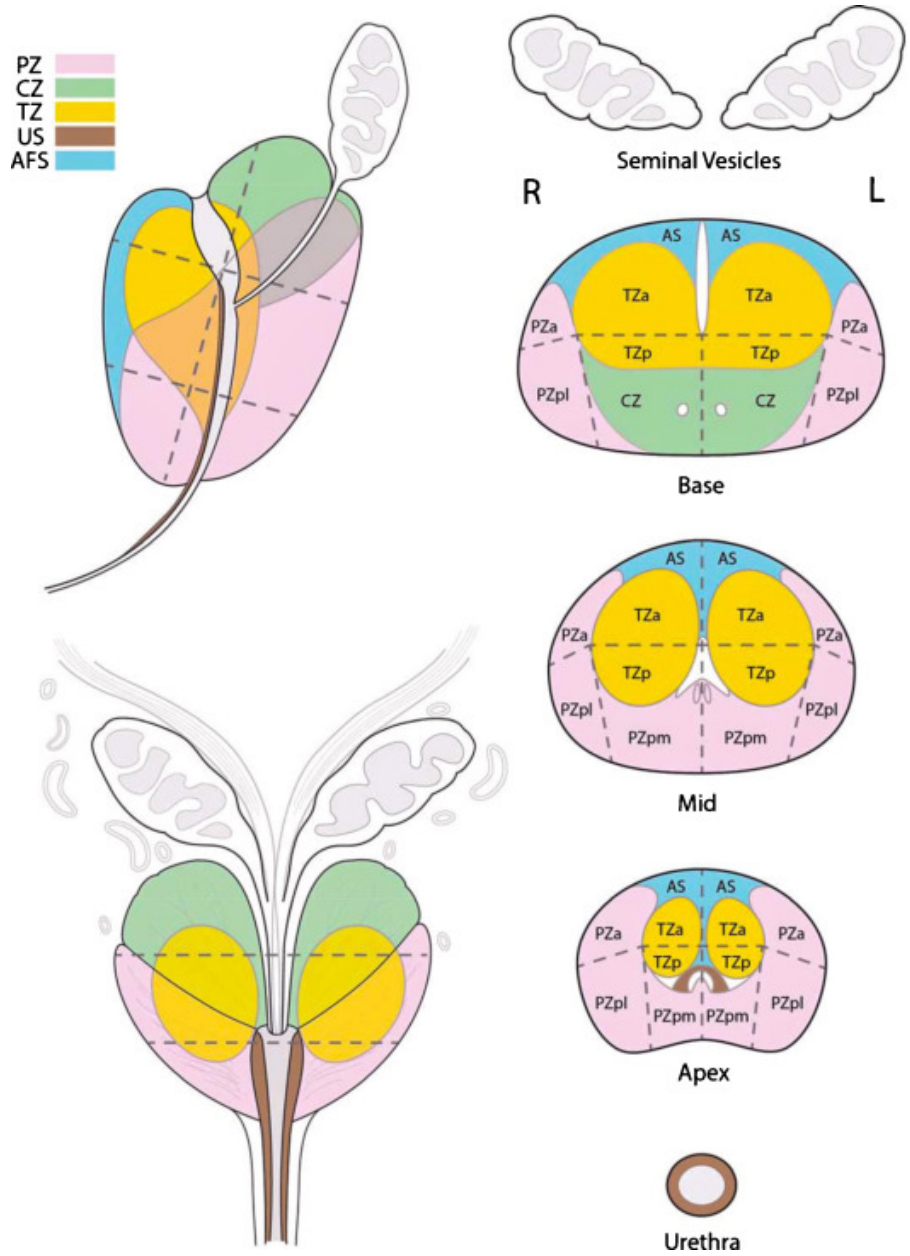


Figure 2.16: Prostate anatomy with division in peripheral zone, central zone, transition zone and anterior fibromuscular stroma. In the upper part of the organ we can observe the seminal vescicles, while in the center we have the urethra. [27]

Challenges of MRI prostate segmentation

MRI is considered the best acquisition technique for what concerns the studies of the prostate and diseases connected with this organ, especially of cancer which can be recognized in MR scans because of huge differences in tissue texture. MR provides better image quality respect to other techniques, it has superior tissue contrast that allows for more accurate tissue discrimination.
The spatial resolution of this type of scanning is higher as well and it has multiplanar capability, that allows for direct coronal or sagittal imaging, thus representing structures in a way closer to the real anatomic.
It has also no radiation risk, thus avoiding harm to human body.

Although this technique guarantees so many advantages, the segmentation of prostate MRI images/volumes presents some difficulties as well, summarized below.

- Indistinct prostate boundaries $\rightarrow$ the appearance of the prostate is similar to the one of surrounding tissues (blood vessels, bladder, rectum and seminal vessels)
- Small effective area of the prostate in the image/volume $\rightarrow$ less information available.
- Large variability in prostate shape among patients.
- Heterogeneity in signal intensity around the endorectal coil.
- Bias field.
- Anisotropic spatial resolution.
- Global inter-scan variability due to different MR scanning protocols.
- Intra-scan intensity variation.

Prostate cancer can be diagnosed with accurate systems (such as CAD systems) that make use of automatic image segmentation to gain anatomical information about the prostate structure and the tissues surrounding it and to restrict the ROI on which to train to a region comprising mostly the object, solving the problem of the small effective area.

As mentioned in the first chapter, clinical manual segmentation suffers from three main problems. It is time consuming, poorly reproducible and error prone, because it depends on the clinician experience and on intra- and inter- operator bias. Automatic algorithms for image segmentation proved to be more efficient and they provide more repeatable and less erroneous results.

2.3.2 State of the art in prostate MRI segmentation

Traditional segmentation algorithms make use of anatomical atlas registration, deformable models, region growing, machine learning, but all these techniques suffer from the problems discussed in section 1.3.
Modern approaches employ deep learning procedures that guarantee better performances, preceded by a little pre-processing, employed for algorithm stability (better contrast, finer details).

We present a list of modern proposals, along with the motivations that led to the development of the specific architecture and a brief explanation of the method.

Holistically Nested Networks with Short Connections

Problem
Lower z axis resolution can produce large segmentation errors; many approaches focus only on the 2D axial slices, ignoring information coming from the coronal and the sagittal view.

Method proposed [19]
2-D HNNsc (figure 2.17) applied to an orthogonal context followed by 3-D surface reconstruction and 3-D mesh optimization to eliminate noise caused by the deep learning model.

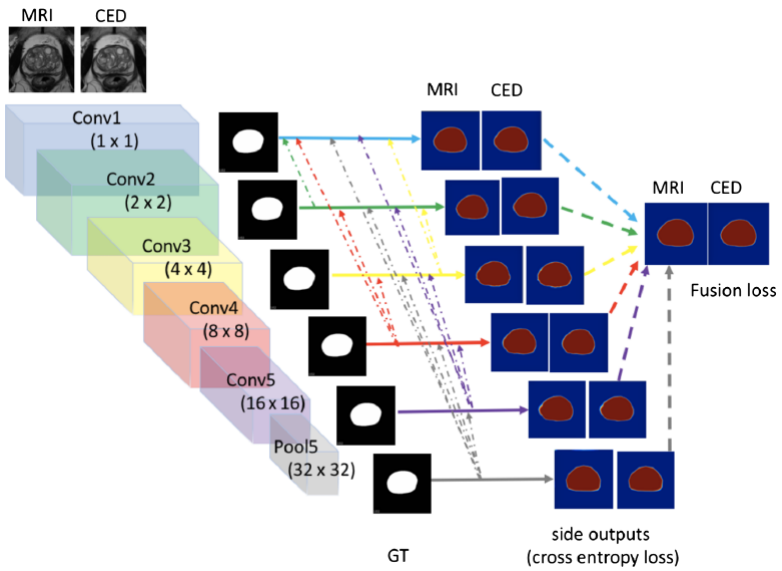


Figure 2.17: HNNsc architecture: outputs from the convolutional layers at each stage are transformed into probability maps to form the final probability map for both MRI and CED images. Short connections are added connecting the deeper side outputs to the ones at a more superficial layer. [19]

The training set is composed by pairs of MRI and CED images which are introduced into the network that holistically learns their deep representations and produces internal convolution neural networks between these and the corresponding ground truth binary mask (that acts as deep supervision).
The reason for the implementation of short connections is to make deeper side output guide the superficial layers to predict in a more accurate way the ROI. At the same time, the shallower layers refine the predicted values of the deep side outputs.
A Surface Reconstruction algorithm is finally applied to obtain a 3D surface from the combined axial, sagittal and coronal views.

Encoder-Decoder with Dense Dilated Spatial Pyramid Pooling

Problem

Large variability of prostate boundaries and artifacts in the images due to surrounding tissues.

Method proposed [4]

A deep neural network based on an encoder-decoder structure with dense dilated spatial pyramid pooling (DDSPP) employed in the encoder module to generate a higher range of features, widening the receptive field of the convolution kernel.

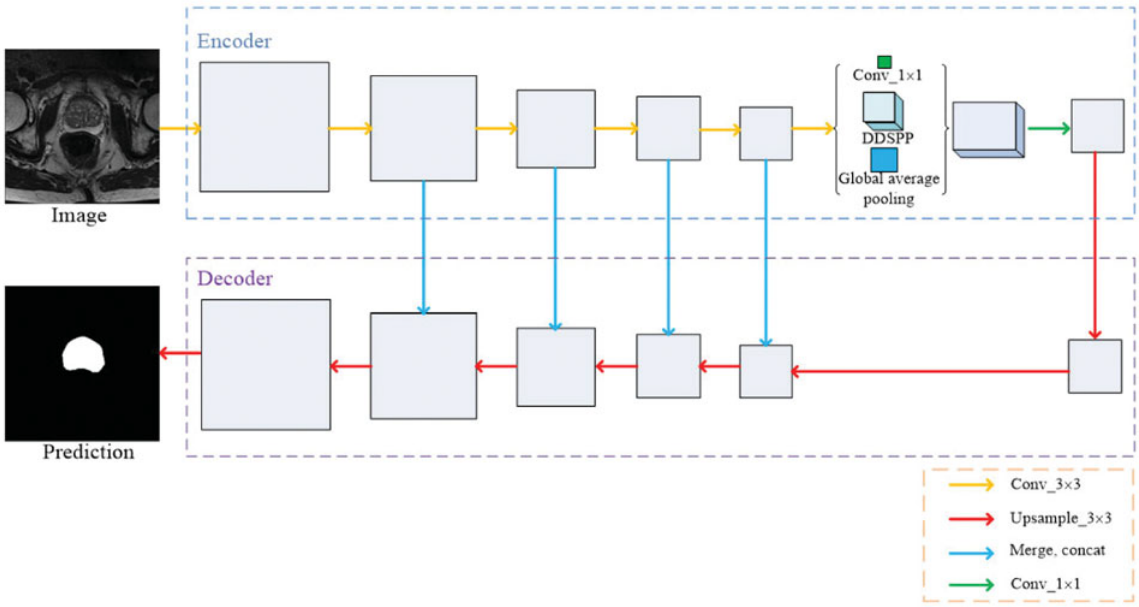


Figure 2.18: Encoder-Decoder architecture: the first encoder module extracts information at multiple scales and subsequently a decoder module is applied to refine the segmentation especially around prostate borders.[4]

3D Adversarial Pyramid Anisotropic Convolutional Network

Problem

Anisotropic spatial resolution of prostate MR images, due to high intra-slice and low inter-slice resolution.

Method proposed [7]

3D APA-Net made of two modules (figure 2.19), the first (PA-Net with an encoder-decoder structure) segmenting the prostates and the second, a 3D adversarial DCNN called 'discriminator', checking whether the image is multiplied by the segmentation result or the ground truth, in order to refine the final result.

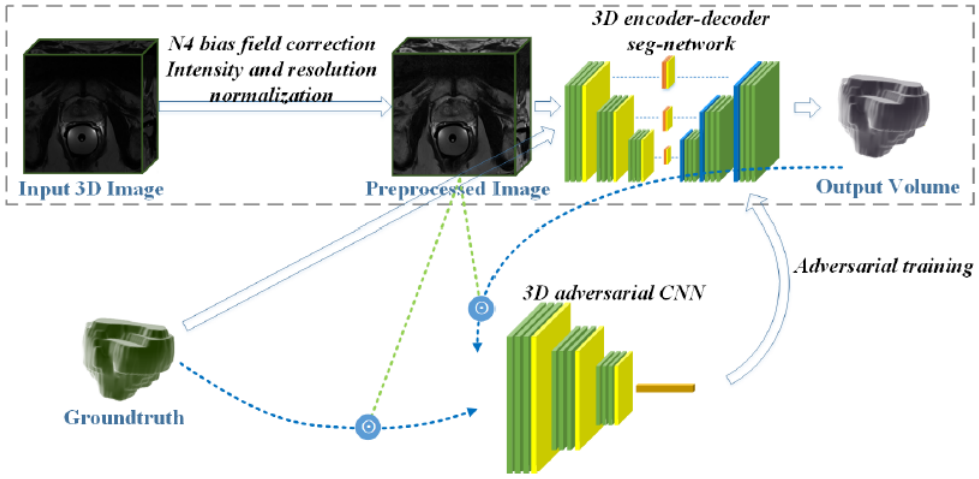


Figure 2.19: 3D APA Net pipeline: A pre-processed image is fed into the network that has an encoder-decoder structure. The dotted box represents the segmentation process, while out of the box we can observe a discriminator based on a 3D adversarial DCNN, which takes the ground truth and the inference output multiplied by the pre-processed image as input for discrimination.[7]

Volumetric ConvNets with Mixed Residual Connections

Problem

Improve training efficiency and capability of the network to discriminate examples under limited training data.

Method proposed [29]

Volumetric ConvNet (figure 2.20) made of two paths, the first one down-sampling the image features and the second one up-sampling to generate the final prediction.

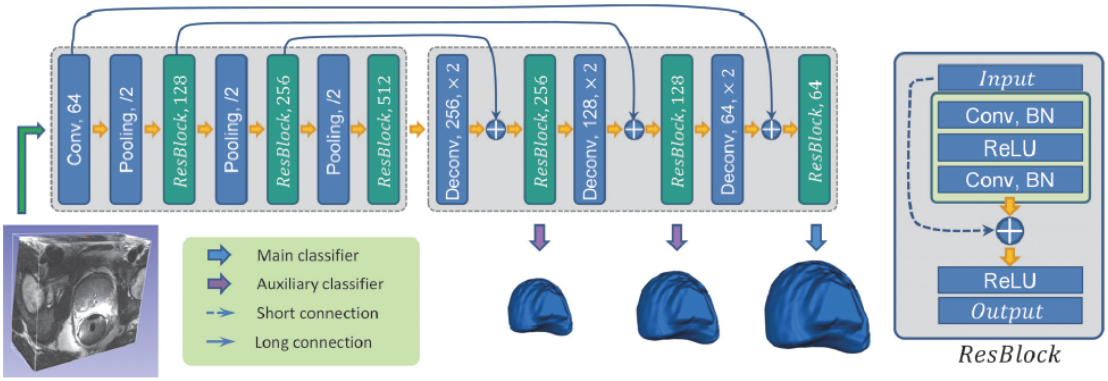


Figure 2.20: Architecture of the volumetric ConvNet. The first path downsamples the image and the second path upsamples it to restore the original dimensions and create the output segmentation. Between the two paths residual connections are inserted to preserve spatial information. [29]

A deep supervision mechanism is added to the basic structure and residual connections are included to preserve spatial context information.

U-Net

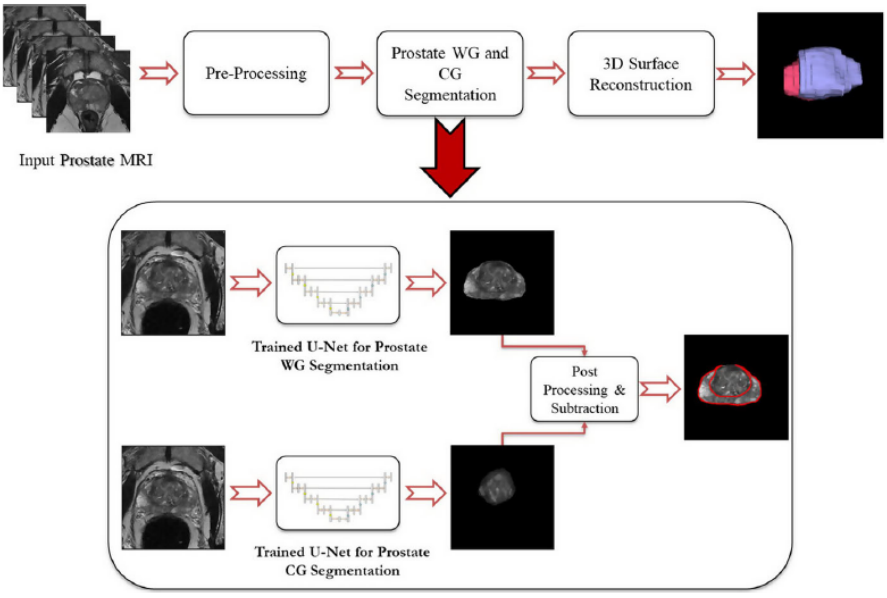


Figure 2.21: Pipeline of the method: After a first preprocessing step, two UNets are used to segment the WG and the CG separately. The outputs of these networks are postprocessed and the CG is subtracted from the WG to obtain the TZ. A 3D reconstruction algorithm is then applied to reobtain the volumetric output. [30]

Problem

Automatic segmentation of prostate whole gland and transitional zone on T2W images and corresponding ADC maps.

Method proposed [30]

Two separate U-Nets for segmenting the whole gland and the CG and subsequent subtraction to obtain the TZ (figure 2.21). The UNets present a CNN-based architecture, made of a first shrinking path followed by an expanding path.

Cascaded Fully Convolution Network

Problem

Segment the whole gland of the prostate and in sequence, the peripheral zone on T2WI.

Method proposed [31]

Cascade of U-Nets (figure 2.22), one to segment the WG and another identical in cascade to segment the PZ.

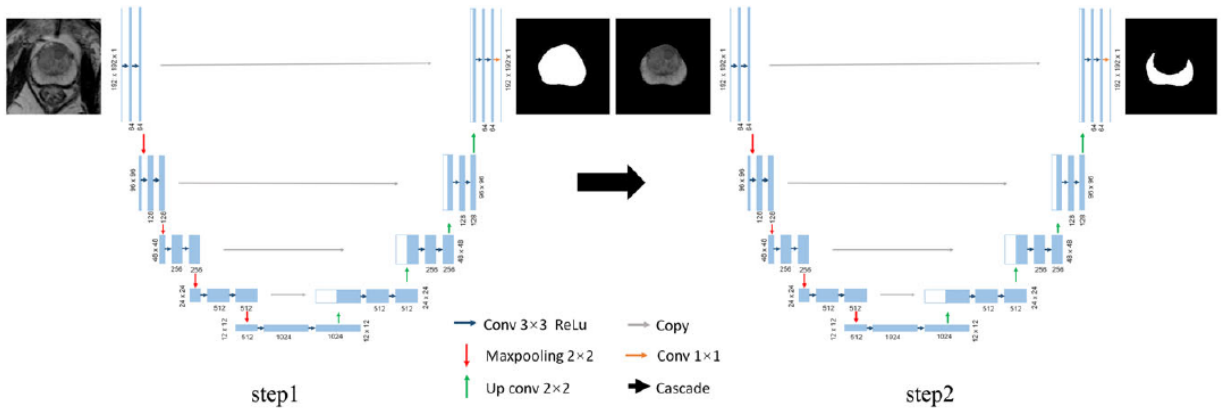


Figure 2.22: Method pipeline: images enter the first UNet that segments the WG. The results of the first network are input into the second network to segment the PZ. [31]

Boundary-Weighted Domain Adaptive Neural Network

Problem

Provide a method capable of adapting to different training sets acquired with different modalities and make the segmentation of the prostate boundaries more accurate, even with weak contours.

Method proposed [32]

3D network inspired by U-Net and Dense Net, designed for domain image segmentation network (SNet). Two SNets segment the source and target domain images separately. A discriminator differentiates the feature representations coming from the first or the second working in ad adversarial fashion.

Both SNet-s and SNet-t follow the classic U-Net configuration, with a first down-sampling path (convolutional blocks, average pooling layers and densely-connected residual blocks) and a second up-sampling path (deconvolutional layers and densely-connected residual blocks).

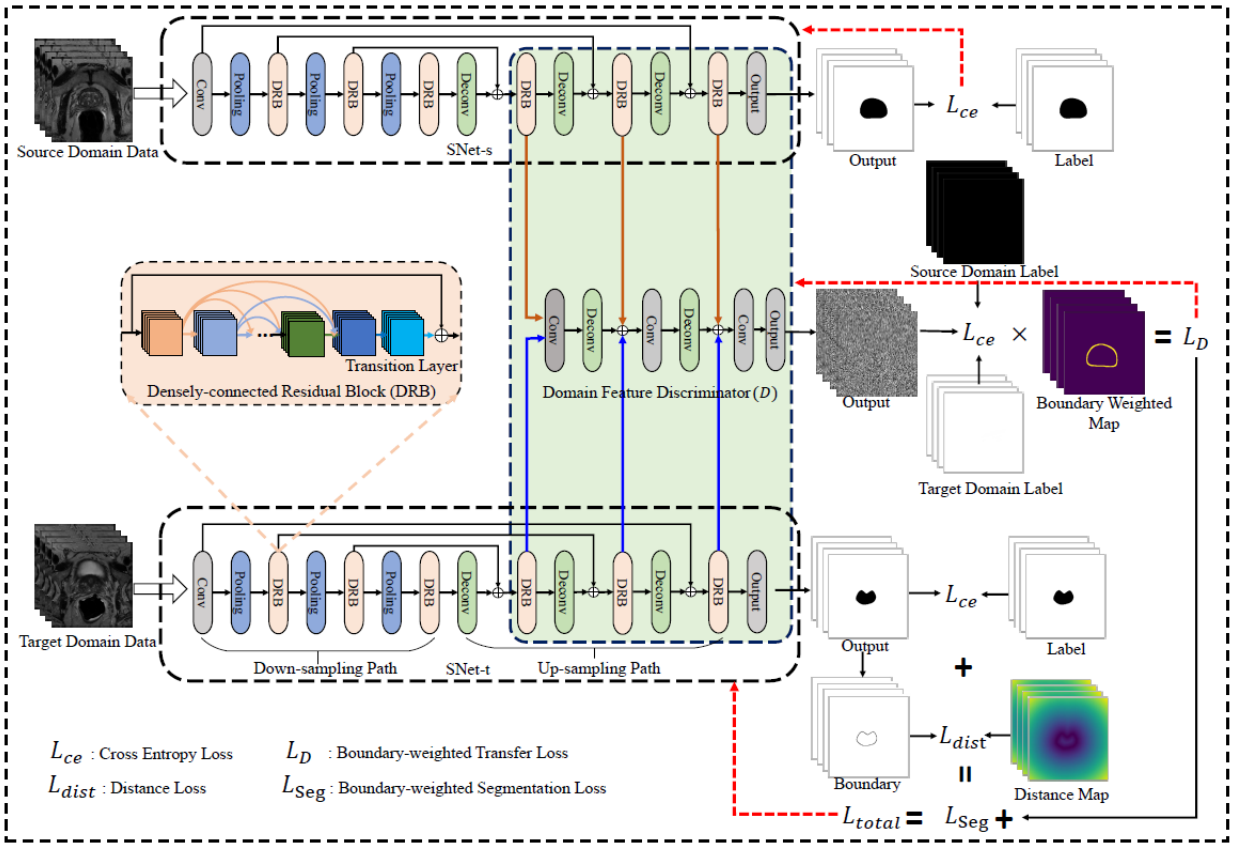


Figure 2.23: Method pipeline: two SNets are built and trained on images belonging to different domains. The outputs from these networks are sent to a discriminator that tries to differentiate features coming from a network from features coming from the other in an adversarial fashion. The total label is obtained as the sum of the label coming from the segmentation of the two networks and of the label obtained from the discriminator. [32]

Hybrid approaches

Some approaches are based on the complementary used of deep learning techniques and statistical models to try to derive from each of these the specific advantages and mix them together to obtain higher performances.

Traditional statistical models alone have problems segmenting the prostate in MRI images because of the great variability of contours among different patients. They are rarely used alone because they require handcrafted features to model the profile on the edges.
Furthermore, the segmentation in correspondence with the contours is complicated by the fact that the initialization of the shape is often erroneous or not precise enough. ASM is sensitive to this shape initialization due to the constraints on the shape imposed by the model parameters and the variable appearance of the prostate among different subjects makes the segmentation task even more complicated.
On the other hand, segmentation models based solely on CNN frameworks take inspiration from methods that were initially developed to segment natural images. These networks are not specific for the medical domain and consequently they do not take into account two important factors:

- Less amount of data available for training.
- The volume to be segmented in medical images has less variability in shape and appearance than in natural images.
  If the images do not have a high number of discriminative features, it is possible that different classes are not well discriminated because positive and negative samples can have similar appearance.

For these reasons, it can be useful to know a priori the shape of the object to be segmented; the segmentation can be improved by integrating knowledge on the variability in the prostate shape within a CNN-based framework.

The latest algorithms proposed use for example deep learning techniques to generate an initial probability map of the voxels belonging to the prostate, which is then modified using deformable models to better adapt to the contours.
The two approaches can be integrated in different ways. The statistical model can be either a shape model or a more complex model that takes into account the characteristics relative to the image texture (appearance models).

Three methods that incorporate in different ways the use of deep features and statistical parameters for prostate segmentation are briefly presented below.

Adaptive Feature Learning Probability Boosting Tree Initialization and CNN-ASM Refinement

This hybrid approach combines an adaptive feature learning probability boosting tree (AFL-PBT) with a deep model, specifically a CNN network and an active shape model (figure 2.24) [6].

