Automating the Detection and Classification of Coronary Stenosis using Deep Learning

Healthcare Engineering and Management Graduation Project 2022

Ahmed Mahmoud Fawzi   Adham Ashraf Khater   Amira Ahmad Noaman   Rawda Ahmed Rumaieh

***Abstract*— Coronary arteries are major blood vessels in your body, supplying blood to your heart. They make it possible for your heart to beat and pump blood throughout your body. So, any blockage in these arteries will most probably lead to heart failure or at least a heart stroke. Non-invasive imaging techniques, specifically Coronary CT angiography (CCTA), are usually preferred as the safest way to visualize the coronaries vasculature and assess stenosis. However, investigating stenosis location and sizes from CCTA is a complicated task that requires a considerable amount of experience. Therefore, doing this task manually is prone to subjective image interpretation and it's also time consuming. In this work, we propose a fully automated process that uses image processing techniques and a CNN model to extract the centerlines that helps in straightening the arteries which allowed us to develop a model that is able to detect the stenosis regions then classify the stenosis type. The deep learning models are the main scope of our work; however, our aim is to create a software as a service that will aid the doctors in an easy manner to finish the required task in much less time and in a cost-efficient way.**

Keywords-- Coronary, Arteries, Stenosis, Deep Learning, Convolutional Neural Network, Recurrent Convolutional Neural Network

I. Introduction

Investigating stenosis location and types in coronary arteries to detect coronary artery disease from CCTA is a complicated task that requires a considerable amount of experience. Therefore, doing this task manually is prone to subjective image interpretation and it's also time consuming.

Several methods have been proposed to automate the detection and quantification of stenosis. While few methods use image features and pattern recognition, the latest accomplishments helped in stenosis detection by accurate lumen segmentation and centerline extraction. However nobody succeeded in classifying the types of stenosis or even its size. Our approach is to find the centerline of the arteries using a CNN model, then using these points we straighten the arteries using image processing techniques and curved multiplanar reconstruction