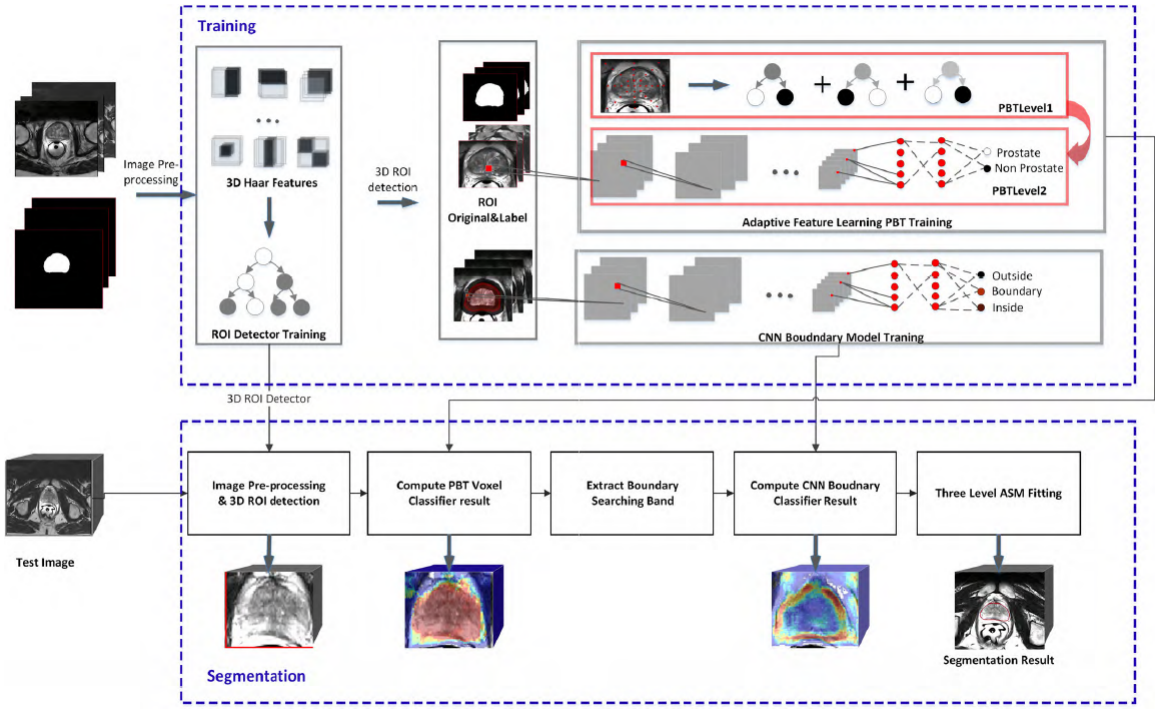


Figure 2.24: Method pipeline: the first training phase starts with a 3D ROI detection followed by the AFL-PBT and CNN models. The test phase consists in computing the PBT voxel classifier result, computation of boundary searching band and CNN classification with final ASM refining. [6]

The AFL-PBT is based on both handcrafted features and features obtained through deep methods for the pre-segmentation of the prostate and for the creation of a boundary searching band used for the initialization of the statistical model.
The training phase includes an initial 3D ROI detector training to find the prostate inside the volumes, followed by the AFL-PBT which takes the form of a hierarchical classifier that clusters samples belonging to the prostate or not, applying the CNN model to find informative features also for those samples similar to each other in correspondence with the edges.
In the test phase 3D ROI detection is performed again and a probability map is obtained by applying the AFL-PBT followed by the CNN model to train a boarder classifier to find internal, external or prostate contour points. The ASM is initialized in the searching band output from the the AFL-PBT and in this region points of the prostate surface are moved according to the probability map of the edges of the prostate.

Active Appearance Model followed by CNN refinement

This method [8] uses an active appearance model (AAM) as initialization to obtain an estimate of the prostate edges with the MRI volumes.
The AAM alone produces borders that are not always accurate; the shapes obtained are

often irregular and the segmentation results unstable in the search range of the model. For this reason, a CNN is applied to obtain a refining of segmentation at the prostate edges (figure 2.25).

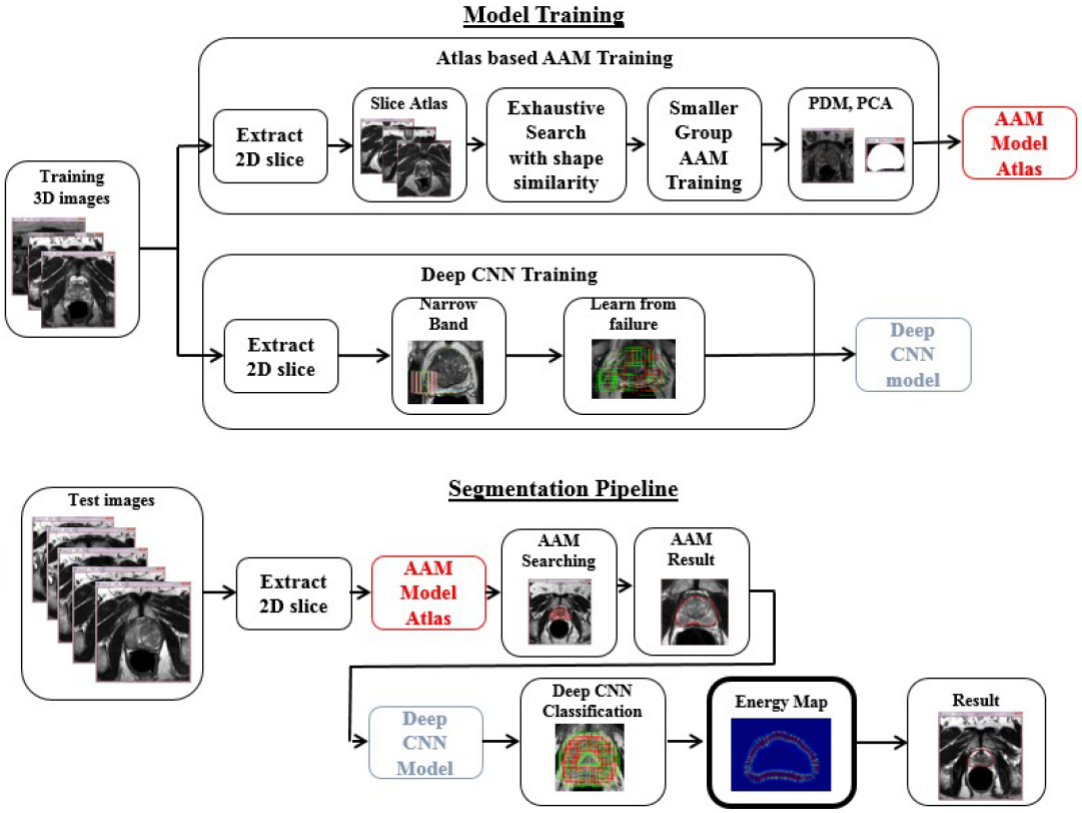


Figure 2.25: The training phase consists in the application of both Atlas based AAM and Deep CNN models. In the test phase the AAM model is applied to the test images, followed by the CNN model which refines the classification on the prostate borders obtaining the final segmentation.[8]

The network training phase takes as input patches around the contour of the prostate obtained with the AAM and the CNN outputs the probability for the voxels to belong or not to the contour.

CNN architecture with training strategy based on an Active Shape Model

This third approach uses a CNN-based architecture and a training strategy based on an active shape model to solve the problem of variability in shape and the limited number of training data (figure 2.26) [1]. This strategy foresees to limit the output of the CNN only to shapes allowed from the statistical model.
The ASM is incorporated into the network as a set of fully connected layers which output the coefficient vector representing the shape of the prostate in the learned shape model.

The network is initially trained to find the center of the prostate and subsequently re-

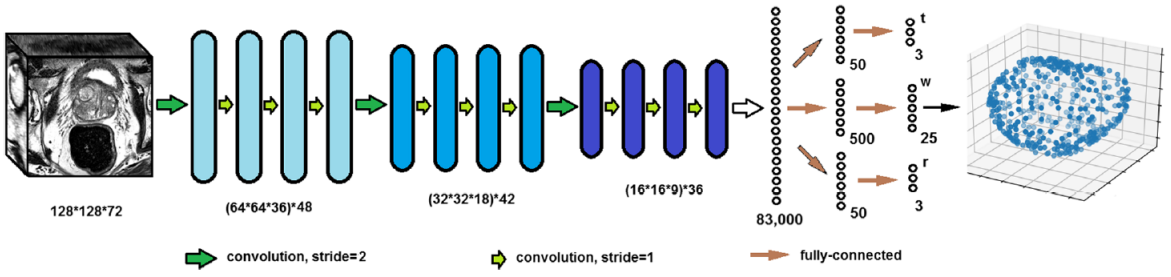


Figure 2.26: Method pipeline: CNN-based framework with ASM parameters included into the final network layers.[1]

trained by introducing the layers related to the shape model adding a regularization term to the cost function relative to the shape model coefficients that allow to deform the model.
The shape model incorporates the information on the expected shape and the variations that this shape can have and the CNN estimates the parameters of the shape model that best describe the organ and outputs points that represent the surface of the prostate.

Table 2.3.2 shows a comparison between the above-mentioned methods, highlighting the most significant characteristics, i.e. the problems that each method tries to solve, the proposed solutions, the dataset used for this purpose, the pre-processing carried out on it, the network architecture and the results obtained.

Method proposed: CNN segmentation followed by an ASM refining

Our proposal presents an approach that starts from a first rough segmentation of the prostate using a CNN model, specifically the UNet, to obtain a first three-dimensional segmentation of the organ within the MRI prostatic volume.
The proposed network model is three-dimensional, so it gives more importance to the spatial coherence that allows to obtain the organ in its entirety, if compared to a two-dimensional network that instead works on the single slices. The three-dimensional segmentation, while having this spatial advantage, focuses less on the precision on the single slice, especially in correspondence with the prostate edges. For this reason, a statistical model is proposed to carry out a refining of the prostate contours on the CNN outputs to obtain a more precise segmentation and to adjust possible unplausible shapes output from the network as they are corrected according to the shape constraints defined by the ASM parameters.
In the next chapter our method will be described in detail and in section 4.1 our results will be examined to better understand the strengths and limitations of using a hybrid model.

Method	Problem	Solution	Database	Pre-processing	Architecture	Score DSC
HNNsc	Low z axis resolution of prostate images	Isotropic upsampling train on orthogonal context	145 T2W images 0.35 x 0.35 x 3.0 mm 512 x 512 x 26	- N4ITK correction - isotropic upsampling to 0.35 x 0.35 x 0.35 mm - reorientation - CED filter - intensity normalization	Holistically nested network with short connections - downsampling conv path - probability maps at each stage averaged to compute the final output - 3D surface reconstruction - mesh optimization	WG 92.35 $\pm$ 3
Encoder-Decoder with DDSP	Large variability of prostate boundaries	Wider receptive field to have a higher connectivity between the submodules	1392 images 256 x 256	not specified	Encoder-Decoder - Encoder reduces feature maps - Decoder recovers space info - DDSP to enlarge receptive field	WG 95.4
3D APA-Net	Anisotropic spatial resolution	As-Conv block: two anisotropic convolutions for x-y features and z features independently	- 80 T2W images intra- res 0.25 - 0.75 mm inter- res 2.2 - 4.0 mm - 60 T2W images intra- res 0.39 - 0.75 mm inter- res 3.0 - 4.0 mm	- resizing to 0.625 x 0.625 x 1.5 mm - normalization - cropping to 96 x 96 x 32	3D PA-Net with adversarial training - Encoder-Decoder with skip connections - DCNN used as a discriminator to refine output	WG 90.6 base 89.6 apex 86.9
Volumetric ConvNet with mixed residual connections	Limited training data	Residual connections to improve information exchange among layers	80 T2W images intra- res 0.25 - 0.75 mm inter- res 3.0 - 4.0 mm	- resizing to 0.625 x 0.625 x 1.5 mm - normalization	3D ConvNet - downsampling path - upsampling path - residual connections	WG 89.42 base 86.42 apex 86.81
UNet	Accurate segmentation of both WG and TZ	Mixed segmentation on T2W and ADC maps to obtain more precise results	- 225 T2W images 0.3906 x 0.3906 x 3.4 mm - 225 ADC maps 1.1719 x 1.1719 x 5 mm	T2W - cropping to 256 x 256 - normalization ADC maps - cropping to 128 x 128	4 UNets - shrinking path - expanding path - output probability map	WG 92.96 $\pm$ 7.8
Cascaded FCN	Accurate segmentation of both WG and PZ	Cascaded architecture	- 1416 T2W images 240 x 240 x 4 mm 324 x 280 - 1416 DWI images 240 x 240 x 4 mm 184 x 184	- ROI selection with k-means clustering - ROI resizing to 192 x 192	Cascade of 2 UNets - shrinking path - expanding path - output probability map	WG 92.7 $\pm$ 4.2 base 91.5 $\pm$ 4.5 apex 94.6 $\pm$ 2.6

(Continues in the following page)

(Continues from the previous page)

BOWDA-Net	Prostate variability Weak contours	Adversarial training: use of two networks of different domain data Long connections between downsampling and upsampling path to preserve context information	- target domain 50 T2W images - source domain 81 T2W images 0.27 x 0.27 x 3 mm 512 x 512 x 26	target domain - resampling 0.625 x 0.625 x 1.5 mm - normalization source domain - normalization	BOWDA Net - Source SNet - Target SNet - UNet architecture - Feature discriminator Adversarial training - Densely-connected residual blocks	WG 91.4 base 89.6 apex 89.3
AFL-PBT with CNN-ASM refinement	Variability in prostate shape and appearance among different parts and subjects	Combine an adaptive feature learning probability boosting tree with CNN to train a boundary model and refine the segmentation with an ASM	Training 50 T2W axial MR images Testing 30 T2W axial MR images 512 x 512	- Intensities normalization - 3D ROI detection - Resampling to 0.625 x 0.625	Training phase: - AFL-PBT - CNN boundary model Testing phase: - Pre-segmentation from the probability map obtained with AFL-PBT and boundary probability map computed by CNN boundary classifier - ASM fitting	WG 0.84 ± 0.04
CNN with training strategy based on ASM	Variability of prostate shape and appearance Small amount of training data	Stage-wise training strategy Train a CNN network to first predict prostate center and then predict parameters of the shape model within the CNN architecture	Training 49 T2W axial MR images Testing 26 T2W axial MR images	Isotropic resampling to 128 x 128 x 72 with B-spline interpolation	- CNN network with a shrinking path followed by separate sets of FCN layers - ASM FCN output fed to final layer which predicts surface keypoints	WG 0.88
AAM - CNN	- Image artifacts - Large inter-patient shape and texture variability - Unclear boundaries	Atlas based adaptive AAM for a first boundary estimate combined with a CNN model to refine prostate boundaries in a 2D context	Training 100 axial MR images Testing 20 axial MR images 0.2734 x 0.2734 x 3.0 mm 512 x 512	not specified	- AAM training with shape model, texture model and shape-texture combined - CNN trained on patches around the AMM segmented prostate boundary line - Learn-from-failure approach: perturb AAM model to generate errors in segmentation which guide CNN segmentation for better refinement	WG 0.925

Table 2.1: Comparison between methods found in literature. The table presents for each method the problem faced with the proposed solutions, along with details on the datasets used for the scope, the pre-processing steps, the architectures and the score achieved. We can observe that when evaluating the network performances, some methods take into account the division of the prostate zones in the segmentation, as it is more complicated to obtain precise results in correspondence of the base and the apex of the organ.

Chapter 3

Materials and Method

The proposed method involves the creation of a custom made 3D CNN for the segmentation of prostate volumes, followed by a refining of the edges of the output segmentation carried out using a 3D Active Shape Model.
The reason behind the use of a 3D network is the fact that it allows to exploit the spatial information that derives from the use of the entire volume instead of the single 2D slices, and of the connections created between the voxels of a slice and those belonging to the following slice.
To demonstrate the superiority of the 3D method over the 2D approach as regards spatial coherence, a comparison is proposed with a 2D network trained on a lower number of slices respect to the ones of all volumes used in the 3D network.

As previously anticipated, the hybrid approach is proposed in order to exploit the advantages of both deep networks and statistical models. The deep 3D network allows to obtain a complete segmentation of the prostate volume without spatial inconsistencies, however it has problems of precision in the segmentation of the edges of the organ.
The active shape model mitigates this problem by bringing the contours of the segmentation output from the CNN closer to the real ones and guaranteeing the plausibility of the shapes obtained as they are limited by the model parameters.

In the following sections the dataset available for this study will be initially presented, with the work done to obtain the manual annotations.
Subsequently, the pipeline of the method will be presented with a brief description of each individual block which will be then analyzed in detail in separate specific sections.

The two networks and the ASM model will be tested on both the training and the test sets to make comparisons between the three models proposed and the results will be properly explained with pros and cons in the final section.

3.1 Database and Manual annotations

The study has been performed using a dataset composed of T2W MRI prostatic volumes. These images have been obtained using the T2 weighted pulse sequence in MRI, which relies upon the transverse relaxation of the net magnetization vector. T2 relaxation consists in the decay of the spins from their aligned precession in the transverse plane, after the RF excitation pulse has been applied and the spins have relaxed from the transverse plane toward the main longitudinal magnetic vector, which corresponds to T1 weighting.

Among the available patients, a number equal to 60 prostatic volumes has been selected, sufficiently different (as shown in figure 3.1) to ensure that the proposed algorithm could have a wide range of image features to better generalize on new unseen data.

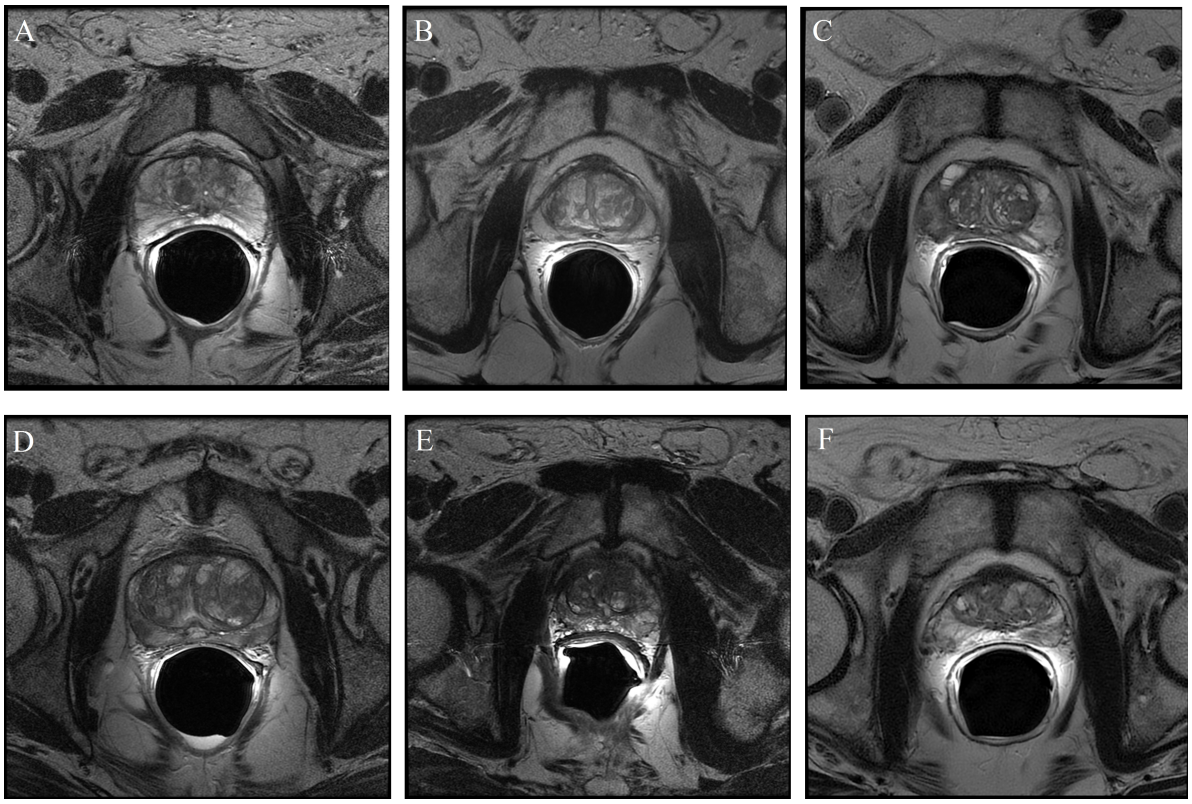


Figure 3.1: Example of the variability in shape among some prostates belonging to both the training and the test set in the axial plane. A) patient 2002 slice 11, B) patient 3030 slice 13, C) patient 3160 slice 13, D) patient 13 slice 12, E) patient 31 slice 13, F) patient 3211 slice 11.

These volumes have been initially divided into two data sets, one containing the patients relative to the training set and validation set and a separate one with volumes used for the test.
The subdivision between training and validation set will be carried out in correspondence with the definition of the 3D CNN input.

The original resolution of most of the volumes on the three planes and their dimension are respectively 0.3125 x 0.3125 x 3 mm and 512 x 512 x 24.
As regards images with a dimension different from the most represented one within the dataset, they have been mapped to have consistent dimensions as required by the network parameters, by performing a 'zero padding' operation adding slices at the end of the volumes to obtain volumes of 512 x 512 x 24 as network input.
Each volume together with the corresponding label is resampled using an isotropic upsampling in order to apply the Active Shape Model.

Each three-dimensional volume is obtained by stacking together a series of images in DICOM format that represent the individual slices, using a function that allows to transform the resulting 3D array into a nrrd file, a format used in the field of medical images that allows to save together with the data representing the value of each voxel, the metadata regarding other image characteristics, which are contained in the file header.

The space in which the DICOM images are sampled is called anatomical space or patient coordinate system and it consists of three planes that describe the standard anatomical positions of a human:

- Axial plane parallel to the ground
  It defines the subdivision between the Superior and the Inferior parts.
- Coronal plane perpendicular to the ground
  It defines the subdivision between Anterior and Posterior parts.
- Sagittal plane
  It separates the Left from the Right.

Different medical applications use different definitions of the 3D basis on which the anatomical coordinate system is defined.
DICOM images use the LPS basis (Left-Posterior-Superior) defined as:

$$LPS = \begin{Bmatrix} \text{from right towards left} \\ \text{from anterior towards posterior} \\ \text{from inferior towards superior} \end{Bmatrix} \tag{3.1}$$

Manual annotations have been obtained using the 3D Slicer software.
The images are loaded from the directory containing the individual slices into the working page, which allows to display them sequentially as if they represented a volume.

The labels have been obtained manually by looking for the prostate within the volumes on the axial, coronal and sagittal planes.

The convention that this software uses to define the 3D basis of the anatomical coordinate system is the RSA (Right-Superior-Anterior) defined as:

$$LPS = \left\{ \begin{array}{r} \text{from left towards right} \\ \text{from posterior towards anterior} \\ \text{from inferior towards superior} \end{array} \right\} \tag{3.2}$$

which is similar to the LPS, with the first two axes flipped.
For this reason, once the volumetric labels in .nrrd format have been saved, they are flipped to match with the convention used for the volumetric images.

An example of manual annotation is shown in figure 3.2 in green, superimposed on the original image in axial, sagittal and coronal views for a patient belonging to the training set.

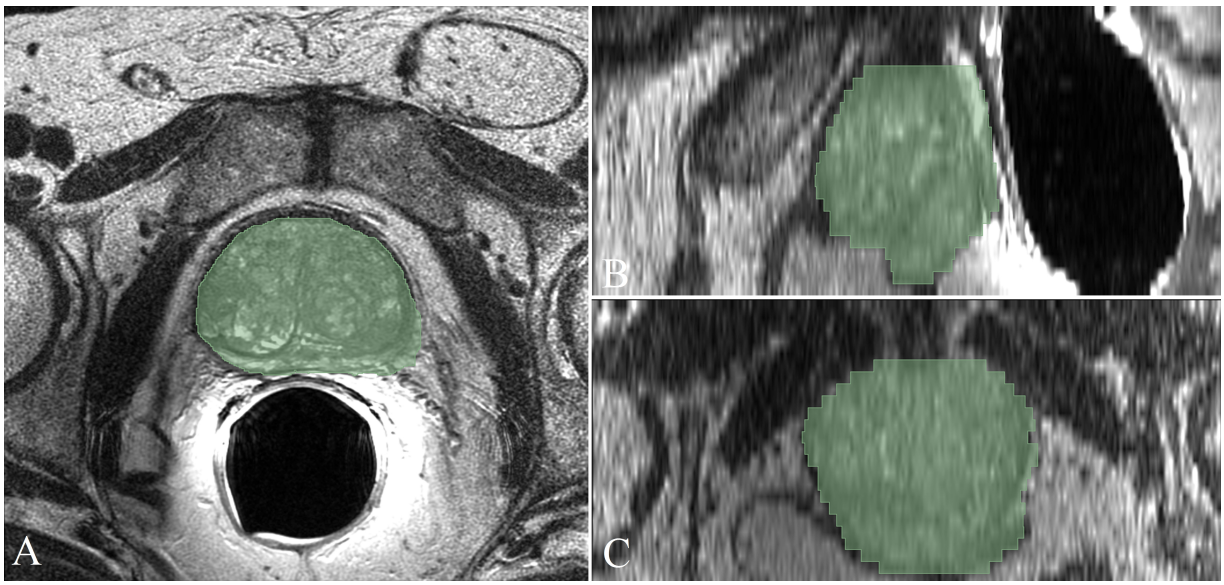


Figure 3.2: Manual label superimposed on MRI image in A) axial, B) sagittal and C) coronal views for patient 55.

3.2 Pipeline

The general pipeline of the method is shown in figure 3.3. The green blocks represent the steps related to the 3D CNN and to the active shape model. The algorithms for the research of coherent vertices and for the 3D reconstruction applied in the 3D case are also fully automatic.

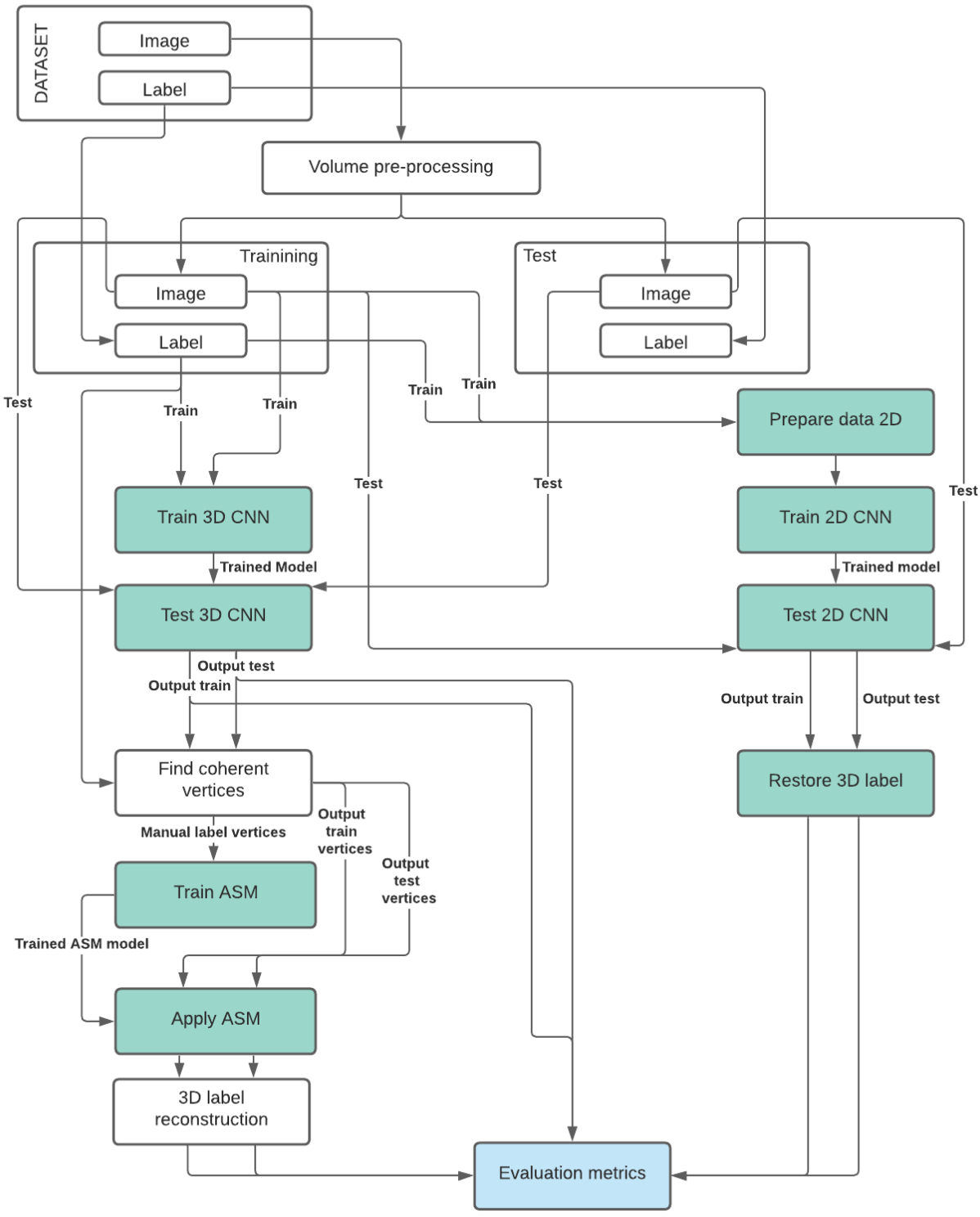


Figure 3.3: General pipeline of the method: the entire dataset is first preprocessed, then subdivided into training and test set. Training images and labels are used for the training of the 3D and 2D CNNs. Manual 3D annotations are also used to train the ASM model, tested on the output of the 3D CNN. After applying the ASM, 3D labels are restored.

The first step performed on the volumes is the pre-processing, illustrated in detail in figure 3.4.

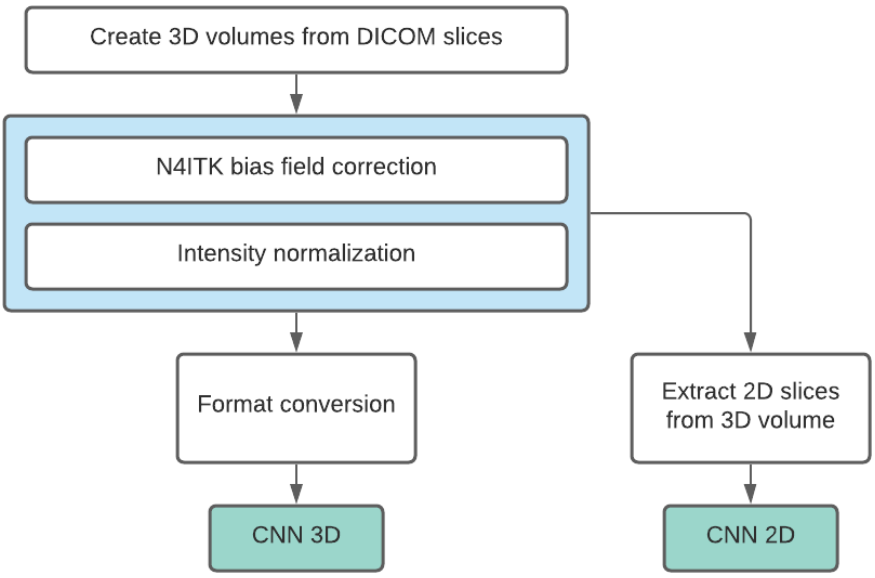


Figure 3.4: Preprocessing steps: first N4ITK bias field correction, followed by intensity normalization.

Using the pre-processed volume, two tests are carried out in parallel, the first using a 3D CNN model and the second using a 2D model, which will serve as a comparison with the performance of our proposed method.
The input of the 3D model is the set of volumes, rearranged in order to match the dimensions required as the network input.
A format conversion has been performed on Python and it is necessary to be able to make use of a function, that will be later explained, to create the Image Data Generator.
For what concerns the 2D network, inputs are two-dimensional slices extracted from each volume and the 2D slices that are output in the test phase, are then stacked together to reobtain the 3D volume.
In order to apply the Active Shape Model, it is necessary to define a set of coincident vertices in all volumes used both for training and for testing the model.
The ASM training is carried out on the volumetric labels obtained manually and consists in the creation of a Mean Shape Model and in the definition of the gray profiles and their derivatives along normals exiting points on the prostate surface.
The model is applied on the volumes obtained as output from the predict phase of the CNN 3D, therefore volumes used as training set and test set.
After applying the model, the volumetric labels are restored from the updated vertices cloud using an algorithm for 3D surface reconstruction to reobtain the volumetric binary masks.

In the results section a comparison is made by evaluating the Dice similarity coefficient and the 95% Hausdorff distance for the train and test volumes in the following cases:

- 3D CNN
- 3D CNN + ASM
- 2D CNN

Pre-processing

Pre-processing is applied in order to improve the quality of the magnetic resonance images by enhancing contrast and making the process of feature recognition easier for the algorithm.
Figure 3.4 shows our pre-processing steps.

- The first operation is called N4 Bias field Correction and it consists in removing the bias field signal that corrupts the MRI image.
  This signal, as mentioned in section 2.3.1, can lead to incorrect results if algorithms based on image gray levels are used.
- The second operation consists in normalizing the image intensities according to a technique proposed by *Nyùl and Udupa*, which works on the intensities histogram.

Further details on these two processes will be presented in separate sections in the following, together with a demonstration of their working on the dataset.
Images are finally saved in .mat format to be used by the network.

CNN 3D

A custom made network is used which works with three-dimensional volumes, based on the basic architecture of a UNet and trained from scratch.

The procedure that will be explained in detail in the chapter relative to the CNN 3D will follow these steps:

- Description of the network architecture and blocks used.
- Construction of the Image Data Generator with preparation of the images output from the pre-processing step in order to match the dimensions that the network requires in input.
- Explanation of callback functions and parameters used in the training process.
- Network training and testing.

Active Shape Model

The Active Shape Model is applied to the 3D network output to improve the segmentation along the prostate boundaries.

The proposed method consists of four steps:

- Compute a set of coherent vertices for:
  - Manual volumetric labels of the training set
  - 3D CNN output volumetric labels of the training set
  - 3D CNN output volumetric labels of the test set
- Model Training
  - Computation of the Shape Model to find the mean shape within the training set and the parameters of the transformation object.
  - Computation of the Appearance Data, or rather the gray levels and their derivatives along profiles normal to the prostate surface.
- Model Application, carried out on training and test images.
- Volumetric labels reconstruction
  Transform the vertices obtained after applying the model into two-dimensional binary masks that will be stacked together to create the volumetric label.

CNN 2D

To make a comparison with the performances of the 3D network, a basic 2D UNet model has been used, initializing the weights derived from a pre-training on the ImageNet dataset.

In the in-depth chapter on CNN 2D the following steps will be further explained:

- Data preparation and construction of the Image Data Generator with data augmentation.
- Network architecture with modification to adapt the network dimensions to our study.
- Description of callbacks and parameters used during training.
- Network training and testing.

3.3 Pre-Processing

3.3.1 N4 Bias Field Correction

Method explanation

As previously mentioned, the use of MRI as an acquisition technique presents among the various disadvantages the possibility of having a low frequency intensity non-uniformity which corrupts the images that are produced.
This inhomogeneity in the illumination is called bias field and it must be removed to avoid segmentation errors due to the impossibility of working with the real intensity values of the tissues.

The standard algorithm used in the medical field to remove bias field effects is called Nonparametric Nonuniform Intensity Normalization (N3) and it is the de facto algorithm because it is simple to use, totally automated, it does not require prior knowledge and it can be used with any image obtained with MR techniques.

This algorithm looks for the smooth multiplicative field that maximizes the high frequency content of the tissue intensity distribution.

A new algorithm, known as N4ITK [25], has been recently proposed as an alternative to N3, to improve the correction of the bias field.
The changes introduced by this algorithm concern the strategy to obtain the B-spline smoothing and the iterative optimization scheme, bringing improvements as regards the quality of the images obtained and the speed of convergence.

Approximation using B-spline is a parametric reconstruction technique used to fit B-spline objects to scattered data.
A B-spline object is defined by a set of B-splines and a grid of control points.
The traditional method for fitting involves the use of least squares fitting to approximate scattered data with B-splines, making use of a weighted regularization term to mitigate the problems of convergence instability.

The N4ITK algorithm does not use the least square fitting, and for this reason, it does not need any regularization parameter to be tuned, so it is more robust towards noise because the approximation is done locally and then all the local solutions are merged together to find the global approximation.
The B-spline approximator introduced by the N4ITK algorithm allows for better control spacing between points, guaranteeing less probability of failure even with magnetic fields

of greater intensity.

Both techniques start from the definition of the image obtained as

$$v(x) = u(x)f(x) + n(x)$$

where v is the image, u is the uncorrupted image, f is the bias field and n is the Gaussian noise.
If noise is not present, transforming the formula into logaritmic notation, we obtain

$$\hat{v}(x) = \hat{u}(x) + \hat{f}(x)$$

The iterative scheme used by the N3MNI algorithm allows to obtain the correct image at the n^{th} iteration as

$$\hat{u}^n = \hat{v} - S\hat{v} - E[\hat{u}|\hat{u}^{n-1}]$$

while for the N4ITK:

$$\hat{u}^n = \hat{u}^{n-1} - S * \hat{u}^{n-1} - E[\hat{u}|\hat{u}^{n-1}]$$

that is equivalent to

$$\hat{u}^n = \hat{v} - \sum_{i=1}^{n} f_r^i$$

N3MNI at each iteration uses the corrected image value to re-estimate the real field value.
In the N4ITK algorithm, the image corrected at the current run becomes the input of the following run.
The optimization is applied to the residual bias field and the estimate of the total field is obtained as the sum of the residual ones.
Hierarchical fitting is used to better approximate the data starting from a low-resolution B-spline and then increasing the resolution of the subsequent meshes to obtain the best-fit.

Implementation

For this study, the N4ITK algorithm has been applied using the N4BiasFieldCorrectionImageFilter class in the SimpleITK package on Python.

The algorithm usually takes an image as input and evaluates its bias field using a mask automatically defined by applying the Otsu thresholding algorithm.
To improve the performance, a binary mask (figure 3.5) has been defined with the region of interest in the input image on which to evaluate the bias field.

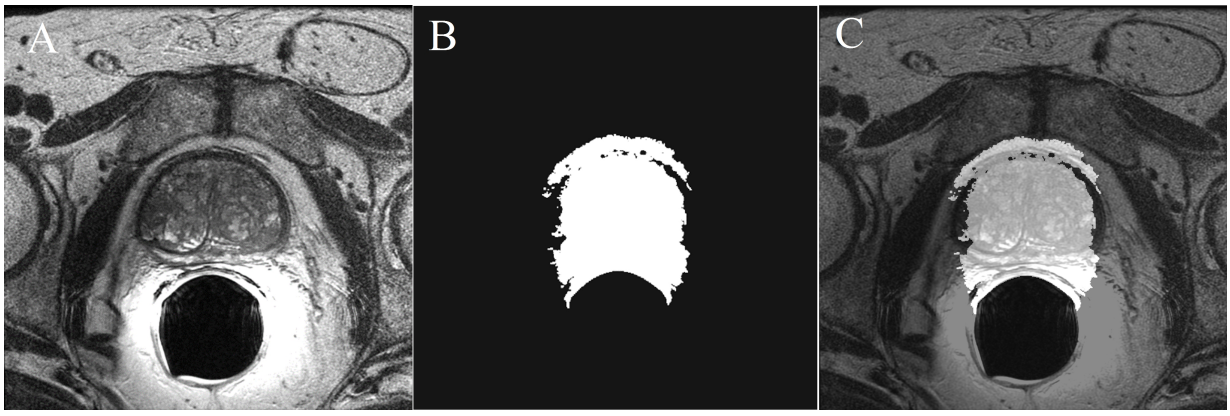


Figure 3.5: Binary mask definition for evaluating the bias field on the original image. A) original image, B) mask obtained with the proposed method, C) mask superimposed on the original image.

This mask has been obtained on Matlab with the following steps:

- Creation of a Gaussian lowpass filter of the same size of the volume and with standard deviation equal to 0.1 * volume size.
- Rescaling of the filter in the range [0 1].
- Volume filtering with the Gaussian filter.
- Binarization of the filtered volume.
- Dilation and erosion of the resulting binary mask using a morphological structuring element; in this case a sphere of radius 5.
- Keep the largest object in the volumetric label

The bias field correction has been performed inputing the volume to be corrected, loaded in float32 format, and the mask in uint8 format.
The corrected volume is then cast to int16 format and saved.

3.3.2 Intensity Normalization

Method explanation

The lack of standard image intensity scale in MRI makes it difficult to display images obtained with this modality and the subsequent image analysis.
MRI intensities have no fixed meaning because of different protocols, patients, scanners, and this can cause various problems in the segmentation phase and quantification of certain parameters.

The proposed standardization method [13] involves two stages:

- Training
  It is performed only once at the beginning; standardizing transformation parameters are learnt from a set of training images.

- Transformation
  Executed for each volume that has to be standardized; the parameters obtained in the training phase are used to map the intensity histogram of a new image into the standardized histogram.

Training

It is performed on a set of volumetric images representing the same object inside the body and obtained using the same protocol.
A volume can be identified as $\mathcal{V} = (V, g)$. This volume depends on a parameter V which represents the voxels belonging to the volume and on a parameter g which represents the intensity function for which a certain intensity value corresponds to each voxel.
The histogram of a volume is defined as $\mathcal{H} = (G, h)$, where G is a set composed of all possible gray values that a voxel belonging to V can take, and h is a function that counts all the voxels for which the intensity value of the voxels in V is equal to x $\in$ G.

If we look at the intensity histogram of an image, the high intensity tail represents artifacts and is responsible for the high variability in intensity values of the MRI image.
Before running the algorithm, three parameters which refer to the intensity histogram, have to be selected:

- p_1: minimum percentile value
- p_2: maximum percentile value
- μ: mode

In the case of the prostate, the histogram has a bimodal behavior, thus the third parameter corresponds to the second mode of the histogram, as shown in figure 3.6.

In this first training phase, these three landmarks obtained from the histogram of each image are mapped to a standard scale, transposing intensities in the range [p_1 p_2] to [s_1 s_2] linearly with the following:

$$x' = s_1 + \frac{x - p_{1j}}{p_{2j} - p_{1j}}(s_2 - s_1)$$

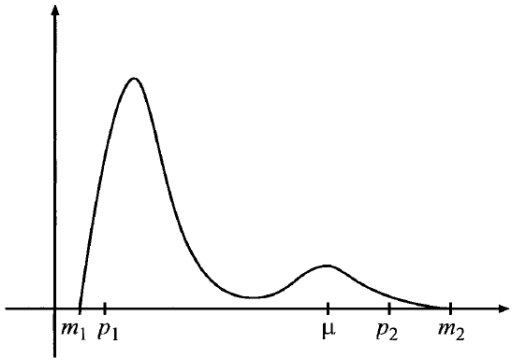


Figure 3.6: Bimodal behavior of the intensity histogram for prostate images. The second mode of the histogram μ is usually used as a histogram landmark to compute the value of μ_s, that is the mode in the standardized range. [13]

This mapping is used to determine μ_s, which is the average of all new μ_j's just found.

Transformation

In this phase for each image, the second mode of the histogram is matched with μ_s through two separate linear mappings:

- first mapping from $[p_1\ \mu]$ to $[s_1\ \mu_s]$
- second mapping from $[\mu\ p_2]$ to $[\mu\ s_2]$

Subsequently, a further mapping is carried out by transposing $[m_1\ p_1]$ to $[s_1\ s_1']$ and $[m_2\ p_2]$ to $[s_2\ s_2']$, where m_1 and m_2 are the tails of the histogram of each image. The image mapping process is shown in figure 3.7.
The transformation that allows to obtain s_1' and s_2' is called standardizer of $\mathcal{V}$ and is defined as

$$\tau_{\mathcal{V}_i}(x) = \begin{cases} \left\lceil \mu_s + (x - \mu_i)\frac{s_1 - \mu_s}{p_{1i} - \mu_i} \right\rceil & \text{if } m_1 i \leq x \leq \mu_i \\ \left\lceil \mu_s + (x - \mu_i)\frac{s_2 - \mu_s}{p_{2i} - \mu_i} \right\rceil & \text{if } m_1 i \leq x \leq \mu_i \end{cases}$$

The range $[s_1'\ s_2']$ can vary from image to image because it depends on the characteristics of the single image, however $[s_1\ s_2]$ is independent from V and in this range the uniformity of intensities can be evaluated.
In order to carry out these steps it is necessary to identify the second mode of the histogram; it is usually done with a thresholding operation using the average intensity of the image as the threshold.
The values for s_1 and s_2 must be chosen in such a way as to ensure that there is no

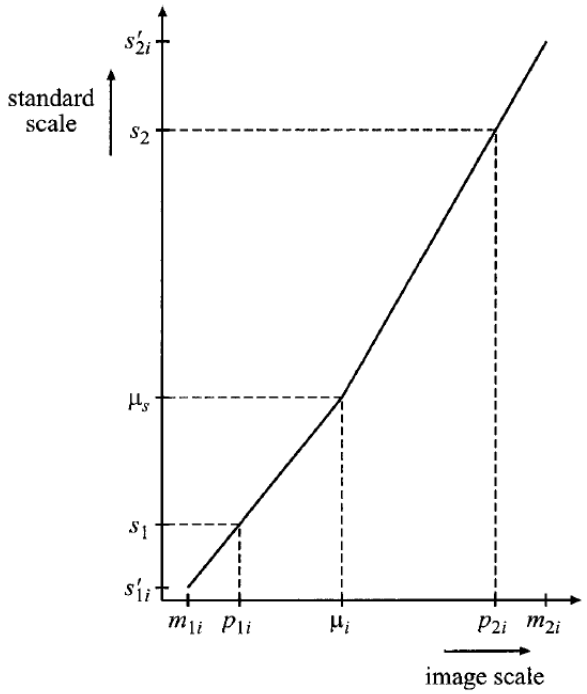


Figure 3.7: Image mapping from the original histogram space to the standardized one through linear transformations. [13]

intensity loss.
After standardization, voxels with the same intensity value are more likely to contain the same type of tissue.

Implementation

For this study, the minimum percentile value p1 has been set to 2 and the maximum percentile value to 99.5.
The minimum and maximum intensity values on the standard scale (s1 and s2) are set to 0 and 255 respectively.

The first step of the training phase consists in computing the histogram for all the images in the dataset. It has been after smoothened using a Savitzky-Golay FIR filter of order 5 and frame length 19.
Figure 3.8 shows a zoom on the rough histogram and on the filtered one, to better observe the smoothing effect of the filter.

A median filter of order $n = \frac{p2-p1}{40}$ has been then applied to the matrix containing the differences of adjacent elements within the histogram and the resulting vector is multiplied by its translated version by one unit along the main dimension.
Elements whose value was less than or equal to -0.005 have been kept.

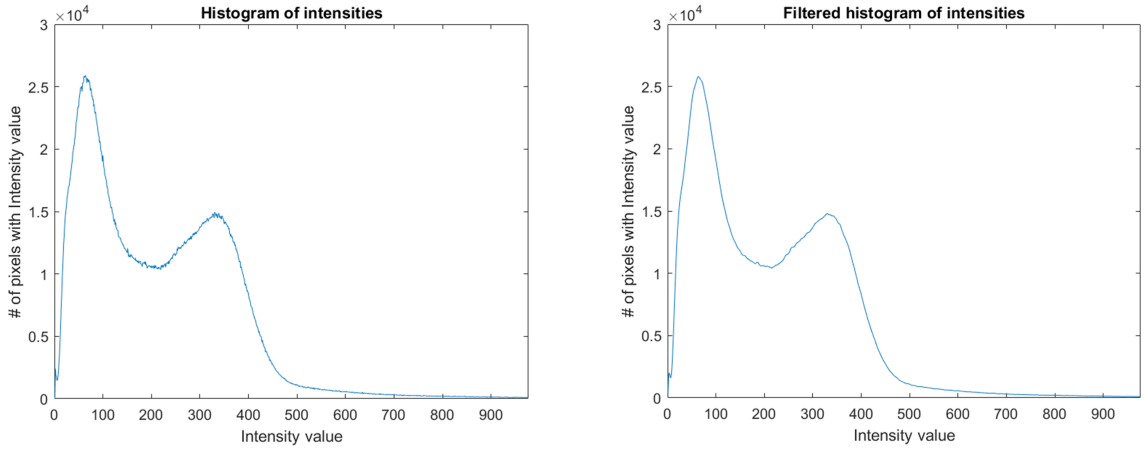


Figure 3.8: Comparison between the original histogram of an image and the filtered histogram with a Savitzky Golay FIR in a range that allows to observe the smoothing effect of the filter.

The histogram landmark chosen for all the volumes in the set instead of the second mode of the histogram is the 'valley' point between the two modes of the histogram, shown in figure 3.9.

In order to find the value for μ, a search within the result of the previous computation has to be performed, looking for the value for which all of the following conditions hold:

1. The values of the differences between successive elements of the histogram in the range $[\mu - 2 * n : \mu - 1]$ are ≤ 0.
2. The values of the differences between successive elements of the histogram in the range $[\mu + 1 : \mu + 2 * n]$ are ≥ 0.
3. μ grater than the 30th percentile with respect to the entire volume.
4. μ lower than the 70th percentile with respect to the entire volume.

Since for some prostate volumes the algorithm returned incorrect values, it has been decided to set μ equal to 150 in all cases in which this value was inconsistent with the most represented values within the dataset.

The μ' values for all prostates in the dataset have been obtained using the formula to obtain x' defined in the training phase; it is possible to obtain μ_s by finding the median of all these values.

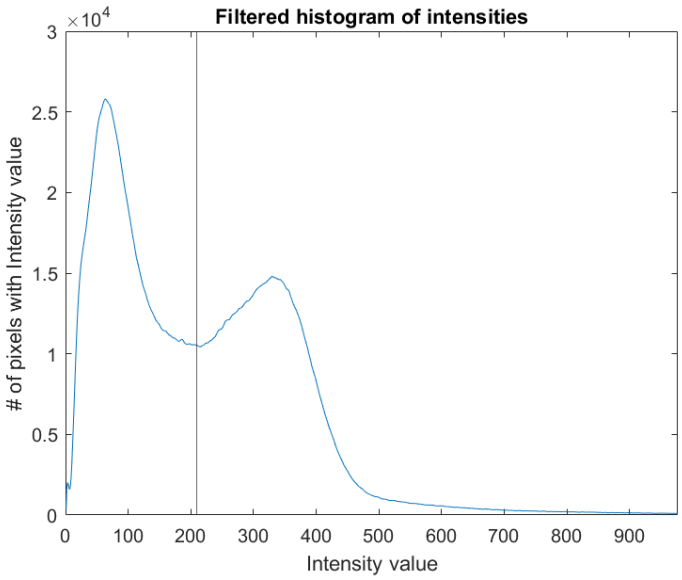


Figure 3.9: Filtered histogram of an image with the value of the computed valley highlighted.

The steps related to the transformation phase are the same. The value for m_1 and m_2 correspond respectively to the minimum and maximum values within the volume. Depending on the position in the volume respect to m_1 and m_2, each point it is translated into the new mapping creating the new image using the standardizer formula.

The results for the whole pre-processing chain are shown in figure 3.10 in axial view. We can observe the original image (figure 3.10A) with the bias field corrupting the lower part of the organ near the coil, the image after the N4 Bias Field Correction (figure 3.10B) and the final image used for the network and model training with intensities normalized (figure 3.10C).

Figure 3.10: Pre-processing results. A) Original image, B) Image after N4 Bias Field Correction, C) Image after Intensity Normalization.

3.4 3D CNN

The basic idea is to create a network inspired by the UNet architecture that, taking a prostatic volume as input, learns from it to discriminate between voxels belonging to the prostate and to the background, and outputs a volumetric label of the same size as the input.

The network discussed in this section is trained from scratch as the weights obtained for similar problems needed different input dimensions and due to the complexity and high computational burden of the volumetric problem, it has been decided to avoid re-sampling to make the dimensions compatible.

The network architecture, based on 3D UNet keras model,is shown in figure 3.11.

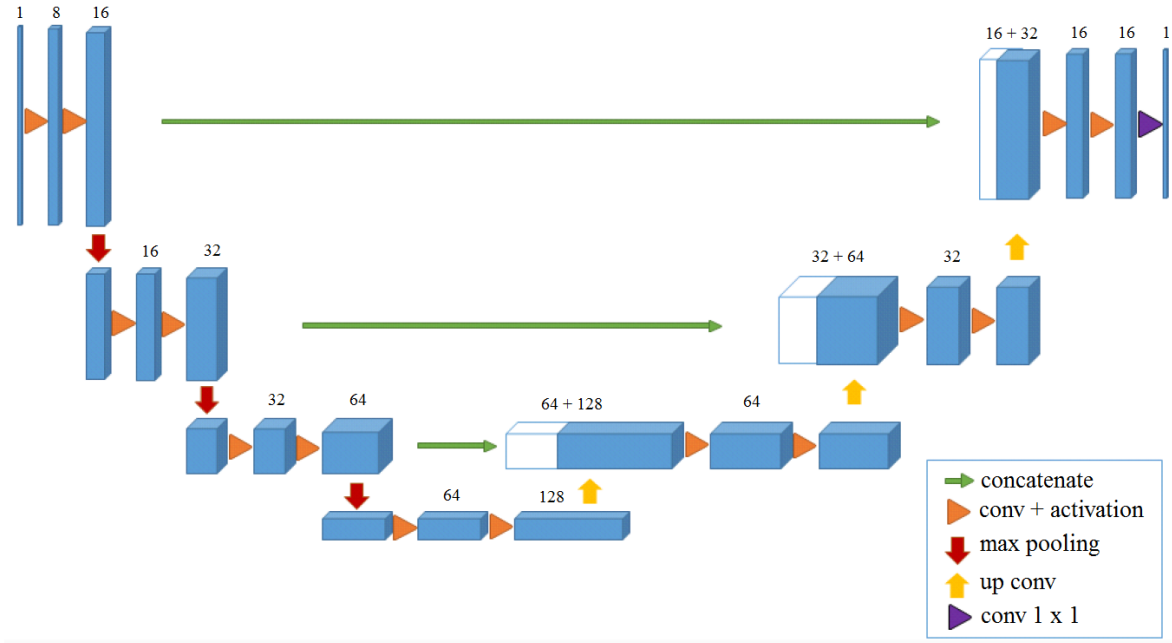


Figure 3.11: 3D UNet architecture: alternation of conv, max pooling, deconv and concatenate blocks.

We can observe that it is composed of a first shrinking path consisting of convolution and max pooling blocks to extract image features at deeper levels.
The spatial information is reconstructed in the subsequent upsampling path consisting of alternating blocks of convolution transpose and convolution in order to restore the original dimensions.

Table 3.1 lists the different layers in detail with the relative tensor output dimensions and learnable parameters at various stages. The blocks composing the architecture will be better detailed in the next section (3.4.1).

Layer Type	Output Shape	Parameters
Input Layer	(None, 1, 24, 512, 512)	0
Conv3D	(None, 8, 24, 512, 512)	224
Activation	(None, 8, 24, 512, 512)	0
Conv3D	(None, 16, 24, 512, 512)	3472
Activation	(None, 16, 24, 512, 512)	0
MaxPooling3D	(None, 16, 12, 256, 256)	0
Conv3D	(None, 16, 12, 256, 256)	6928
Activation	(None, 16, 12, 256, 256)	0
Conv3D	(None, 32, 12, 256, 256)	13856
Activation	(None, 32, 12, 256, 256)	0
MaxPooling3D	(None, 32, 6, 128, 128)	0
Conv3D	(None, 32, 6, 128, 128)	27680
Activation	(None, 32, 6, 128, 128)	0
Conv3D	(None, 64, 6, 128, 128)	55360
Activation	(None, 64, 6, 128, 128)	0
MaxPooling3D	(None 64, 3, 64, 64)	0
Conv3D	(None, 64, 3, 64, 64)	110656
Activation	(None, 64, 3, 64, 64)	0
Conv3D	(None, 128, 3, 64, 64)	221312
Activation	(None, 128, 3, 64, 64)	0
Conv3DTranspose	(None, 128, 6, 128, 128)	131200
Concatenate	(None, 192, 6, 128, 128)	0
Conv3D	(None, 64, 6, 128, 128)	331840
Activation	(None, 64, 6, 128, 128)	0
Conv3D	(None, 64, 6, 128, 128)	110656
Activation	(None, 64, 6, 128, 128)	0
Conv3DTranspose	(None, 64, 12, 256, 256)	32832
Concatenate	(None, 96, 12, 256, 256)	0
Conv3D	(None, 32, 12, 256, 256)	82976
Activation	(None, 32, 12, 256, 256)	0
Conv3D	(None, 32, 12, 256, 256)	27680
Activation	(None, 32, 12, 256, 256)	0
Conv3DTranspose	(None, 32, 12, 256, 256)	8224
Concatenate	(None, 48, 24, 512, 512)	0
Conv3D	(None, 16, 24, 512, 512)	20752
Activation	(None, 16, 24, 512, 512)	0
Conv3D	(None, 16, 24, 512, 512)	6928
Activation	(None, 16, 24, 512, 512)	0
Conv3D	(None, 1, 24, 512, 512)	17
Activation	(None, 1, 24, 512, 512)	0

Table 3.1: Network layers with layer type, output shape and learnable parameters.

3.4.1 Network Architecture

One of the first parameters that have to be defined when building the neural network, is its depth, which defines the number of max pooling layers that the network will have. Several tests have been carried out by changing this parameter and a depth of 4 has been chosen, since using a higher depth would have brought to higher memory amount required for training.

The network presents an alternation of blocks of 3D convolution, 3D max-pooling, activation and 3D deconvolution, which will be briefly discussed below.

- 3D Convolution
  It can be defined as a spatial convolution over volumes.
  This layer creates a convolution kernel which is convolved with the input layer to produce an outputs tensor.
  The kernel size, a list representing the depth, height and width of the 3D convolution window, is set before training to (3,3,3).
  The stride is another parameter that must be set before the training phase, and manages how much the convolution window moves along each spatial dimension (set to (1,1,1)).
  The padding for convolution has been set to 'same', i.e. it pads all sides of the volume equally, so that the output has the same size as the input.
  The number of filters used by the first layer has been set to 8; the additional layers will contain a number of filters that is a multiple of this number.

- 3D Deconvolution
  It can be applied in two different ways:

  - 3D up-sampling: only resizes the image by copying the pixels as many times as necessary; the only parameter that has to be specified in the size.
  - Transpose Convolution (Deconvolution)
    It is the opposite of the convolution block, it needs filters, kernels, strides and all the parameters that the convolution blocks needs. For this reason, it is more complicated, it increases the memory required for training a little.

  For this specific study, the Transpose Convolution has been used with kernel size set to (2,2,2) and stride set to (2,2,2).

- Activation
  This block applies an activation function to an output. The activation function must be chosen based on the task to be faced.

In our study binary classification is the final task, for this reason the last output layer consists in a sigmoid activation block.

- Max-pooling 3D
  The max pooling operation reduces the size of the input tensor by taking the maximum value for each pooling window applied to the image.
  The pool size determines the factor by which the input is down-scaled; in this case it is equal to (2,2,2), which means that the 3D input is halved in each of the three dimensions.

- Final Convolution
  It is a 3D convolution that depends on number of labels in which the elements belonging to the volume are classified.
  The kernel size has been set to (1,1,1).

The final phase of creating a model is called model compilation and it is necessary for the training to begin.
Compilation is performed using a method provided by Keras. This method requires the definition of three parameters:

- Loss function
  It is a function defined to find the error during the learning process.

- Optimizer
  It is used to optimize the input weights by comparing the predicted output with the loss function.
  The optimizer used in this study is known as 'Adam optimizer' and it consists in using estimations of first and second moments of gradient to adapt the learning rate for each weight in the network.

- Metrics
  A metric is a function used to evaluate the performance of the model. At the end of each epoch during the training process, the metric values are recorded for both the training set and the validation set.

Different tests have been carried out changing the loss function and the metrics. More details on the implementation and results will be given in the section relative to the procedure (subsection 3.4.4).

3.4.2 Custom Image Generator

The input of the network, as seen in section 3.4.1, is a tensor of dimensions $(\text{None}, 1, 24, 512, 512)$.

The volumes in .mat format coming out of the preprocessing step, have been resized using various models of Image Generators to obtain the correct dimensions for the input. However, all these tests resulted in an out-of-memory state as the memory required to load all volumes and train a sufficient number of parameters during the training process was too high.

This problem has been solved by loading the dataset using a function that allows to bypass the computational load due to the size of the input volumes, loading the directory into memory instead of the volumes.
If 2D networks developed with the basic Tensorflow packages are used, this function is present in the Keras package and allows to iterate within the directory where data is present to obtain the image generator to train the network.

Since this is a custom made 3D network, in order to use the directory iterator, some changes must be made to both the format of the volumes in the directories and the iterative functions.

The basic iterator needs as a parameter the directory from which to read the images. Each subdirectory in this directory must contain images belonging to a class.
In this study, since the purpose is binary classification, the part related to classes has been ignored, collecting all the images in a subdirectory and creating two separate generators, one for the volumes and one for the corresponding volumetric labels.

Images to be uploaded in this way must have a specific extension.
For this reason, it has been decided to convert the preprocessing output volumes into one of the permitted formats, specifically the .tiff extension has been chosen since it allows to work with three-dimensional data.

Within the function that manages the conversion, volumes are subdivided in two sets:

- Training set: used to train the network.
- Validation set: used to validate the network performances during training.

It has been decided to use training set $= 0.8 *$ validation set as the proportion, therefore 36 volumes are used for training and 9 for validation.

In order to have images consistent with the chosen format, the volumes in .mat format have been transposed, so as to finally have a dimension of 24 x 512 x 512.

Initially all images in the directory are listed and the ones with the correct extension, in our case the tiff files, are iterated and their relative paths are saved.
Starting from these filenames, the transformed samples batches are built. We have chosen to use batch size equal to 1 for memory requirements, that is one image processed at a time.

Images are loaded in the form of PIL instances, from which the 3D Numpy array is extracted.
Tiff files contain the number of frames in the header; the 3D array is constructed by loading the individual slices of the tiff file and stacking the frames together in float32 format. The array is arranged in such a way as to respect the chosen settings for what concerns the channel ordering. For this study *'channel first'* has been selected, so the number of image channels is the first value in the array and is equal to 1 since gray scale images are used.

The standard Image Data Generator has not been used because data augmentation has not been performed, as 3D volumes generally have sufficient spatial information.

Four directory iterators have been built, two for training and validation images and two for the respective volumetric labels.
Class mode has been set to *'None'* because, as mentioned before, binary classification is among the voxels of the image, each of which can be classified as backgroud (class 0) or prostate (class 1).
The iterators corresponding to training and validation are zipped separately in order to have two single generators with image and corresponding mask.

For the training phase of the model, the number of steps that the algorithm performs per iteration is set as:

$$\text{number of training steps} = \frac{\text{samples in training generator}}{\text{training batch size}}$$

$$\text{number of validation steps} = \frac{\text{samples in validation generator}}{\text{validation batch size}}$$

with training batch size = validation batch size = 1.

3.4.3 Callbacks and network training

A callback is an object used to perform actions in different stages of the training process. Two types of callbacks have been used for this study:

- Model Checkpoint
  It is a type of function used to save the model or model weights at certain points, so that training can be restarted from the saved state at a later time by loading the model or the weights at that point.
  The variable to be monitored is the loss defined during the compilation of the model.

- Early Stopping
  It is a callback function that allows to stop training based on a variable being monitored.
  The variable is a metric defined in the compilation; when it stops improving, the training is stopped.
  Parameters for the early stopping criterion must be set, depending on the training goal. If the accuracy of the network has to be monitored, the *'mode'* parameter has to be set equal to *'max'*, so that the training will be interrupted when the monitored quantity stops improving.
  On the contrary, if the quantity to be monitored is the loss, the mode has to be set equal to *'min'* and the training is stopped when the quantity stops decreasing.

  The change of the variable is evaluated on the basis of a parameter called *'min delta'*, which corresponds to the minimum change in the monitored quantity that corresponds to an improvement.
  Training is interrupted if the improvement stays under this value for a number of epochs equal to the *'patience'* parameter.
  In our case *'min delta'* has been set to 0.01 and *'patience'* to 25 epochs.

This list of callbacks is passed to a function known as *'fit generator'* which allows to train the model on data that is generated using the iterator previously introduced.
This function has been used instead of the *'fit'* method because we are dealing with datasets huge in size and harder to fit into the computer memory.
What this function does is take one batch at a time from the dataset and perform back-propagation on it and then update the model weights.
The process is repeated for the predefined number of epochs, which is also passed as a parameter and corresponds to 50 in our case.

3.4.4 Parameters Optimization

The first tests have been carried out on Python using a virtual environment and the packages necessary to start the training on GPU.
The chosen settings for the network parameters are:

- Network depth: 4
- Number of base filters: 8
- Total number of trainable parameters: 1.192.593
- Loss: Dice similarity coefficient loss
- Metric: Dice similarity coefficient

These parameters have been chosen after carrying out various tests.
Initially the number of base filters was set to 32, obtaining a total number of trainable parameters equal to 19.068.993.
In order to limit the amount of memory required for the training, the number of trainable parameters has been limited to the last layers of the network, thus freezing a large part of the first ones.
Performing the tests using the GPU/CPU of a standard computer, however powerful it may be, brought the maximum number of trainable parameters, given this initial setting, to a very low number in our case equal to 65.

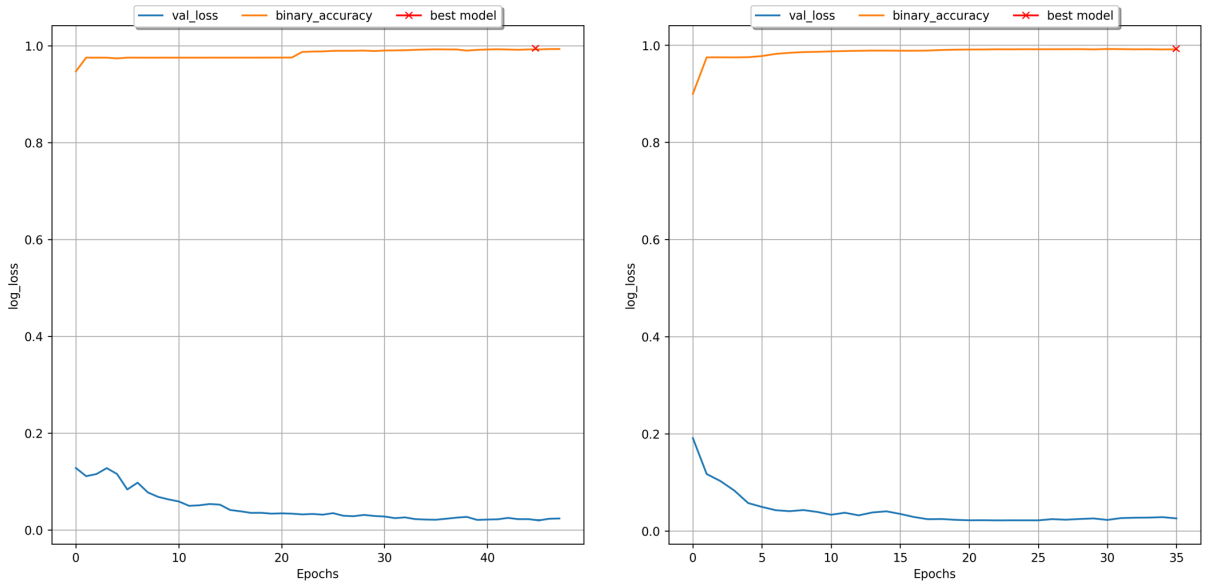
It has been therefore decided to train the network using Kaggle, a subsidiary of Google LLC which provides free access to NVidia K80 GPUs in kernels and 13 GB of RAM.
Tests have been carried out initially maintaining the previous setting of parameters, and subsequently making small changes.
The online cloud computing provided by Kaggle let us train the network up to 442.561 parameters in a reasonable time.
To take advantage of the possibility of training a network in it entirety, therefore without freezing any layer, the number of base filters has been changed to 8.
In this way the total number of parameters of the network is equal to 1.192.593 and it is possible to train them all without incurring the out-of-memory condition.

The first test have been performed using binary crossentropy and binary accuracy respectively as loss and metric. The model has been compiled in two different ways to test the influence of the initial learning rate on the Adam optimizer.
We have test the following two cases:

- No learning rate specified $\rightarrow$ use the default 0.001

- Specify learning rate equal to 0.0001

The learning curves obtained for the two cases are shown in figure 3.12, with the validation loss shown in blue and the binary accuracy in orange.
In correspondence with the red cross marker, we have the binary accuracy value for the best model saved.



(a) Binary accuracy with default learning rate, equal to 0.001.

(b) Binary accuracy with learning rate equal to 0.0001.

Figure 3.12: Comparison between the learning curves obtained by training the network with binary cross-entropy as loss and binary accuracy as metric with initial learning rates equal to 0.001 and 0.0001.

We can observe that the early stopping criterion led the network to interrupt the training by convergence at the 48^{th} epoch in the first case and at the 36^{th} in the second case.

A problem that occurs when dealing with volumes in which the object to be identified occupies a small part with respect to the background, is the fact that very high levels of accuracy can be achieved since the number of correctly classified pixels takes into account the large number of pixels belonging to the background. Two ways have been tried to solve this problem:

- Use the class weights parameter to be passed during the model fit phase to assign a greater weight to the pixels corresponding to the object to solve class imbalance.
- Use a loss and a metric that take into account the pixels of the object and not those of the background.

In order to use the class weighting in a voxel classification problem on three dimensional data, it is necessary to create a volume of the same size of the volumetric label during

the Image Generator definition phase.

This volume must contain a weight value to be assigned to each single voxel.
This value can be obtained by counting the number of voxels representing background and the object within the entire dataset and then evaluating

$$\text{background} = \frac{\text{background}}{\text{background} + \text{object}} * 100$$

$$\text{object} = \frac{\text{object}}{\text{background} + \text{object}} * 100$$

$$\text{class weight} = \frac{\text{background}}{\text{object}}$$

and assigning 1 as weight to the voxels belonging to the background class and the class weight just calculated as weight to the voxels belonging to the object class.

A test has been carried out using this method, which however led to a problem of high computation and memory required, as to perform the voxel-wise multiplication in the weighting phase during training, the volume is rendered 1D obtaining a huge number of values to be computed for each image, equivalent to 24 x 512 x 512.

The second possible way has been therefore tried, looking for a loss function and a metric that is more suitable for our situation.
The dice coefficient and the corresponding dice loss have therefore been implemented as:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

$$DL = 1 - DSC$$

with X and Y being the flattened versions of the true volumetric label and of the predicted volumetric label respectively, and are passed as loss and metric when compiling the model.

Two tests have been carried out using two different learning rates, equal to 0.0001 and 0.00001.

The resulting learning curves are shown in figure 3.13. We can observe how the curve relative to the dice coefficient reaches higher values using a learning rate of 0.0001.

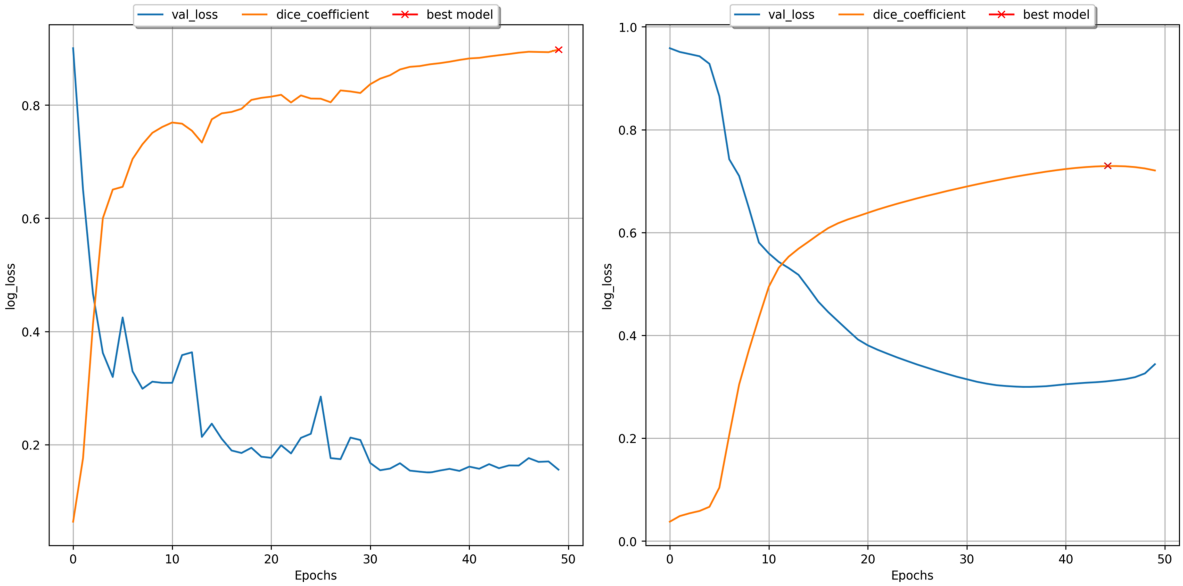


Figure 3.13: Comparison between learning curves obtained for a training using dice similarity coefficient loss as loss and dice similarity coefficient as metric with initial learning rates equal to 0.0001 and 0.00001.

To evaluate if the performances could improve over a higher number of epochs, this parameter has been set equal to 100.

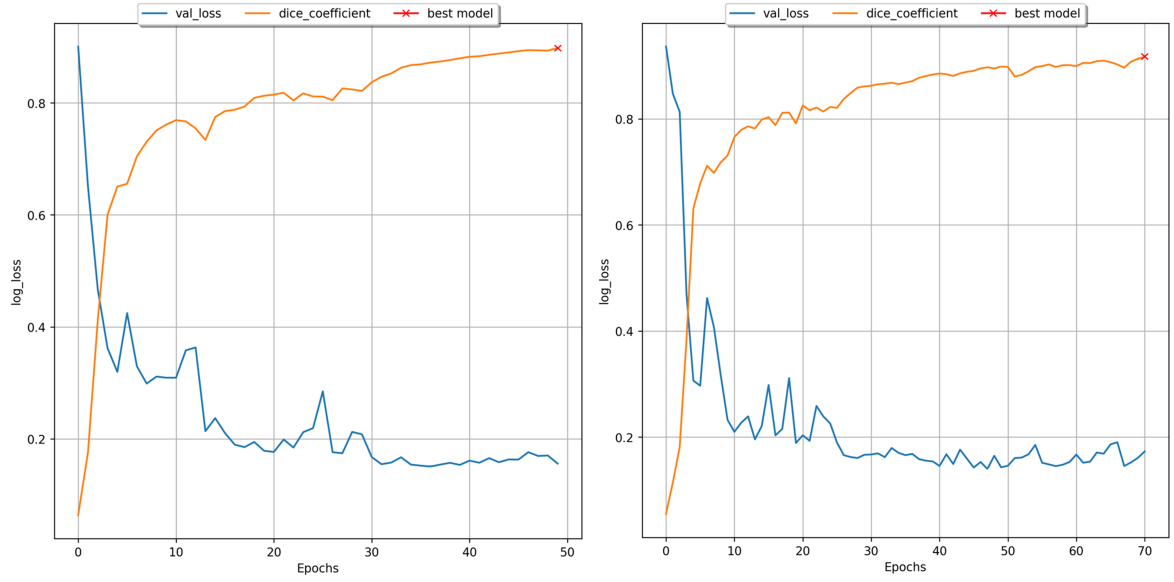


Figure 3.14: Comparison between learning curves obtained training the network with dice similarity coefficient as metric and dice similarity coefficient loss as loss for 50 epochs on the left and 100 epochs on the right.

We can see in figure 3.14 that the resulting curve is very similar to the previous one, so 50 epochs have been considered enough for training.

3.5 Active Shape Model

3.5.1 Coherent vertices computation

The first step necessary for the creation of a mean shape model is the determination of a set of coherent vertices for all the prostates belonging to the training set (manual labels and 3D CNN output) and to the test set (3D CNN output).
The selected landmarks must match in each volume so that they can be aligned in the next step to determine the average model and to further apply it on the outputs of the 3D CNN.
All the following operations and the evaluation of the model are performed on the volumes after applying an isotropic upsampling operation to have the same resolution, set to 0.5 x 0.5 x 0.5 mm on the three axes. The three-dimensional contours of each volumetric label are obtained by eroding the volume with a matrix of dimensions $3x3$ made of logical ones that allows to obtain the outer edges.

3D-surface fit

The external surface of each prostate is fitted with a three-dimensional standard surface in order to determine the key points to be taken as a reference in the following steps for each prostate volume. These key points can be for example the origin of a spherical surface used to approximate the volume, or in the case of an ellipsoidal fit, points obtained from the foci on different axes.
Two tests have been carried out for this study, the first of which involves the approximation using a spherical surface, while the second an ellipsoidal surface. The results are shown in figure 3.15.

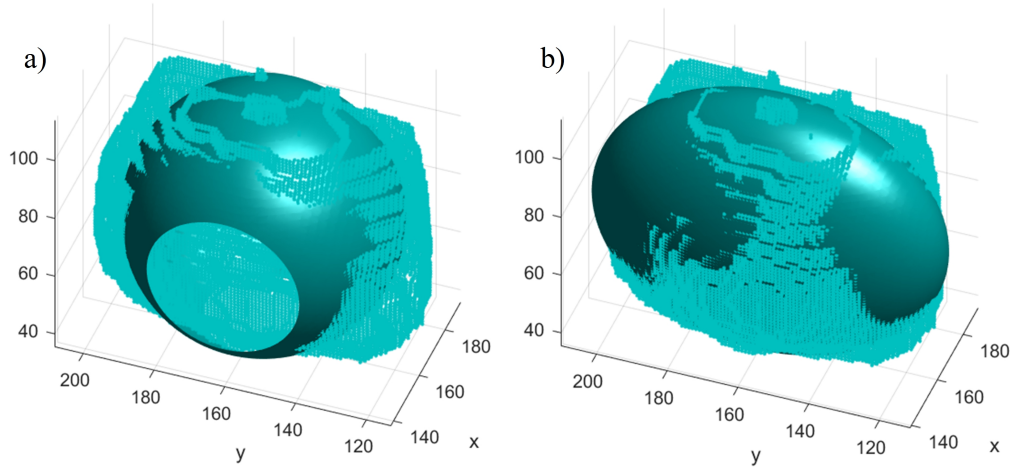


Figure 3.15: Comparison between surface approximation using a sphere (a) and an ellipsoid (b). The ellipsoidal surface better approximates the prostate shape.

As can be seen, the shape of the prostate is best approximated by using the ellipsoidal surface, so four key points have been selected starting from this fit in the x-y plane, following two modalities.

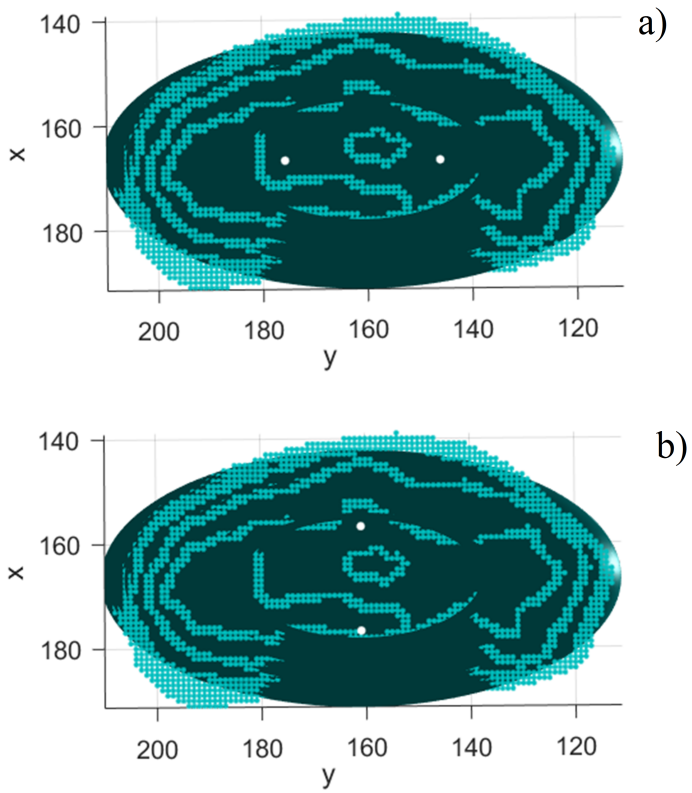


Figure 3.16: Prostate surface and ellipsoidal approximation in the x-y plane with the chosen four points on the major (a) and minor (b) axes.

- For what concerns the major axis in the x-y plane (figure 3.16a), a parameter c has been calculated as:

$$c = \sqrt{radius_{max}^2 - radius_{min}^2}$$

and the following two points have been chosen, starting from the center of the ellipsoid:

$$F_1 = [x_{center}, y_{center} - \frac{c}{2}, z_{center}]$$

$$F_2 = [x_{center}, y_{center} + \frac{c}{2}, z_{center}]$$

The half of c has been used instead of c because the latter, in some cases, exits the prostate boundaries, thus causing erroneous results.

- For the minor axis, the parameter c assumes the value:

$$c = \sqrt{radius_{min}^2}$$

and the chosen points (figure 3.16b) are:

$$F_1 = [x_{center} - \frac{c}{2}, y_{center}, z_{center}]$$

$$F_2 = [x_{center} + \frac{c}{2}, y_{center}, z_{center}]$$

Triangulation

Starting from the 3D vertices obtained in the first step, a triangulation algorithm, known as Alpha Shape Triangulation has been used to divide the 3D surface into a variable number of triangles, that are necessary to compute the triangle-ray intersection in the following section.
An Alpha Shape is a family of linear curves defined in the Euclidean plane, whose purpose is to approximate the shape of a finite set of points.
The Alpha Shape [14] derives from the boundary of an Alpha-complex, a derivative of the Delaunay triangulation.
The shape obtained for a specific set of points depends on a parameter called α that defines the level of refinement of the structure, which ranges from a very coarse shape approximating a convex hull to a shape that closely approximates the points of the set.
The level of refining that determines the number of points used to approximate this surface inversely depends on the α parameter, therefore, the smaller it is, the finer the fit.

In our study, an α parameter equal to 50 has been chosen to suit all shapes belonging to the training set (manual annotations and 3D CNN output) and to the test set (3D CNN output) in order to obtain closed surfaces for all the shape cases.
Figure 3.17 shows an examples of the final triangulation (patient 101).

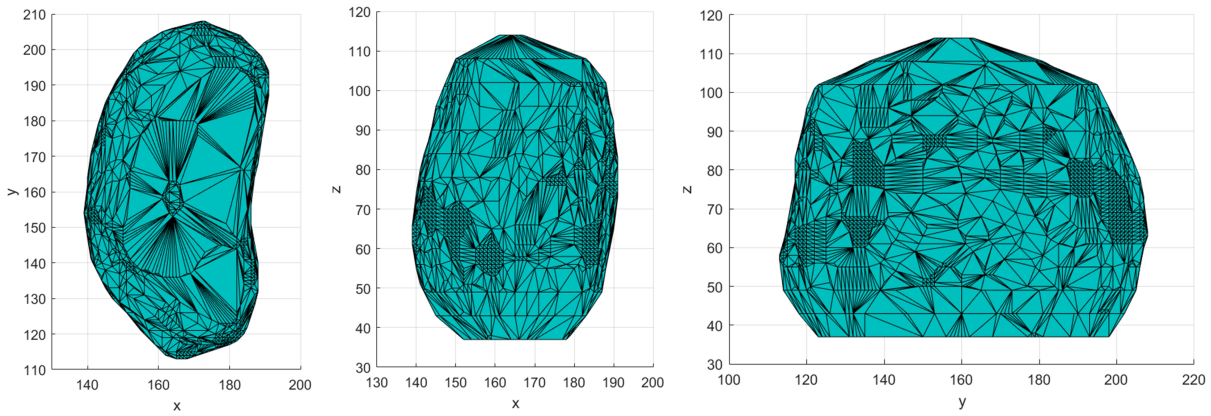


Figure 3.17: Alpha Shape for patient 101 in the x-y, x-z and y-z planes.

The boundary facets of the shape have been computed together with the vertex coordinates that make up the boundaries.

The triangles that make up the alpha shape have been extrapolated by connecting the vertices belonging to a single face of the triangulation and saved in a $3x3$ matrix, containing the x, y and z coordinates for each of the three points that determine the single triangle.

Ray-Triangle intersection

In order to obtain corresponding points in each prostate, we have used the Moller-Trumbdore [11] algorithm to compute the intersection between a set of rays originating in each of the key points found in the previous step and each of the triangles that make up the triangulation.
The elements involved in the algorithm are:

- A ray defined as
$$R(t) = O + tD$$
where O is the origin of the ray and D is a normalized direction.

- A triangle defined by the three vertices V_0, V_1 and V_2.
A point within this triangle is defined by:
$$T(u, v) = (1 - u - v)V_0 + uV_1 + vV_2$$
where (u, v) are the barycentric coordinates.

Finding the intersection is equivalent to finding the values of t, u and v for which $R(t) = T(u, v)$.
Rearranging the equivalence gives the linear system of equations:

$$[-D, V_1 - V_0, V_2 - V_0] \begin{bmatrix} t \\ u \\ v \end{bmatrix} = O - V_0$$

The solution is obtained using the Cramer's rule:

$$\begin{bmatrix} t \\ u \\ v \end{bmatrix} = \frac{1}{|-D, E_1, E_2|} \begin{bmatrix} |T, E_1, E_2| \\ |-D, T, E_2| \\ |-D, E_1, T| \end{bmatrix}$$

which can be rewritten, keeping in mind that $|A, B, C| = (A\ x\ B) \cdot C = -(A\ x\ C) \cdot B = -(C\ x\ B) \cdot A$, as:

$$\begin{bmatrix} t \\ u \\ v \end{bmatrix} = \frac{1}{(D\ x\ E_2) \cdot E_1} \begin{bmatrix} (T\ x\ E_2) \cdot E_2) \\ (D\ x\ E_2) \cdot T \\ (T\ x\ E_1) \cdot D \end{bmatrix} = \frac{1}{P \cdot E_1} \begin{bmatrix} Q \cdot E_2 \\ P \cdot T \\ Q \cdot D \end{bmatrix}$$

where $P = (D\ x\ E_2)$ and $Q = (T\ x\ E_1)$. For this study, the number of rays chosen to have a sufficient amount of points in the final shape has been obtained by using the main normalized directions

- x → [1 0 0]
- y → [0 1 0]
- z → [0 0 1]
- xy → [1 1 0], [1 -1 0]
- xz → [1 0 1], [1 0 -1]
- yz → [0 1 1], [0 -1 1]
- xyz → [1 1 1], [-1 1 -1], [-1 -1 1], [-1 1 1]

and in addition the same directions scaled by 2, 4 and 8 and the corresponding products. The final number of rays is represented in figure 3.18, superimposed on the Alpha Shapes.

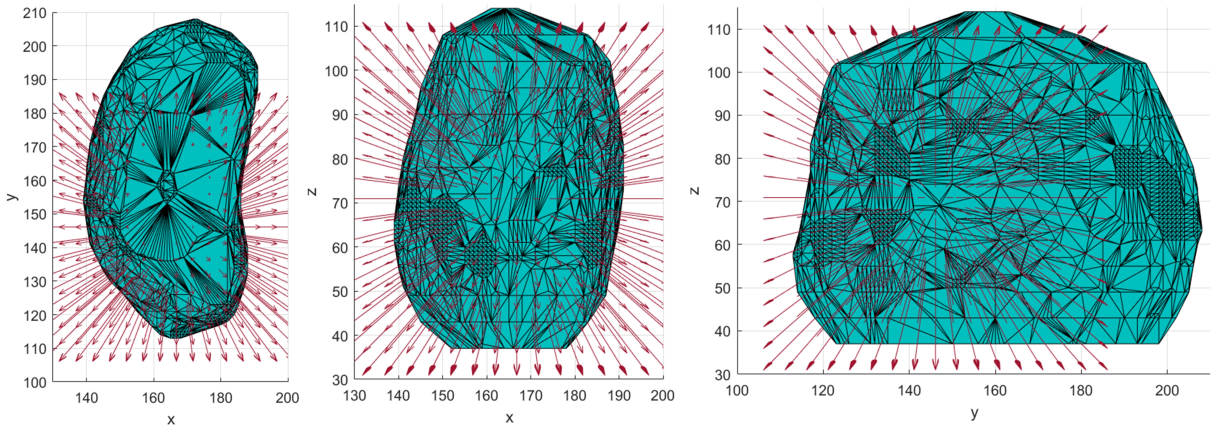


Figure 3.18: Rays intersecting the Alpha Shape surface for patient 101 in x-y, x-z and y-z planes.

The intersections are found assigning the four key points as the origin of the rays one at a time, scrolling through the matrix of directions and looking for the intersection between

each ray and each triangle making up the triangulation.
Directions in which no intersections are found are eliminated from the total matrix to ensure that the points remain corresponding in each prostate.

Check Mismatch

The triangulation used to define the three-dimensional surface can introduce errors in finding intersection points, depending on how many triangles are used to define the surface and where they are positioned in space.

To ensure consistency between the points found for each prostate in each direction, the space around each key point is initially divided into eight quadrants.
Normally, if the triangulation allows to obtain a closed volume, for each ray an intersection should be found for each of the two directions (positive and negative respect to the origin).
If the prostate is not closed, as can be observed in figure 3.19, it is possible that the points of intersection are found in only one direction, and therefore, in only one of the quadrants previously discussed.

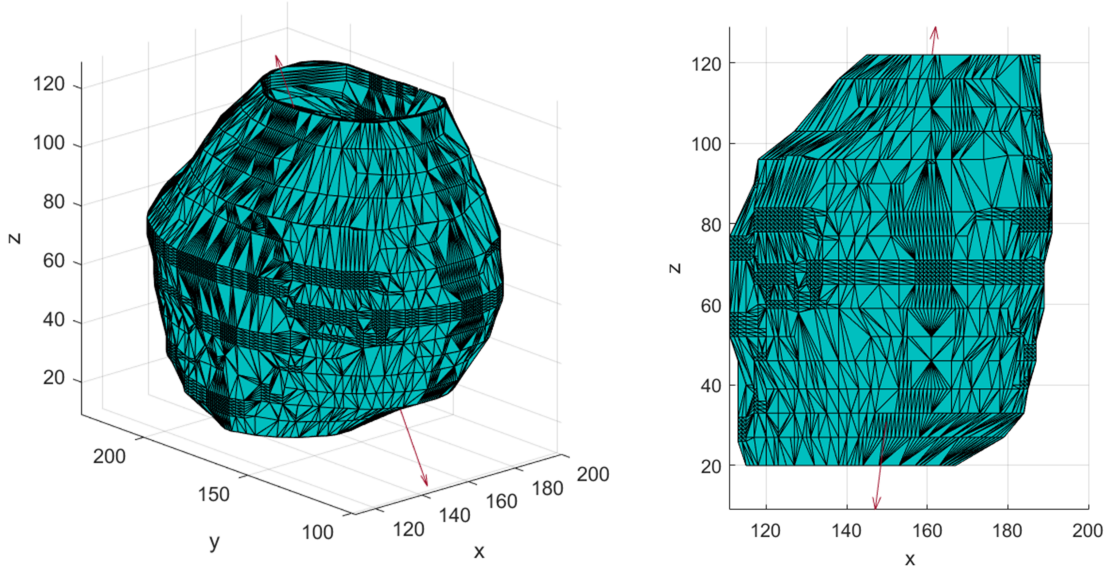


Figure 3.19: Example of an open surface. In the upper part of the prostate rays exit the surface without intersecting, while in the lower part intersections are found.

If this happens, a flag is set to one, which allows to consider for the specific direction only points belonging to that quadrant, for all the prostates in the training set.
If this situation arises for more than one prostate, it is necessary to check the quadrant where points are found; in the case for a specific direction a point is find in a prostate in quadrant 1 and in another prostate the point is found in the opposite quadrant, the

direction in which this occurs is removed.

Find Vertices

After having removed the directions in which no consistent quadrants are found within the different prostates, for each direction the intersection points are calculated as:

$$\text{intersection} = \text{origin} + t * \text{direction}$$

For all these points the quadrant to which they belong is determined and in each of these quadrants, the Euclidean norm of the points with respect to the origin of the rays is determined. The norms are sorted and for each quadrant the points with the highest distance are selected.

Figure 3.20 shows an example of vertices computation for a specific direction. The red cross represents the origin of the ray connecting the scattered points in the image.
The selected points placed on the outer surface of the prostate volume have been filled in order to demonstrate the algorithm behavior.
If the flag defined in the previous step is equal to 1, only points belonging to the corresponding quadrant are taken into consideration.

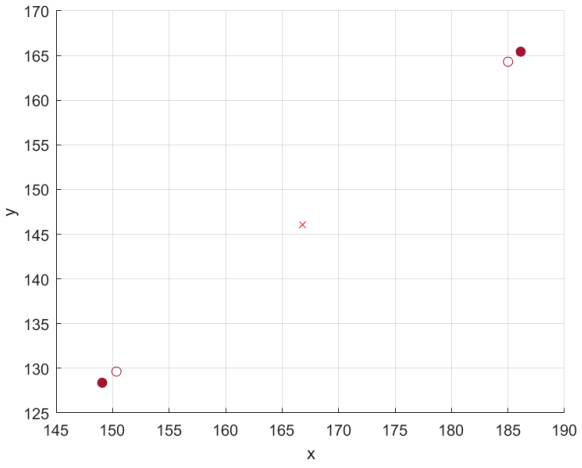


Figure 3.20: Vertices found in one direction with the proposed algorithm.

Figures 3.21 and 3.22 show two examples of vertices matrices found with this methodology in the xyz plane and zooming in x-y, x-z and y-z plane respectively.
The two prostates are different from each other, both in terms of shape, alignment and rotation, but the points found are quite corresponding in both examples.

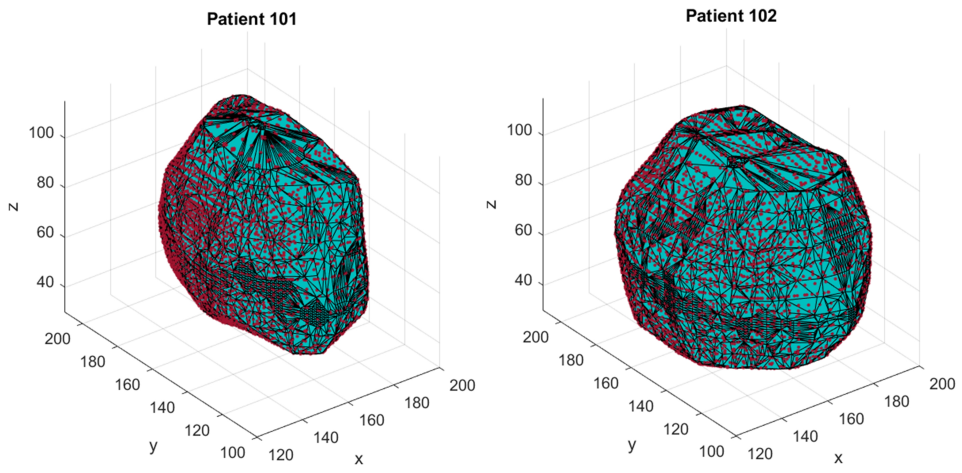


Figure 3.21: Vertices comparison in xyz plane for patients 101 and 102.

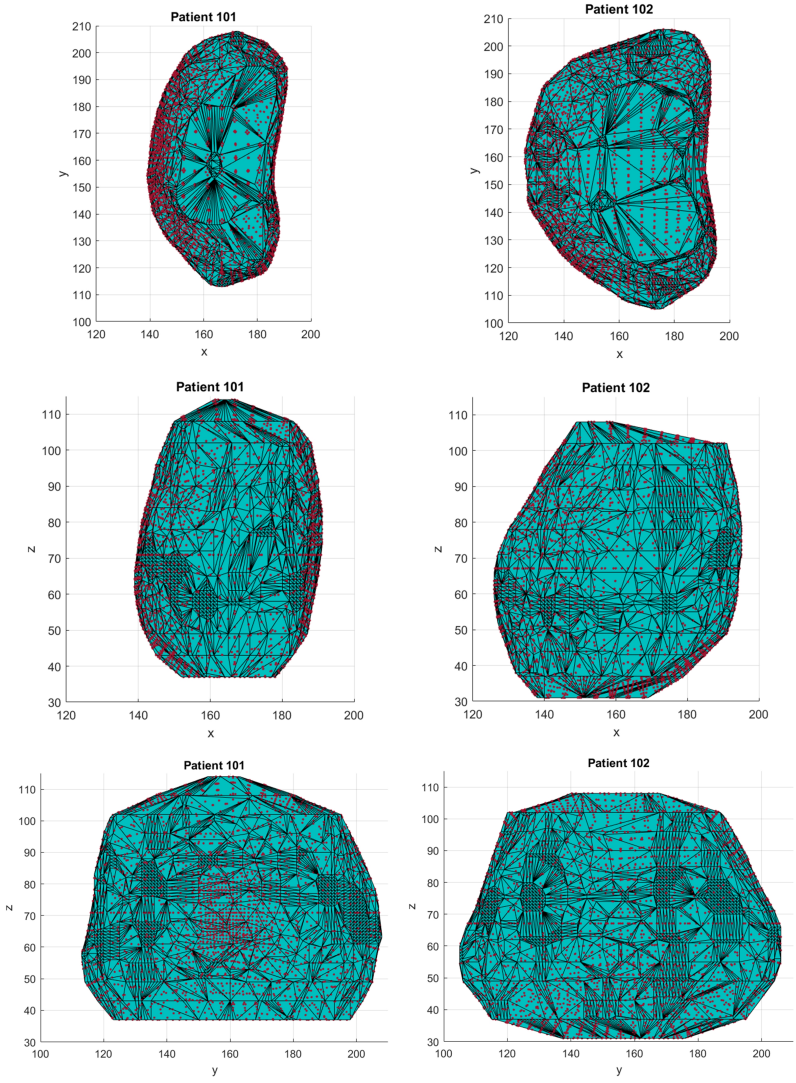


Figure 3.22: Vertices comparison in x-y, x-z and y-z planes for patients 101 and 102.

3.5.2 Shape Model

The vertices obtained in the previous step for the manual labels must be aligned to create the shape model.
The model input is a cell array containing the matrices representing the vertices for all manual annotations in the training set.

The mean vertices matrix is initialized with the value of the vertices of the first prostatic volume.
The contour positions of the other volumes are aligned, centered and any rotation is removed with the following operations.

1. Remove Translation

   Data are centered by removing the offset value for each prostate.

   $$\text{offset} = -\text{mean}(\text{vertices})$$

   $$\text{vertices} = \text{vertices} + \text{offset}$$

2. Correct Rotation

   The correction is performed using the Four-Quadrant Inverse Tangent in two steps, firstly removing the angle in the x-y plane and after removing the angle in the y-z plane.
   Figure 3.23 shows the range of values returned by the Four-Quadrant Inverse Tangent, depending on the values of x and y [-π π].

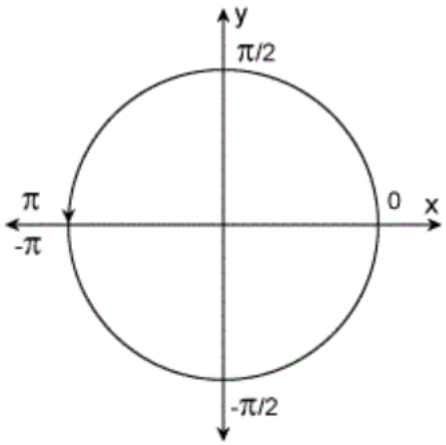


Figure 3.23: Four quadrant inverse tangent range [-π +π].

In the x-y plane, the angle for the current volume points and for the mean vertices corresponding to the first volume, are obtained as:

$$\text{rot} = \text{atan2}(y, x)$$

$$\overline{rot} = \text{atan2}(\bar{y}, \bar{x})$$

The mean angle in the x-y plane is obtained as follows.
First the offset angle in the x-y plane is computed as:

$$\text{offset}_{xy} = \overline{(\overline{rot} - rot)}$$

and subtracted from the current angle

$$\text{rot} = \text{rot} + \text{offset}_{xy}$$

The new point all with same rotation are obtained with:

$$\text{dist} = \sqrt{x^2 + y^2}$$

$$x = dist * \cos rot$$

$$y = dist * \sin rot$$

The same calculations have been performed to obtain the new vertices after removing the rotation in the y-z plane. Thus, the new values for the y and z coordinates are:

$$y = dist * \cos rot$$

$$z = dist * \sin rot$$

where:

$$\text{dist} = \sqrt{y^2 + z^2}$$

and the value for rot is obtained evaluating the Four-Quadrant Inverse Tangent between the z and y coordinates and adding the offset term in the y-z plane.

The prostate volume corresponding to patient 18 is presented in figure 3.24 as an example of alignment, showing for each view the aligned volume on the left and the volume in the original space on the right.
The transformation object with the three offsets is saved for each prostate volume to obtain the mean transformation object by averaging over the entire training set.

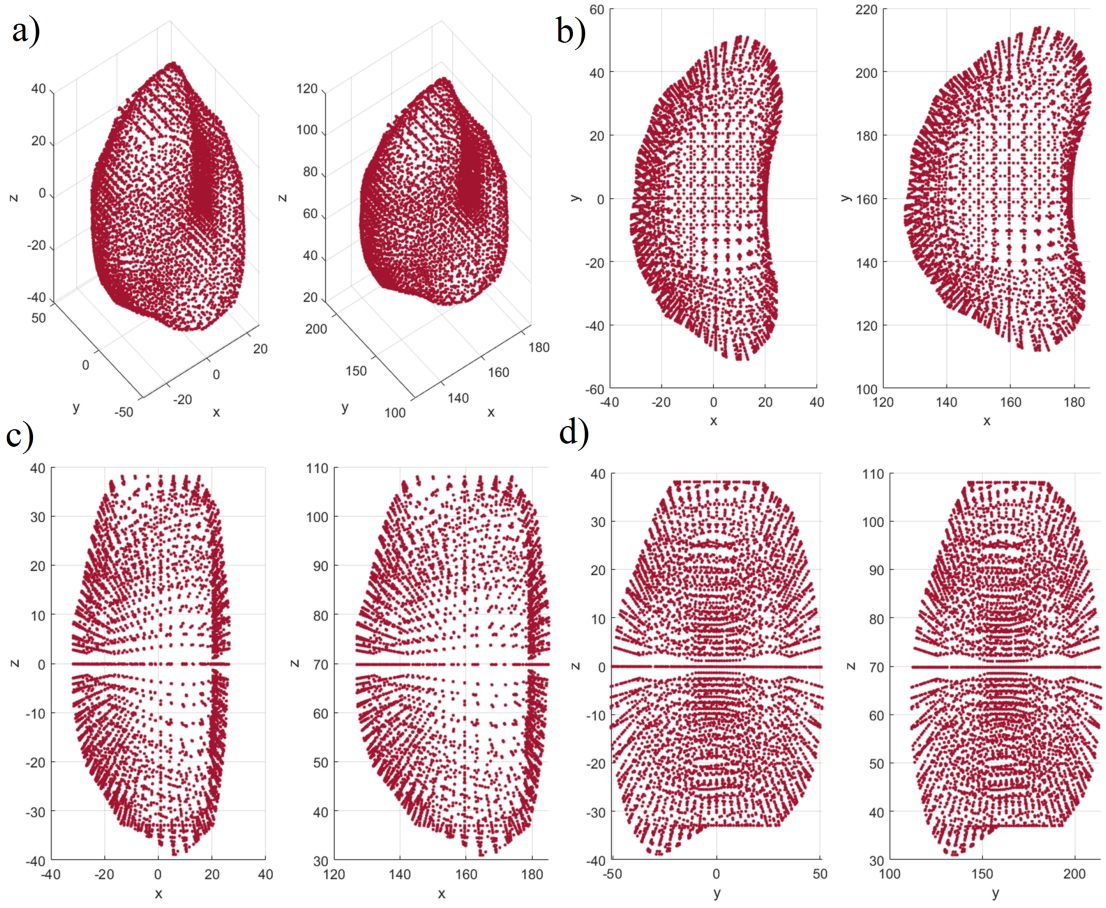


Figure 3.24: Example of alignment: Prostate shapes in xyz (a), x-y (b), x-z (c) and y-z (d) planes after (left) and before (right) alignment.

Subsequently a matrix is constructed containing all the contour point data of the training set and PCA is applied to this to obtain the eigen values, eigen vectors and the mean shape.
The contour noise is removed by keeping only the 98% of the Eigen vectors, i.e. the first for which the cumulative sum is grater than their sum multiplied by 0.98.

The mean shape is shown in figure 3.25.
It is possible to observe how the axes of the mean prostate shape are not perfectly aligned. This fact is due to the presence of incorrectly aligned prostates within the dataset (an example is presented in figure 3.26).

One of the possible reasons for this misalignment could be the fact that the points taken and a reference to identify the ray-triangle intersections change slightly from prostate to prostate, causing a variability which reflects on the difficulty of the algorithm in finding perfectly matching points.

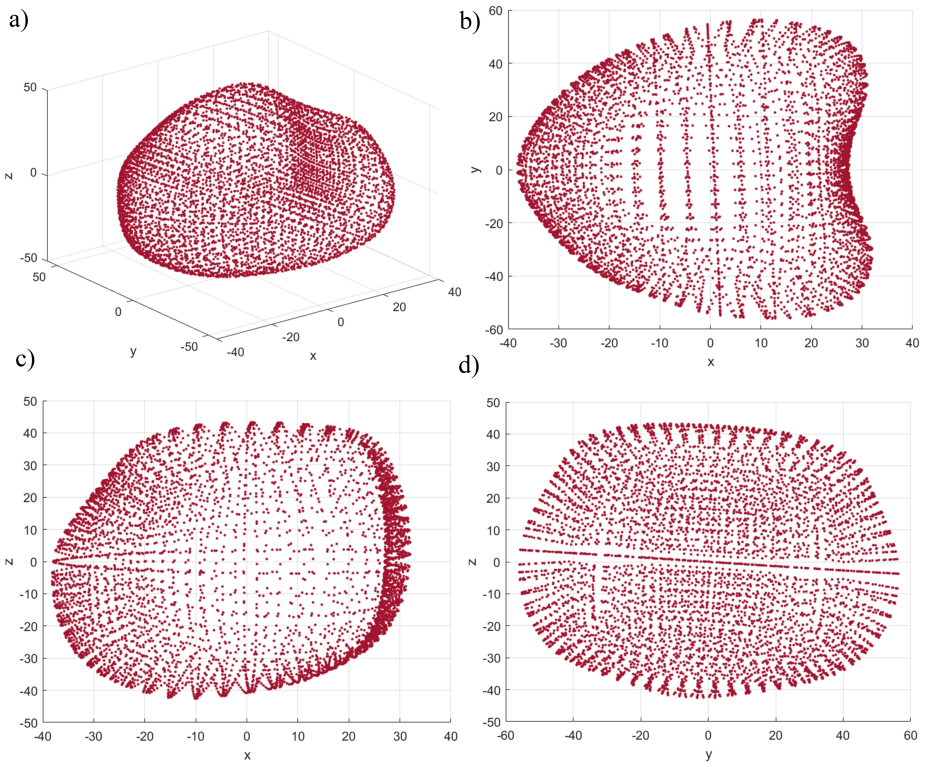


Figure 3.25: Mean Shape in xyz (a), x-y (b), x-z (c) and y-z (d) planes.

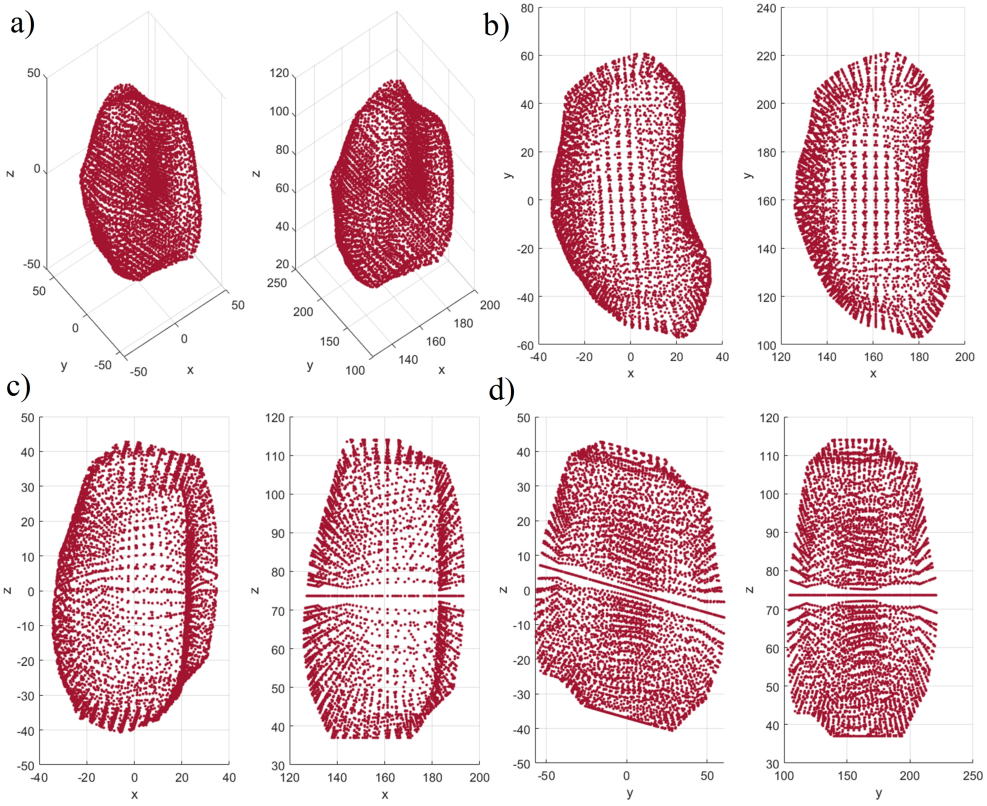


Figure 3.26: Example of non-alignment: Prostate shapes in xyz (a), x-y (b), x-z (c) and y-z (d) planes after (left) and before (right) alignment.

3.5.3 Appearance Data

The appearance of gray levels of the original images is used to optimize the search for the perfect outline.

Two types of models will be proposed, the first of which is based on the gray profiles computed along a normal to all points of the surface, while the second takes into account the derivative of these gray levels.

Both methods can make use of a multi-scale approach, which allows to obtain the parameters of interest using profiles calculated on different image scales.

Our approach employs three levels of resolution, from which it is possible to obtain the scale to which resize the image as:

$$\frac{1}{2^{res-1}}$$

In order to obtain the normals to a surface, the faces that constitute the different triangles of a triangulation are usually used.

The problem with this method is that the number of faces for each prostate volume changes depending on the triangulation algorithm.

To solve this issue the strategy we used was to create a point cloud starting from the vertices of the volume and from this construct the normals, considering for each point a neighborhood to approximate a surface to which the normal can refer.

As can be seen in figure 3.27, the normals have been oriented outwards with respect to the central point of the prostate volume.

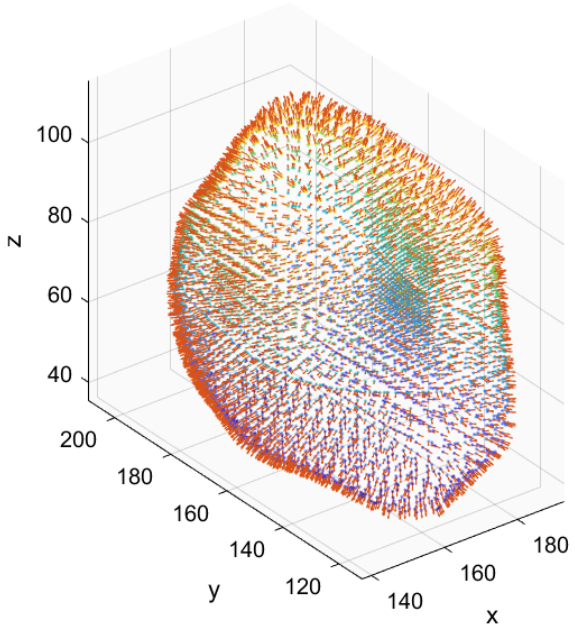


Figure 3.27: Point cloud of prostate vertices with normals oriented outwards with respect to the central point of the volume.

Gray Profiles

The algorithm for the computation of the gray profiles starts with the resolution corresponding to the original size of the volume (scale = 1) and then repeats the calculations for two other resolutions, respectively scale = 0.5 and scale = 0.25.
In the latter two cases, the volume is resized using linear interpolation.
Since the scale is smaller respect to the original one, an anti-aliasing filter is used during the resizing procedure to avoid image distorsions.

To calculate the profiles along the normals found in the previous point, a value k must be initially chosen, which represents the length of landmark intensity profile in one normal direction.

For each point belonging to the cloud of vertices, a profile is created whose length is $(k * 2) + 1$, centered at one point and with directions both entering and exiting the cloud (figure 3.28).
A cubic interpolation is then implemented to sample on the normal lines and to obtain the gray profiles corresponding to the points selected along the profile.

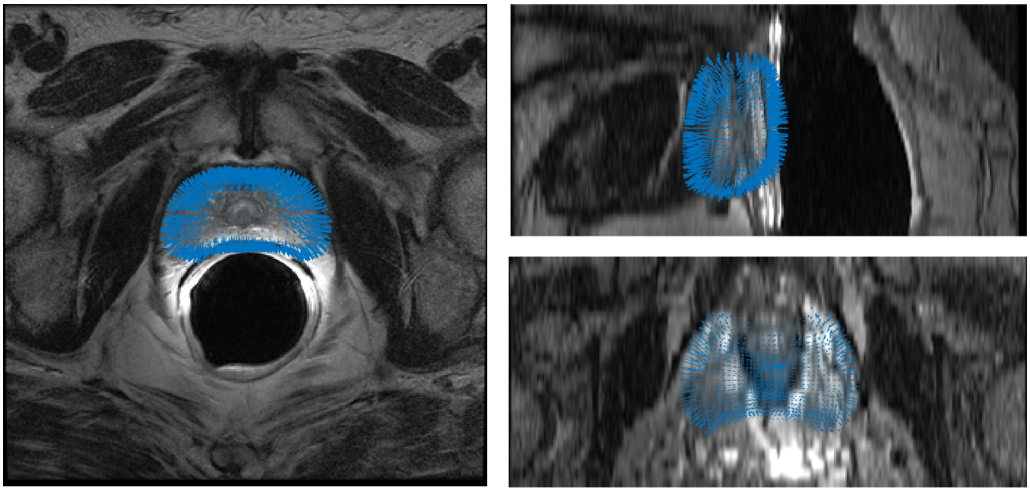


Figure 3.28: Normals (blue arrows) exiting the prostate surface in the axial, coronal and sagittal planes.

Since the prostate has a great variability along the contours, a clustering algorithm has been used to select only the most similar profiles for each point.
This algorithm is called Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and consists in finding clusters of arbitrary shape, taking into account the noise that may exist in the data distribution.
Before starting the search for the different clusters, it it necessary to declare two parameters:

- minPoints: a minimum number of points to be considered in order to define a set a cluster.
- ϵ: a parameter that will be used to determine the similarity between the different profiles.

As previously said, for each point of the vertices cloud, a normal profile is created. The first step of the algorithm consists in the random selection of a profile among all the profiles for that point within the training set and the subsequent determination of the neighborhood area of radius ϵ. Within this neighborhood similar profiles are chosen. If at least MinPoints are found, a cluster is created and enlarged according to the similarity of the neighborhood of all points belonging to the cluster, otherwise, if the value of MinPoints is not reached, the point is considered as an outlier, thus not reachable from any points belonging to a cluster (noise). Iteratively all the profiles and the space around them are inspected, trying to find regions with high density and to separate them from those with low density.

In our case the value 1 has been chosen as MinPoints, thus meaning that only one point is sufficient to define a cluster.
An optimization procedure has been carried out for the ϵ parameter in order to obtain a cluster with a number of profile between 5 and 20, which has been considered a good compromise to obtain sufficiently similar profiles. This procedure is repeated for each point belonging to the cloud of vertices. Principal component analysis is then applied to the profiles belonging to the cluster with more elements to find the eigen values, eigen vectors and the mean profile.
The comparison between the gray profiles obtained along the normal to a surface point for all 36 prostates of the training set and those obtained after applying clustering is shown in figure 3.29.

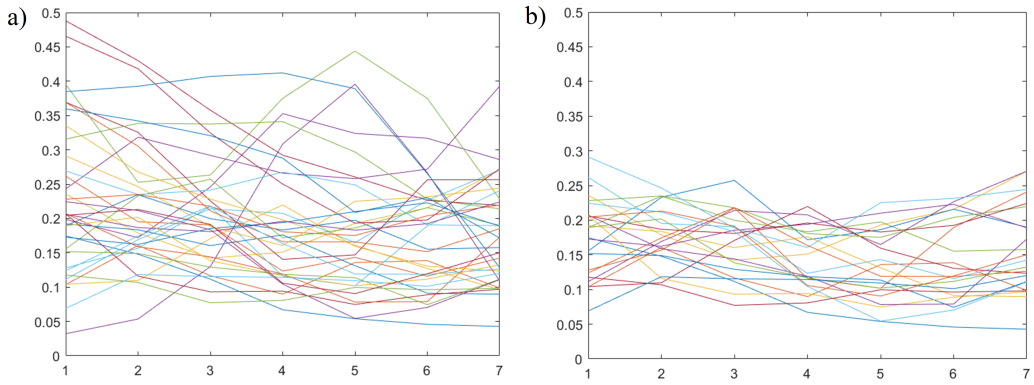


Figure 3.29: Gray-level profiles (a) for all 36 prostates belonging to the training set (36 curves). Selected profiles after DBSCAN (b).

Different lines represent different prostates into the dataset, on the x-axis we have the sample points on the normal profile while the y-axis represents the gray levels on the image.

Derivative of gray profiles

The second method involves the use of the derivative of the gray levels along the profiles normal to the surface. This latter is calculated starting from the grey-levels found in the previous case and subsequently normalized. The precedent clustering method is used to find the closest derivatives along normal profiles.
Also in this case an optimization for the ϵ parameter has been carried out to obtain a final cluster with a number of elements within the range $[5 - 20]$. The mean value of the derivatives inside the chosen cluster is computed and saved for each point of the prostate surface. The covariance matrix of the derivatives and the inverse of it are then computed and stored.
Figure 3.30a shows the derivative of the gray profiles for all 36 prostates, while in figure 3.30b we can observe the chosen profile after DBSCAN.

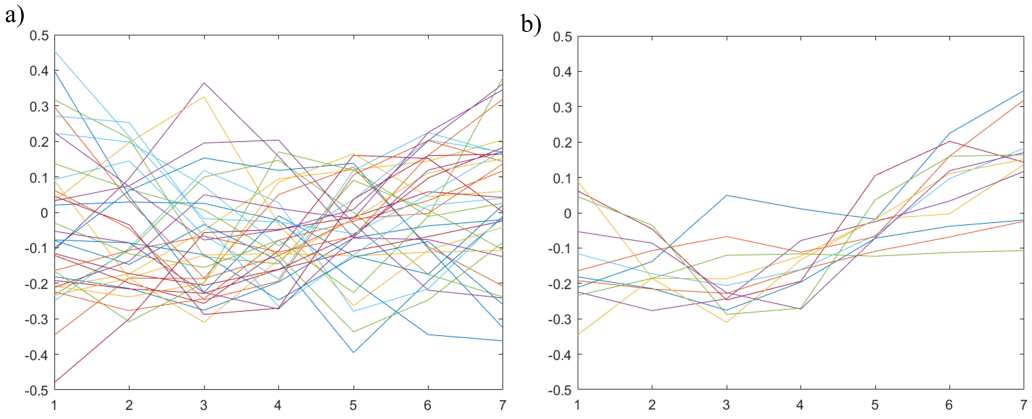


Figure 3.30: Derivatives of gray-level profiles (a) for all 36 prostates belonging to the training set (36 curves). Selected profiles after DBSCAN (b).

3.5.4 Apply Model

The model is applied on the vertices out from the 3D CNN after performing isotropic upsampling.
As previously mentioned, this method usually involves a multi-scale approach, thus with all the operations iteratively carried out on three levels.
With this method, the model is applied starting from a scale equal to 0.25 times the original size and optimized up to the original size.

The input of the algorithm is the matrix of vertices corresponding to a single volume.
Initially the point cloud of the volume points is computed in order to obtain the normals to the surface, then oriented towards the outside, as in the training phase.
Along each normal a profile is placed with length $n = k + n_s$, with k being the length of the profile defined in the training phase and n_s being a parameter corresponding to the search length in both directions (entering and exiting the cloud).

The gray profiles and the respective derivatives are computed for each landmark and an objective function is defined to calculate the energy that has to be minimized in order to move points in space to obtain the best fit.

The method dealing with the derivative of the gray levels requires the calculation of the Mahalanobis distance between the current values in the profile and values of the training data set profiles using the inverse correlation matrix.
For each point of the Vertices cloud the distance is evaluated as:

$$v = dg_{actual} - dg_{mean}$$

The objective function is then obtained as:

$$f = v' * S_{inv} * v$$

The second method involves calculating the PCA parameters and subsequently normalizing them with the variances.
The PCA parameters are obtained and normalized with the following formulas:

$$bc = \text{eigen vectors}' * (g - \bar{g})$$

$$bc = \frac{bc}{\sqrt{\text{eigen values}}}$$

The objective function is therefore:

$$f = \sum bc^2$$

The regularization error is calculated as the sum of the objective function values for all the points corresponding to the search length into the profile length.
The minimum value of f is selected at each iteration to calculate the movement of each point as:

$$movement = (ind - 1) - n_s$$

where

- n_s: length of search profile in both directions.
- *ind*: position in the profile at which the f reaches the minimum value.

Once the points have been moved to positions that are closer to the optimal ones, it is necessary to correct the set of points by removing possible outliers.
An outlier is defined as an anomalous value with respect to a given set of points, thus a data point that differs significantly from other observations.
In this specific case, a metric is used to define the degree of proximity for each point with respect to the ones surrounding it.
The correction of the outliers consists in the following steps:

- Alignment of the points according to the mean values of the Shape Model.
- Computing of the difference between the actual positions and mean positions obtained from the training set, scaled by the eigen vectors.

$$V = \text{eigen vectors} * (x_{search} - x_{mean})$$

- Outlier selection
  A point is defined an outlier if his value in V is $> 2.5\bar{V}$.
- Outlier removal
  For each outlier point define a k-Nearest-Neighbor space ($K = 50$ has been chosen for our specific case).
  The value on the contour corresponding to the outlier is then set to the mean value of his neighborhood defined by parameter k.

After having corrected the outliers, the model is realigned, removing translation and rotation, exactly as done in the training phase.
The b parameters are then obtained as:

$$b = \text{eigen vectors}' * (x_{search} - x_{mean})$$

and then limited to respect the constraints of the model using the eigen values of the PCA model obtained in the training phase.
Each point belonging to the final shape should lie within a bounding box whose limits are given by a parameter $m(= 3)$:

$$b_{max} = \pm m * \sqrt{\text{eigen values}}$$

The b parameters of the model are finally reset to the positions of the boundary in the original space with:

$$x_{search} = x_{mean} + \text{eigen vectors} * b$$

and the translation and rotation are restored with an inverse aligning procedure (that is the inverse of the procedure performed before to align points).

Several tests have been performed using both proposed methods.
A model has been chosen on the basis of a visual and subsequent numerical inspection and corresponds to the one that uses the derivatives of the gray levels to evaluate the Appearance Data along profiles normal to the surface. It has been applied to a single resolution (equal to 1) for 30 iterations.
The comparison between the various methods will be presented at the end, in the section relative to the intermediate results.
Figure 3.31 shows the ASM output superimposed on the 3D CNN output for a patient belonging to the training set.

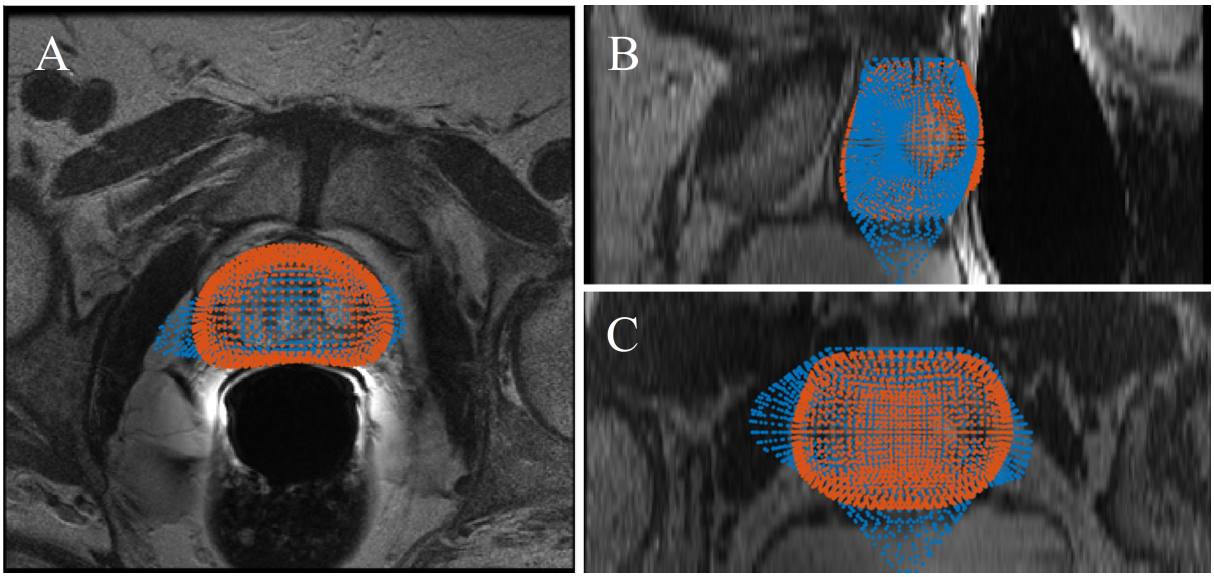


Figure 3.31: Comparison between ASM output (orange) and 3D CNN rough output (blue) in A) axial, B) sagittal, C) coronal planes. The model corrects the borders making them approach the real edges of the prostate by means of forces based on the grey-level derivatives.

3.5.5 Post-processing

The final vertices output from the Active Shape model must be post-processed in order to make a comparison with the ground truth volumes and extrapolate quantitative metrics to evaluate performance.

The clouds of vertices obtained with the model should in theory be very close to the real shape of the prostate, being limited by the bounding box defined by the model constraints.
A 3D reconstruction algorithm is applied to the vertices clouds, consisting of the following steps:

- Triangulation.
- Definition of the prostate range within the different volumes output from the model.
- Intersection between the volumes and perpendicular planes corresponding to the slices positioning in the original volumes.
- 3D reconstruction from 2D slices.
- Volume sub-sampling.
- Morphological operators.

Triangulation

The volume is reconstructed starting from the points of the surface using again the Alpha Shape method derived from the Delaunay triangulation. To obtain a closed surface without holes, the alpha radius has been set to 30.
After having obtained the Alpha shape, the vertices of the triangles that make up the triangulation are extrapolated in order to evaluate in the next point the intersections of these triangles with a set of rays defined in such a way to approximate the planes corresponding to the different slices.

As can be seen in figure 3.32 , the prostate surface after applying the model has a higher shape coherence respect to the one output from the 3D network as it is defined on the basis of a statistical model which takes into account the mean shape over the whole training set.

Slices interpolation and 3D reconstruction

The points corresponding to the contours of the prostate for the individual slices can be obtained by intersecting the volume obtained with the triangulation, with a series of planes whose z-coordinate corresponds to the number of slices in which the prostate appears.

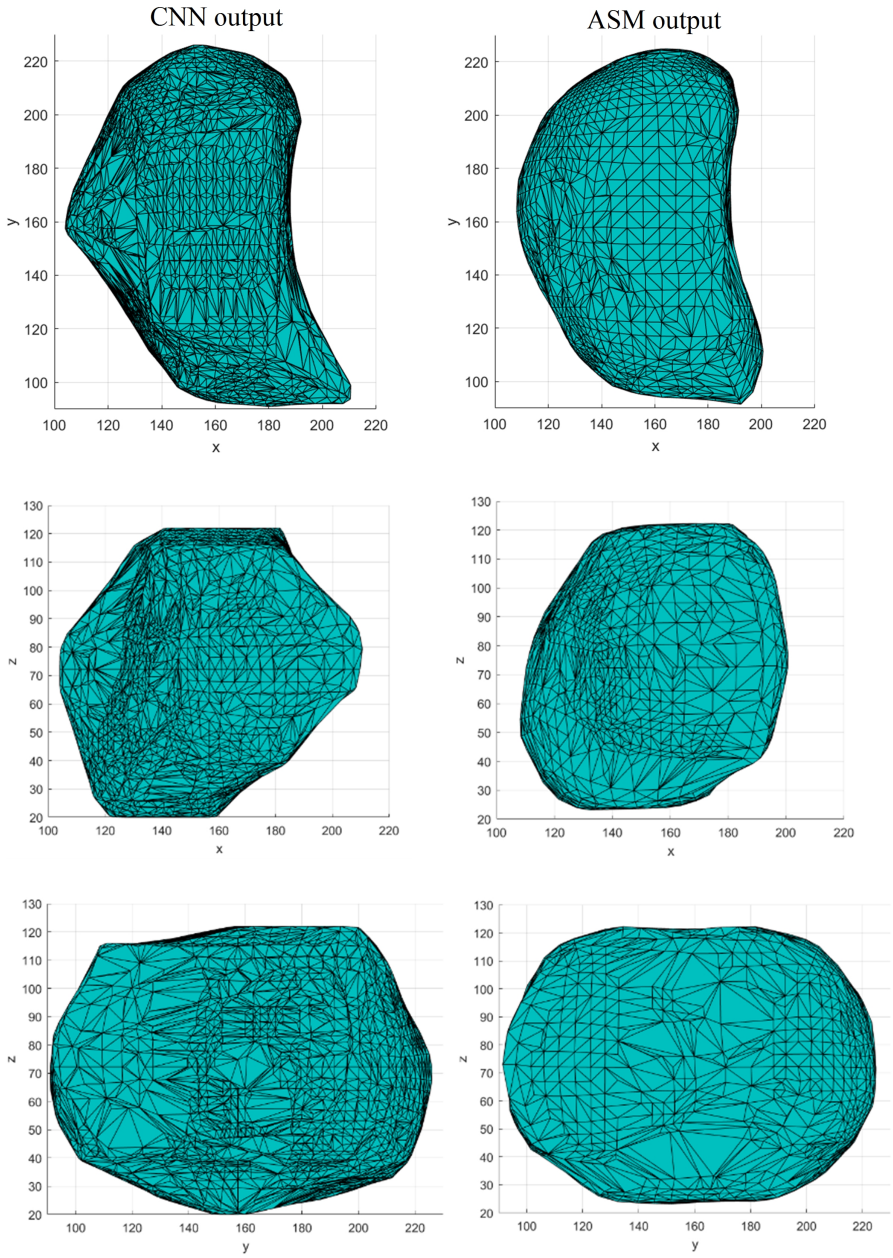


Figure 3.32: Comparison between the Alpha Shape obtained after 3D CNN alone and after applying the ASM for a prostate belonging to the test set.

The simplest way to obtain these intersections is to create a set of rays whose origin corresponds to the midpoint of the prostatic volume and which develop in the x-y plane. This set of rays is iteratively translated along the z coordinate to obtain the whole volume along the height dimension (figure 3.33).
To ensure an accurate approximation of each plane, 256 directions have been used for each slice, which correspond to 512 points found if the triangulation ensures a closed volume for all prostates in the set.

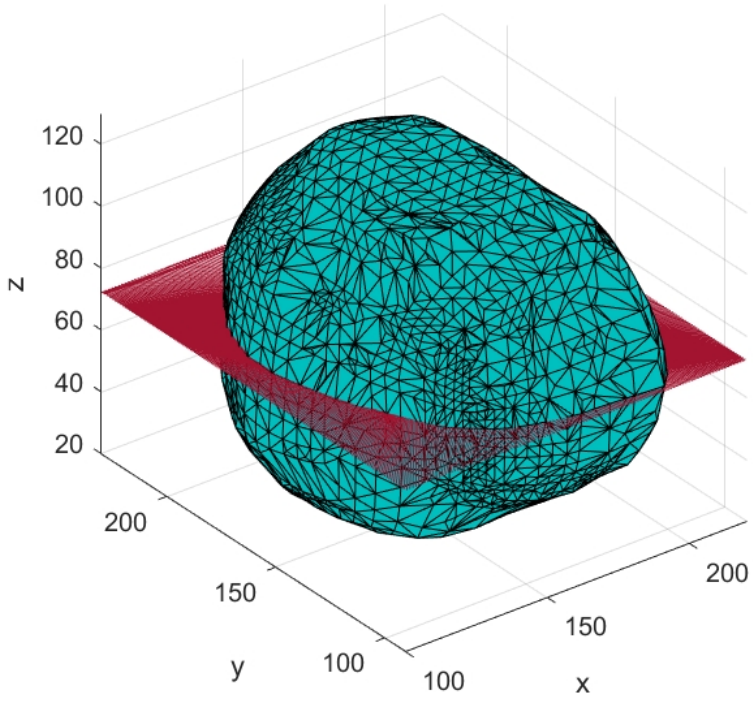


Figure 3.33: Prostate surface intersection with a plane obtained with rays that develop in the x-y plane.

The new vertices have been found using the same procedure described in section 3.5.1, thus computing the intersection between each ray and each triangle and taking the furthest points from the center of gravity, which correspond to a first approximation to the points on the outermost surface of the triangulation.

The 2D slices are obtained by creating a binary mask of the vertices corresponding to the edges of the prostate and adding empty slices where the prostate is not present.

Stacking together the 2D slices, the final 3D volume can be obtained, on which a little post-processing is subsequently carried out by filling holes and using a 3D morphological closing with a spherical structuring element of radius 4 to have smoother contours.
In order to make a quantitative comparison with the volumes corresponding to the ground truth, the volumes obtained are down-sampled to the original resolution.

3.6 2D CNN

The 2D Network has been proposed to make a comparison with the 3D Network.
Theoretically, the three-dimensional architecture should have grater spatial coherence, as the voxels are connected throughout the entire volume and processed during training using blocks that exploit the connection along the entire depth of the volume.

Little tuning has been performed for what concerns the parameters as the aim is a comparison with out proposed method and obtaining good results from the two-dimensional architecture was not out main scope.

3.6.1 Data preparation and Image Generator

Data to be passed as input to the network has been obtained by selecting a number of representative slices from the 3D volume as follows.
The idea is to ideally divide the total 24 slices into 3 slots, corresponding to the base of the prostate, the mid-gland and the apex.
The central part of the volume is the one that generally contains the gland with higher probability and it is therefore more defined, while in the other two slots, the slices containing the prostate are fewer and have less resolution.

A number of slices equal to 2 for the training set and 1 for the validation set have been randomly selected for each slot.
The same division as for the 3D network has been employed for the training-validation selection.
Two examples of slices used to train the network are presented in figures 3.34 (patient 3) and 3.35 (patient 20).

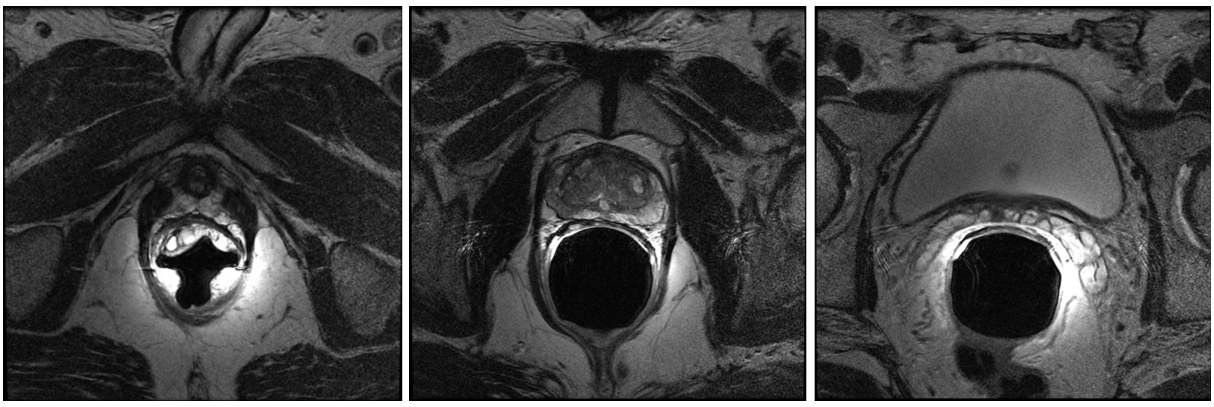


Figure 3.34: Prostate slices representing in order apex, mid gland and base for patient 3.

Figure 3.35: Prostate slices representing in order apex, mid gland and base for patient 20.

For the predict phase, in order to make a comparison between the performances of this network and those of the 3D network, two sets of data have been built without making any selection as regards the number of slices, testing the network on all the slices in the training set and subsequently on those of the test set.
The slices representing the images and the relative labels are saved in PNG format and subsequently loaded to create the network input, respectively in uint8 and float32 format.

An Image Data Generator is built to generate batches of tensor image data with real-time data augmentation.
The operations applied to the images are:

- Rotation: image randomly rotated within a degree range (20 degrees).
- Width Shift: shift along the width dimension in the range [- width shift range + width shift range]. Set to 0.2 (fraction of total width).
- Height Shift: shift along the height dimension in the range [- height shift range + height shift range]. Set to 0.2 (fraction of total height).
- Horizontal Flip: randomly flips input horizontally.
- Fill mode: fills the points outside the boundaries using the image values. Set to 'reflect' (image values are reflected along the width and height dimension).

The generator is created using the *'flow'* method, which takes data arrays and labels and generates batches of augmented data.
This method returns an Iterator yielding tuples of (x, y), where x and y are lists of Numpy arrays representing the images and the corresponding labels.

3.6.2 Network Architecture and parameters optimization

The 2D network is built using a basic UNet model present in the *'segmentation models'* package, shown in figure 3.36.
The chosen backbone, the part of the network that deals with feature extraction, is the *resnet34*.

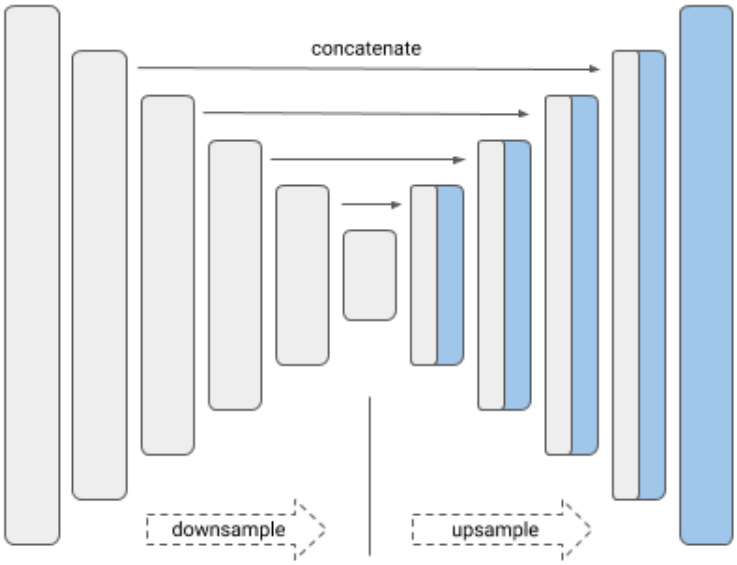


Figure 3.36: Basic UNet architecture: first downsampling path followed by an upsampling path which restores the initial dimensions.

The network is not trained from scratch since using the *resnet34* network as the backbone structure allows to use the weights trained on *2012 ILSVRC ImageNet dataset*.
In order to use these weights, it is necessary to have consistency between the size of the input data and the size of the input on which the network has been trained.
For this reason, a 2D convolution block has been added to map the current number of channels, in our case equal to 1 since gray-level images are used, to 3 channels.

The model is compiled with the following settings:

- Optimizer: Adam.
- Loss function: Binary cross-entropy.
- Metric: Binary accuracy.

For what concerns the callbacks, the same functions have been used as in the 3D network, i.e. Model Checkpoint and Early Stopping with the same parameters.

Network training is performed using the *'fit generator'* method.
Unlike before, within this function data augmentation is carried out in parallel by calling the generator function which is passed to the *'flow'* method.
The generator function yields a batch, of size set to 4 in this case, to the *'fit generator'* function. This latter accepts the batch of data, performs backpropagation and updates the weights of the model.
The process is repeated until the preset number of epochs is reached or ends earlier if the requirements for the Early Stopping criterion are reached.
The values for the training and validation steps are set to:

$$\text{training steps} = \frac{\text{samples in the training set}}{\text{training batch size}}$$

$$\text{validation steps} = \frac{\text{samples in the validation set}}{\text{validation batch size}}$$

The learning curve is shown in figure 3.37. The accuracy value reached is very high, as the loss value is very close to zero.

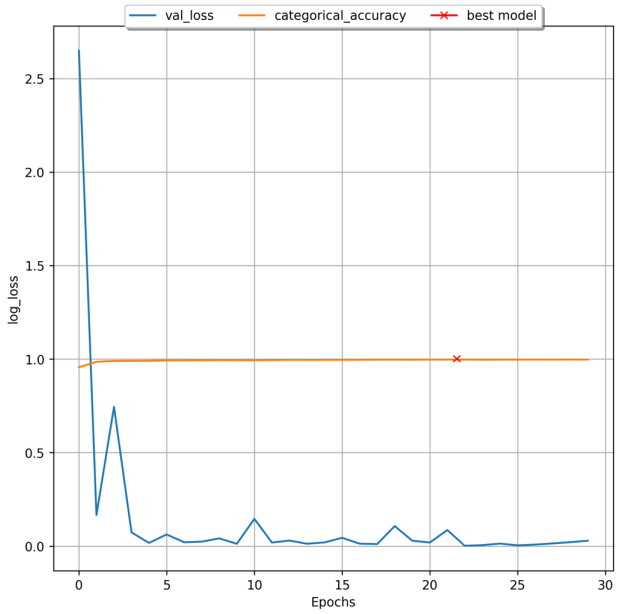


Figure 3.37: Learning curve for 2D CNN trained using binary crossentropy as loss and binary accuracy as metric.

The segmentations obtained from the 2D network are stacked together to obtain the volumetric label, which is subsequently binarized.

3.7 Performance metrics

Two metrics have been used to evaluate the final performance of the segmentation using the three models:

- Dice Similarity Coefficient

  It determines the similarity between two volumetric labels in terms of intersection over union.
  Given two labels, X and Y, defined as sets of voxels, the DSC is given by:

  $$\text{DSC} = \frac{2|X \cap Y|}{|X| + |Y|}$$

  where $|X|$ and $|Y|$ represent the two cardinalities of the two volumetric labels, thus the number of elements in each set.
  Mathematically, this index counts twice the number of voxels common to the two labels and divides it by the sum of the number of elements of each label. It ranges from 0 (no overlap) to 1 (total overlap).

- 95% Hausdorff Distance

  The maximum Hausdorff distance determines the resemblance between two labels in space as the greatest of all the distances from a point belonging to a label to the closest point belonging to the other label.
  Given two sets of points X and Y, the maximum HD is given by:

  $$d_H(X,Y) = max(d_{XY}, d_{YX}) = max(\max_{x \in X} \min_{y \in Y} d(x,y), \max_{y \in Y} \min_{x \in X} d(x,y))$$

  In our specific application the HD between the two volumetric labels has been calculated evaluating the maximum value of the Euclidean distance transform of the surface points of a binary label in correspondence with the surface points of the other label.
  The HD between two volumetric labels, if both of them are bounded, is a finite number and it is lower if the points of one set are very close to the points of the other set.
  The 95% HD calculates the 95^{th} percentile of the distances between boundary points in X and Y. It is used as it eliminates the impact of possible outliers.

For what concerns the intermediate results, the considerations that led to the choice of a set of parameters or a model over another have been based simply on the Dice Similarity coefficient (mean value and standard deviation).

Chapter 4

Results and discussion

4.1 Final results

The numeric values for the two performance metrics described in section 3.7 obtained for the train and test sets using the three different models are shown in tables 4.1 and 4.2, while figures 4.1 and 4.2 show two examples of final segmentation for training and test set respectively.

Training set				
Model	**Mean DSC**	**Devst**	**Mean HD**	**Devst**
3D CNN	**0.893**	**0.0205**	7.941	3.2115
3D CNN + ASM	0.881	0.0399	8.618	2.1658
2D CNN	0.886	0.0308	6.287	2.7191

Table 4.1: DSC and HD values with standard deviations obtained with the three models tested on the training set.

Test set				
Model	**Mean DSC**	**Devst**	**Mean HD**	**Devst**
3D CNN	0.840	0.0387	10.747	5.3995
3D CNN + ASM	0.845	0.0479	9.545	3.0378
2D CNN	**0.878**	**0.0275**	6.403	2.1045

Table 4.2: DSC and HD values with standard deviations obtained with the three models tested on the test set.

Concerning the results obtained on the training set, we can observe that the 3D network demonstrates better performances in terms of DSC with relative standard deviation compared to the 2D network, but a little worse if we observe the HD value.
Applying the Active Shape Model worsens the performances of the 3D CNN alone a little in terms of DSC and HD, but it guarantees the lowest mean standard deviation for what

concerns the HD value, as it deforms all the shapes to try to approach the real shape, based on the image characteristics.

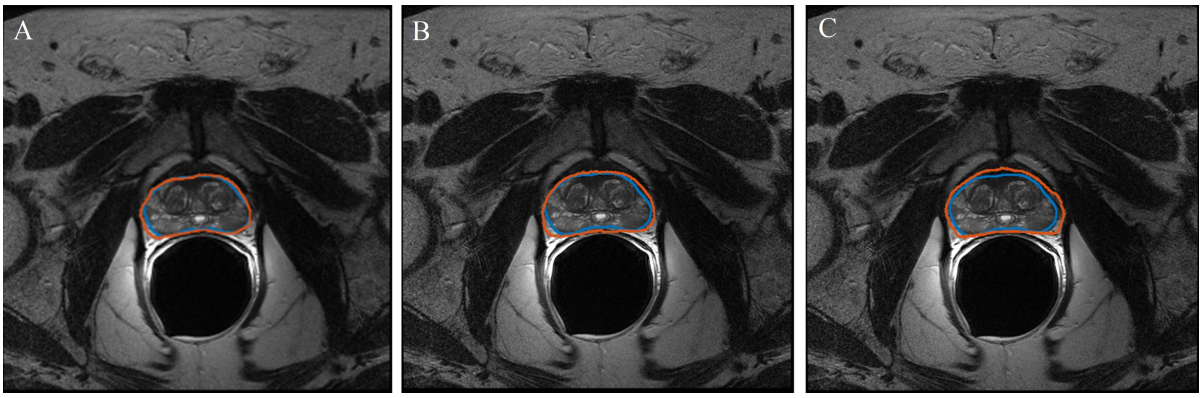


Figure 4.1: Comparison between manual and final segmentation on an image belonging to the training set obtained for A) 3D CNN, B) 3D CNN + ASM, C) 2D CNN. The blue curve represents the manual segmentation, while the orange one is the segmentation obtained with our algorithms.

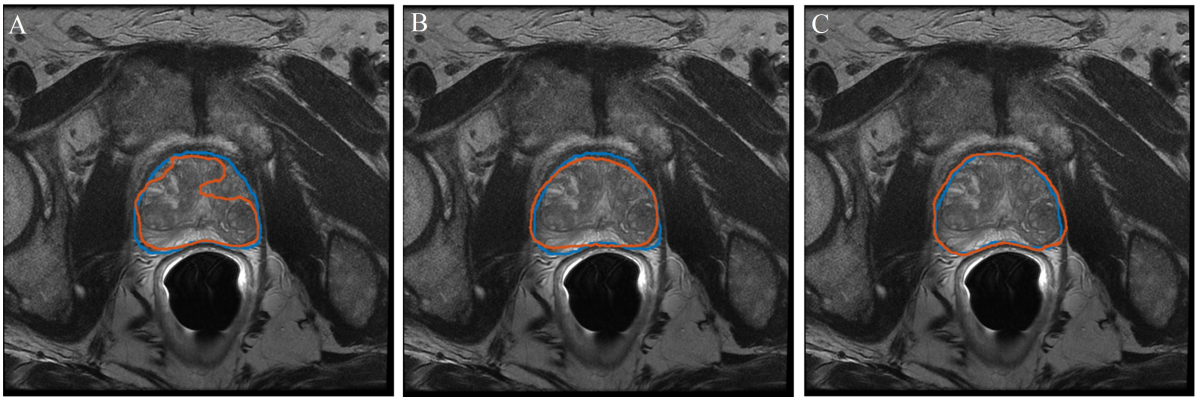


Figure 4.2: Comparison between manual and final segmentation on an image belonging to the test set obtained for A) 3D CNN, B) 3D CNN + ASM, C) 2D CNN. The blue curve represents the manual segmentation, while the orange one is the segmentation obtained with our algorithms.

The results obtained for the test set, on the other hand, show another trend. The best performance has been obtained using the 2D network both in terms of DSC and mean HD.

The Active Shape Model improves the performance of the 3D CNN alone both in terms of DSC and HD, probably because the contours coming out of the network on the test set differed much from the real ones and the model, being trained on manual labels and on derivatives of the gray levels of the images, allows the edges of the segmentation to approach those of the ground truth with higher precision.

For the test case, the mean HDs and their relative deviations are higher respect to the ones obtained for the training set as within the test set there are few single elements with a much higher HD than the mean one.

Performance explanation

Possible explanations of the difference in performance between the 3D implementations compared to the 2D model can be:

- The 2D CNN model used for the comparison is deeper and pre-trained on the ImageNet dataset, which guarantees better results because more parameters are used for training.
- Approximation errors when computing the coherent vertices for the ASM

  The triangulation algorithm used to determine the prostate surface from the points of the labels output from the CNN, as discussed in section 3.5.1, needed a parameter alpha to compute the triangle to approximate the surface at a specific resolution.
  The algorithm proposed for the determination of coherent vertices requires closed surfaces in order to accurately approximate the shape of the prostate.
  The manual labels are open surfaces because they have been obtained by focusing mainly on the central part of the prostate, avoiding segmenting extreme slices as the base and the apex part in some cases were not well defined.
  To close these surfaces an α parameter equal to 50 has been used, which worsened the refinement as can be seen in figure 4.3.
  In order to obtain better results for the final segmentation using this triangulation algorithm two possible solutions can be applied to solve the problem of bad refinement:

  1. Close the surface of the manual labels by adding points to interpolate the upper and lower slices after computing the outer vertices (explained in section 3.5.1). In this way the following interpolation creates triangles also on the upper and lower parts of the surface.
  2. Obtain a smoother surface to approximate the data
     Keep a high α parameter to close the volume and relax the Alpha Shape to fit the surface using a penalty when the triangles obtained with the triangulation are far from the original outer points of the label and another parameter to control the curvature between adjacent triangles from being too high.

The numerical results obtained are in line with those of other approaches cited in the literature.

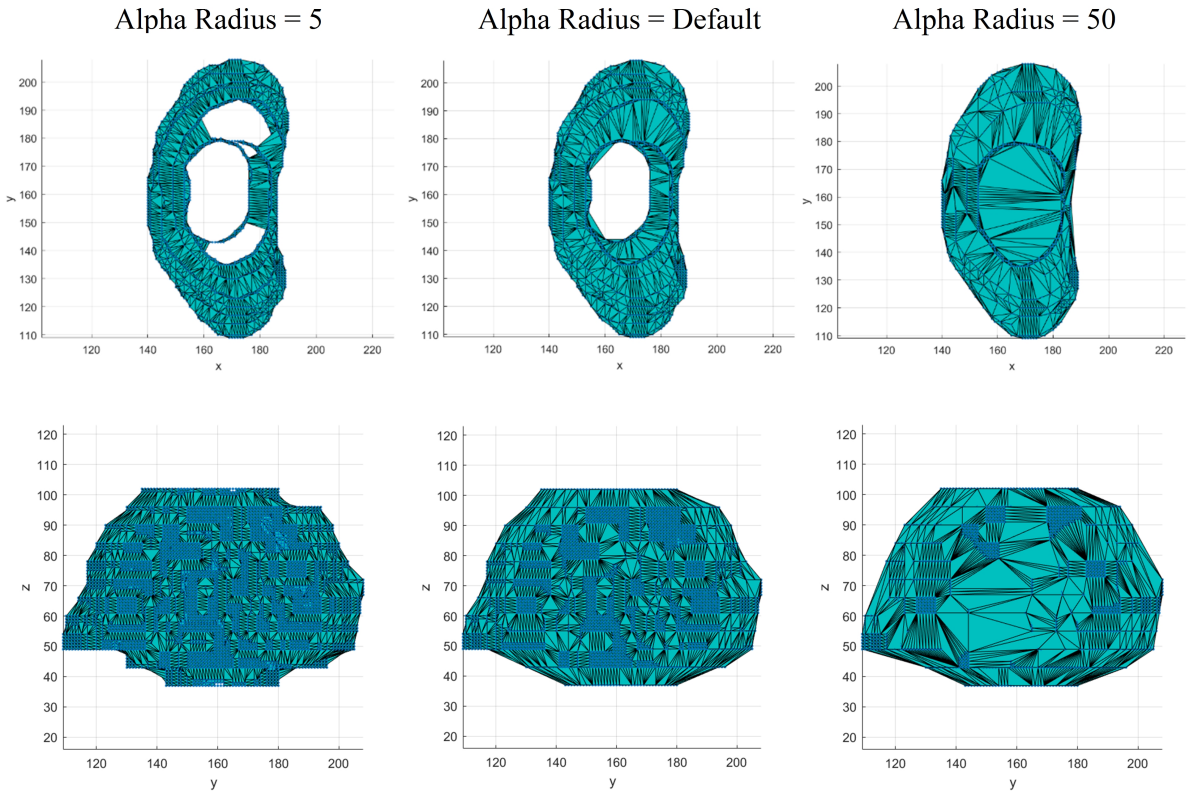


Figure 4.3: Different surface approximations obtained by changing the α parameter of the Alpha Shape triangulation. The ideal case would be setting no parameter, so that an optimal *alpha* is automatically set to obtain the best approximation for the blue points, representing the real contour of the label.

The performances are a bit lower due to the above-mentioned problems, specifically a simple 3D network model has been used because of memory/computational power issues, which does not allow to reach sufficiently high depth and number of parameters to achieve better results.

The hybrid approach we have used proved useful as is allows to improve the DSC and HD values of the 3D network alone in the test phase, making unplausible shapes closer to the real ones and bringing the edges closer to the desired ones.

The limitations concerning the active shape model have been discussed in terms of approximations introduced by the triangulation used to find the points that determine the Point Distribution Model.

This statistical model is sensitive to the initialization of the points, so if the contours obtained with the CNN are located in regions of the image where it is difficult to distinguish discriminating features, the model may struggle to move towards the real contours.

These limitations can be overcome by firstly optimizing the correspondence between the points found on the label output from the CNN and the points obtained with the triangulation, and secondly by optimizing the terms that govern the movement of the points when the model is applied (function to be minimized).

4.2 Intermediate results

Output 3D CNN

In the part of the method relating to the 3D network, two test have been carried out using two metrics and loss functions and setting different initial learning rates passed as a parameter to the optimizer when compiling the model before starting the training.
Table 4.3 presents the four cases dealt with, while in table 4.4 we can observe the results of the DSC on the test with relative mean and standard deviation.
The network trained with DSC as metric and DSC loss as loss visually presents the best performance among the four (figure 4.4), confirmed by the fact that is has the highest mean DSC (table 4.4).

Case	Loss	Metric	Learning rate (Adam)
1^{st}	Binary crossentropy	Binary accuracy	None (Default 0.01)
2^{nd}	Binary crossentropy	Binary accuracy	0.0001
3^{rd}	Dice Similarity Coefficient loss	Dice Similarity Coefficient	0.00001
4^{th}	Dice Similarity Coefficient loss	Dice Similarity Coefficient	0.0001

Table 4.3: DSC and HD values with standard deviations obtained with the three models tested on the test set.

PatID	1^{st}	2^{nd}	3^{rd}	4^{th}
110	0.783	0.811	0.79	**0.827**
13	0.867	0.872	0.833	**0.873**
19	0.738	**0.801**	0.759	0.796
2	0.786	0.767	**0.809**	0.787
2004	0.891	0.87	0.839	**0.909**
28	0.818	0.823	0.722	**0.844**
3013	0.871	0.876	0.803	**0.877**
3033	0.84	**0.86**	0.72	0.802
3059	0.773	0.767	0.754	**0.81**
31	0.824	0.834	0.855	**0.857**
3120	0.671	0.85	0.685	**0.87**
3211	0.727	0.778	0.722	**0.816**
41	0.816	0.834	0.832	**0.887**
53	0.775	0.783	**0.785**	0.783
58	0.869	0.844	0.816	**0.872**
Mean DSC	0.803	0.825	0.782	**0.84**
DevSt	0.0527	0.0209	0.0379	0.0224

Table 4.4: DSC and relative standard deviations obtained for the four cases in table 4.3

As can be observed, most of the highest values have been obtained using the 4^{th} method, namely DSC as metric and DSC loss as the loss function.

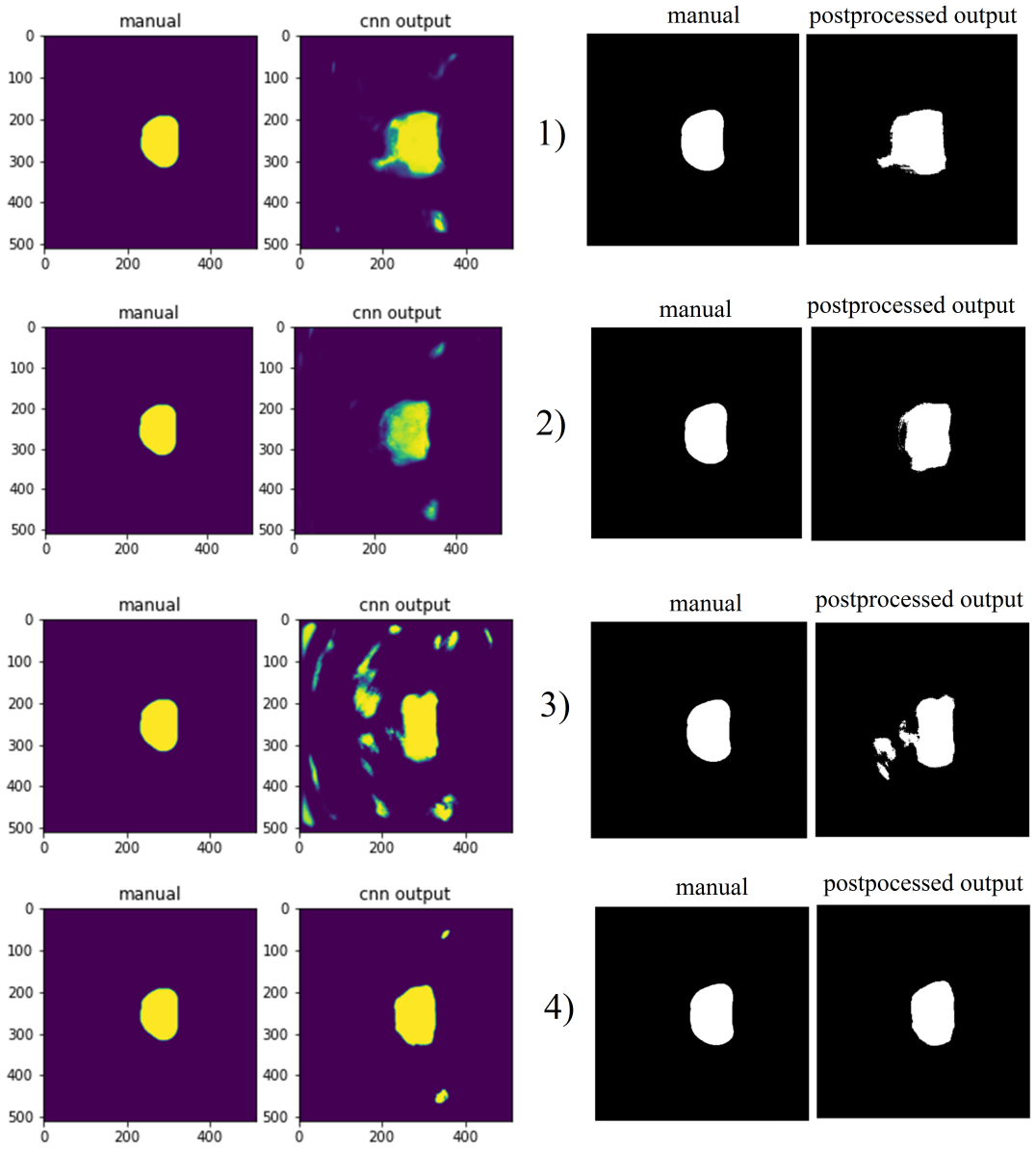


Figure 4.4: Comparison between manual annotation and 3D CNN outputs for the four cases: 1) Binary crossentropy - Binary accuracy lr=0.01, 2) Binary crossentropy - Binary accuracy lr=0.0001, 3) DSC - DSC loss lr=0.00001, 4) DSC - DSC loss lr=0.0001. On the left rough CNN output, on the right after post-processing.

Furthermore, for our specific problem, it can be observed that even in the 1^{st} case (binary crossentropy and binary accuracy with learning rate equal to 0.0001) the results obtained are higher than the other two since the learning rate seems to be the most suitable.

As regards the optimization of the parameters, a test has been carried out keeping the depth of the network fixed and increasing the number of convolution filters to 12.
The performance with this model has been evaluated calculating the DSC with relative standard deviation, obtaining an average DSC value similar to that obtained with 8 filters as regards the evaluation on the training set, but a lower value on the test set.

The network in this case undergoes overfitting and for this reason it has been decided to carry out a batch normalization operation that can allow to mitigate this effect by reducing the internal covariate shift.

Training set		
Number of conv filters	**Mean DSC**	**Devst**
8	0.893	0.0205
12	0.896	0.0285
12 with batch norm	0.939	0.0288

Table 4.5: DSC values with standard deviations obtained for the three cases tested on the training set.

Test set		
Number of conv filters	**Mean DSC**	**Devst**
8	0.840	0.0387
12	0.831	0.047
12 with batch norm	0.848	0.0424

Table 4.6: DSC values with standard deviations obtained for the three cases tested on the training set.

The numerical results for training and test sets are shown in tables 4.5 and 4.6 respectively, together with those obtained for the case of 8 filters, while in figure 4.5 it is possible to observe the corresponding learning curves setting the max number of epochs to 150.

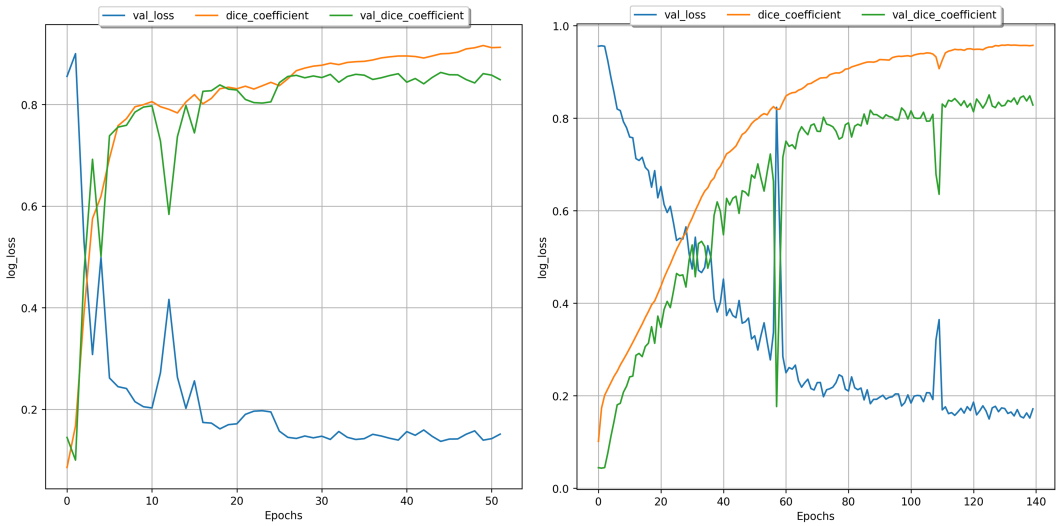


Figure 4.5: Learning curves obtained by training the network with 12 initial conv filters without (left) and with (right) batch normalization. Both training processes stop before reaching the maximum number of epochs because the early stopping criterion is satisfied.

The problem of overfitting continues to arise even after performing batch normalization, reaching higher values for both training and test set, but with a greater gap. For this reason, this network configuration has not been chosen as the best one for our study.

3D-2D Model comparison

The 2D network we have used to make a comparison with our proposed model, for the above-mentioned reasons, is advantageous as it allows to exploit the weights pre-obtained on a different dataset and for this reason, the low level features must not be recalculated once the training has started.
An advantage that the 3D network presents compared to the 2D model, which led us to propose the three-dimensional approach, is the spatial coherence that creates between successive slices.

Figure 4.6 shows an example of four slices extracted from the rough output of both the 3D and 2D models (patient 3042).

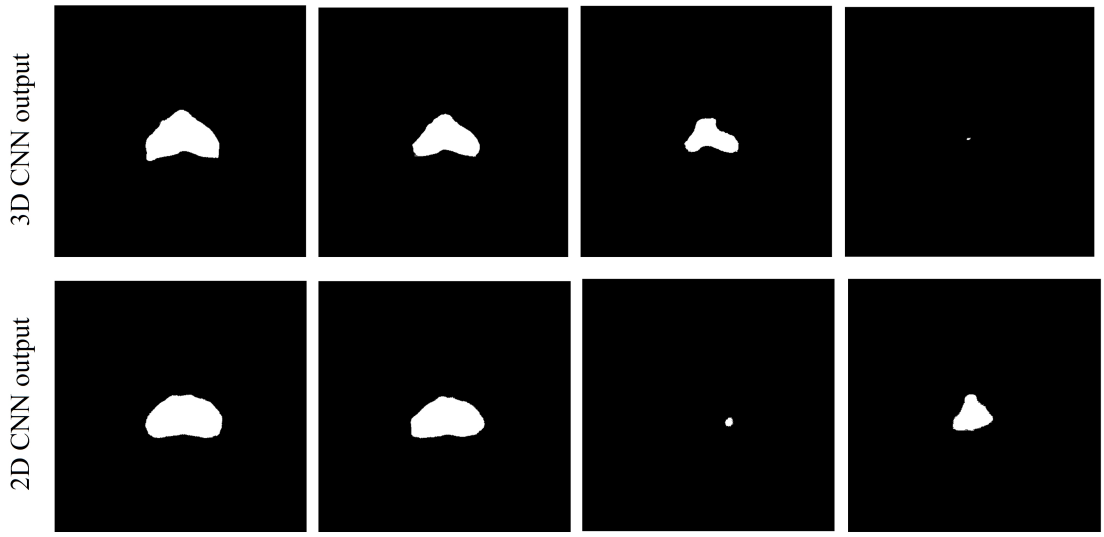


Figure 4.6: Comparison between rough 3D and 2D CNN outputs (slices 16-19, patient 3042). We can observe that the 3D CNN classifies voxels taking into account their relative position in space, while for the 2D CNN the classification is performed on the single slice, thus results can be erroneous when observed in 3D space.

As we can observe, since the slices are classified individually by the 2D network, the values obtained do not take into account the position of the slice within the volume and therefore, the connection that is established between successive voxels.

Active Shape Model

As described in section 3.5.4, after carrying out different tests using both approaches

(gray profiles or their derivatives to calculate the statistics along the profiles normal to the prostate surface), the one using the derivatives has been selected as the best model. The chosen method foresees an approach that uses only one scale (res = 1) instead of the multiresolution approach and the calculation of the best profile is performed over 30 iterations.

Tables 4.7 and 4.8 show the numerical results in terms of DSC with relative standard deviation obtained for the test and the training set respectively in the following three cases:

- DGL-1: Derivatives of gray levels with multiscale approach, 1 iteration per scale.
- DGL-30: Derivatives of gray levels with single scale approach, 30 iterations.
- GL-3: Gray profiles with single scale approach, 3 iterations.

We can see that, apart from the method that has been selected, the second best result both for the tests on the training set and those on the test set, has been achieved using a single scale approach.

Test set

PatID	**DGL-1**	**DGL-30**	**GL-3**
110	0.815	0.813	0.838
13	0.853	0.873	0.859
19	0.824	0.815	0.8252
2	0.759	0.811	0.76
2004	0.903	0.9	0.921
28	0.878	0.874	0.877
3013	0.867	0.893	0.888
3033	0.86	0.886	0.864
3059	0.737	0.742	0.728
31	0.87	0.861	0.871
3120	0.89	0.89	0.892
3211	0.815	0.803	0.806
41	0.871	0.887	0.876
53	0.769	0.766	0.777
58	0.858	0.86	0.882
Mean DSC	0.838	**0.845**	0.844
DevSt	0.0527	**0.0209**	0.053

Table 4.7: DSC and relative standard deviations for the three methods on test set: DGL-1 (Derivatives of gray levels, 1 iteration - multiscale approach), DGL-30 (Derivatives of gray levels, 30 iterations - single scale approach), GL-3 (Gray levels, 3 iterations, single scale approach).

Training set			
PatID	**DGL-1**	**DGL-30**	**GL-3**
101	0.841	0.848	0.861
102	0.890	0.897	0.904
16	0.849	0.857	0.852
18	0.844	0.841	0.85
20	0.883	0.898	0.887
2001	0.822	0.825	0.841
2002	0.797	0.826	0.787
2013	0.863	0.869	0.848
2022	0.893	0.895	0.893
22	0.889	0.899	0.889
23	0.886	0.88	0.875
24	0.862	0.872	0.853
26	0.878	0.9	0.875
3	0.907	0.917	0.899
3005	0.884	0.916	0.882
3007	0.876	0.875	0.864
3009	0.905	0.906	0.888
3030	0.882	0.87	0.889
3032	0.913	0.94	0.901
3036	0.861	0.869	0.878
3042	0.854	0.847	0.866
3154	0.935	0.934	0.933
3160	0.921	0.917	0.921
32	0.857	0.854	0.846
33	0.829	0.813	0.837
37	0.889	0.9	0.889
40	0.816	0.813	0.839
44	0.902	0.893	0.905
47	0.904	0.871	0.904
49	0.87	0.884	0.893
50	0.868	0.879	0.883
54	0.908	0.924	0.902
55	0.896	0.903	0.897
59	0.871	0.908	0.872
68	0.914	0.927	0.923
8	0.841	0.849	0.85
Mean DSC	0.875	**0.881**	0.877
DevSt	0.0315	**0.0209**	0.0295

Table 4.8: DSC and relative standard deviations for the three methods on training set: DGL-1 (Derivatives of gray levels, 1 iteration - multiscale approach), DGL-30 (Derivatives of gray levels, 30 iterations - single scale approach), GL-3 (Gray levels, 3 iterations, single scale approach).

Chapter 5

Conclusion and future remarks

In this study we propose a fully automatic hybrid approach for the 3D MRI prostate segmentation using CNN and ASM models.
The 3D CNN is employed to obtain a first segmentation to be used as initialization for the statistical model, avoiding the computation of manual features for the positioning of the model in the initial phase. The ASM is subsequently used to obtain a refining in correspondence with the contours of the prostate as this statistical model is trained on the labels manually obtained to have a statistical estimate of the image characteristics near the contours of the organ.
Another reason that led us to propose this model in addition to the segmentation with the deep neural network is the fact that the 3D UNet model does not guarantee that the predicted point cloud is noise free, which leads some boundaries to move away from the actual ones.
During the application of the ASM, the shape that the segmentation can have is limited on the basis of the mean one within the dataset, which guarantees the plausibility of the shapes obtained as final output, mitigating the problem of noise.

The results obtained demonstrate how the application of the statistical model in cascade to the CNN improves the performance of the segmentation, increasing the values of DSC and HD and decreasing the relative standard deviations, indicating that examples that after the segmentation with the CNN demonstrate a value that deviates from the correct one are adjusted, bringing the segmentation closer to the real one especially in correspondence with the contours.

However, our method has some limitations. The CNN model used presents a simple architecture and to obtain better results it is necessary to increase the depth of the network and the number of filters used for the convolution operations, in order to have more parameters available for the learning process.
The neural network model can be improved by optimizing the architecture and the training

parameters in order to create a more complex network without running into the problem of overfitting.
As far the statistical model is concerned, the limitations regard the approximation of the shape obtained as output from the CNN.
The triangulation algorithm that allows to obtain the corresponding points in each prostate approximates the shape of the segmentation in a non-optimal way. Since this algorithm is applied on the volumetric labels following an isotropic upsampling, the interpolated slices are approximated obtaining an overestimated shape, not perfectly corresponding to the real one of the CNN output.
For this reason, the values of the performance metrics only slightly improve the performance of the CNN alone.
As explained in section 4.1, the starting point for the model can be improved by adding terms that better approximate the shape of the CNN output by penalizing the curvature within the triangulation algorithm or by using different triangulation algorithms for this purpose, observing how these affect the final performance.
Another limitation of our work is the rather limited number of volumes used for training. A wider dataset could allow the network to learn more cases, limiting possible overfitting problems once the depth of the network and the number of trainable parameters are increased.
A possible future work can be applying this method to other acquisition modalities, for example DWI volumes, using a higher number of subjects if possible to observe the behaviour of the network in a first moment and subsequently of the network with the statistical model applied, facing a wider dataset and different image characteristics due to different acquisition modalities.

Bibliography

[1] Ruida Cheng et al. "Active appearance model and deep learning for more accurate prostate segmentation on MRI". In: Mar. 2016, p. 97842I. DOI: 10.1117/12.2216286.

[2] TF Cootes et al. "Use of active shape models for locating structures in medical images". In: *Image and Vision Computing* 12.6 (1994). Information processing in medical imaging, pp. 355–365. ISSN: 0262-8856. DOI: https://doi.org/10.1016/0262-8856(94)90060-4. URL: https://www.sciencedirect.com/science/article/pii/0262885694900604.

[3] BJ Erickson et al. "Machine Learning for Medical Imaging". In: *Radiographics* 37.2 (2017), pp. 505–515. DOI: 10.1148/rg.2017160130.

[4] Lei Geng et al. "Encoder-decoder with dense dilated spatial pyramid pooling for prostate MR images segmentation". In: *Computer Assisted Surgery* 24.sup2 (2019), pp. 13–19. DOI: 10.1080/24699322.2019.1649069.

[5] B van Ginneken et al. "Active shape model segmentation with optimal features". In: *IEEE Trans Med Imaging* 21.8 (2002), pp. 924–33. DOI: 10.1109/TMI.2002.803121.

[6] Baochun He et al. "Automatic Magnetic Resonance Image Prostate Segmentation Based on Adaptive Feature Learning Probability Boosting Tree Initialization and CNN-ASM Refinement". In: *IEEE Access* PP (Dec. 2017), pp. 1–1. DOI: 10.1109/ACCESS.2017.2781278.

[7] H. Jia et al. "3D APA-Net: 3D Adversarial Pyramid Anisotropic Convolutional Network for Prostate Segmentation in MR Images". In: *IEEE Transactions on Medical Imaging* 39.2 (2020), pp. 447–457. DOI: 10.1109/TMI.2019.2928056.

[8] Davood Karimi et al. "Prostate segmentation in MRI using a convolutional neural network architecture and training strategy based on statistical shape models". In: *International Journal of Computer Assisted Radiology and Surgery* 13 (May 2018). DOI: 10.1007/s11548-018-1785-8.

[9] Q. Li et al. "Medical image classification with convolutional neural network". In: *2014 13th International Conference on Control Automation Robotics Vision (ICARCV)*. 2014, pp. 844–848. DOI: 10.1109/ICARCV.2014.7064414.

[10] AS Lundervold and A. Lundervold. “An overview of deep learning in medical imaging focusing on MRI”. In: *Z Med Phys* 29.2 (2010), pp. 102–127. DOI: 10.1016/j.zemedi.2018.11.002.

[11] Tomas Möller and Ben Trumbore. “Fast, Minimum Storage Ray-Triangle Intersection”. In: *Journal of Graphics Tools* 2 (Aug. 2005). DOI: 10.1145/1198555.1198746.

[12] Michael A. Nielsen. *Neural Networks and Deep Learning*. 2015.

[13] László G. Nyúl and Jayaram K. Udupa. “On standardizing the MR image intensity scale”. In: *Magnetic Resonance in Medicine* 42.6 (1999), pp. 1072–1081. DOI: 10.1002/(SICI)1522-2594(199912)42:6<1072::AID-MRM11>3.0.CO;2-M.

[14] Artem Oppermann. “Regularization in Deep Learning-L1,L2,and Dropout”. In: *Towards Data Science* (2020).

[15] Kim P. “Convolutional Neural Network”. In: *MATLAB Deep Learning*. Apress, Berkeley, CA, 2017. DOI: 10.1007/978-1-4842-2845-6_6.

[16] T Penzkofer and CM Tempany-Afdhal. “Prostate cancer detection and diagnosis: the role of MR and its comparison with other diagnostic modalities–a radiologist’s perspective”. In: *NMR Biomed.* 27.1 (2014), pp. 3–15. DOI: 10.1002/nbm.3002.

[17] Tekla S. Perry. “Move Over, Moore’s Law: Make Way for Huang’s Law”. In: *IEEE Spectrum* (2018).

[18] Dzung L. Pham, Chenyang Xu, and Jerry L. Prince. “Current methods in medical image segmentation”. In: *Annual Review of Biomedical Engineering* 2.1 (2000), pp. 315–337. DOI: 10.1146/annurev.bioeng.2.1.315.

[19] Cheng R et al. “Fully automated prostate whole gland and central gland segmentation on MRI using holistically nested networks with short connections”. In: *J Med Imaging (Bellingham)* 6.2 (2019). DOI: 10.1117/1.JMI.6.2.024007.

[20] D. Ravi et al. “Deep Learning for Health Informatics”. In: *IEEE Journal of Biomedical and Health Informatics* 21.1 (2017), pp. 4–21. DOI: 10.1109/JBHI.2016.2636665.

[21] AB Rosenkrantz et al. “Clinical utility of quantitative imaging”. In: *Acad Radiol* 22.1 (2015), pp. 33–49. DOI: 10.1016/j.acra.2014.08.011.

[22] Indolia Sakshi et al. “Conceptual Understanding of Convolutional Neural Network-A Deep Learning Approach”. In: *Procedia Computer Science* 132 (2018), pp. 679–688. DOI: 10.1016/j.procs.2018.05.069.

[23] Cootes T.F. and Taylor C.J. “Active Shape Models — ‘Smart Snakes”. In: *Hogg D., Boyle R. (eds) BMVC92*. Springer, London, 1992. DOI: 10.1007/978-1-4471-3201-1_28.

[24] Xiaoli Tang. "The role of artificial intelligence in medical imaging research". In: *BJR—OPEN* 2.1 (2020), p. 20190031. DOI: 10.1259/bjro.20190031.

[25] NJ Tustison et al. "N4ITK: improved N3 bias correction". In: *IEEE Trans Med Imaging* 29.6 (2010), pp. 1310–20. DOI: 10.1109/TMI.2010.2046908.

[26] YX Wang and CK NG. "The impact of quantitative imaging in medicine and surgery: Charting our course for the future". In: *Quant Imaging Med Surg* 1.1 (2011), pp. 1–3. DOI: 10.3978/j.issn.2223-4292.2011.09.01.

[27] Jeffrey C. Weinreb et al. "PI-RADS Prostate Imaging – Reporting and Data System: 2015, Version 2". In: *European Urology* 69.1 (2016), pp. 16–40. ISSN: 0302-2838. DOI: https://doi.org/10.1016/j.eururo.2015.08.052. URL: https://www.sciencedirect.com/science/article/pii/S0302283815008489.

[28] R. Yamashita et al. "Convolutional neural networks: an overview and application in radiology". In: *Insights Imaging* 9 (2018), pp. 611–629. DOI: 10.1007/s13244-018-0639-9.

[29] Lequan Yu et al. "Volumetric ConvNets with Mixed Residual Connections for Automated Prostate Segmentation from 3D MR Images". In: Jan. 2017.

[30] F Zabihollahy et al. "Automated segmentation of prostate zonal anatomy on T2-weighted (T2W) and apparent diffusion coefficient (ADC) map MR images using U-Nets". In: *Med Phys* 46.7 (2019), pp. 3078–3090. DOI: 10.1002/mp.13550.

[31] Y Zhu et al. "Fully automatic segmentation on prostate MR images based on cascaded fully convolution network". In: *J Magn Reson Imaging* 49.4 (2019), pp. 1149–1156. DOI: 10.1002/jmri.26337.

[32] Zhu et al. "Boundary-Weighted Domain Adaptive Neural Network for Prostate MR Image Segmentation". In: *IEEE Transactions on Medical Imaging* 39.3 (2020), pp. 753–763. ISSN: 1558-254X. DOI: 10.1109/tmi.2019.2935018. URL: http://dx.doi.org/10.1109/TMI.2019.2935018.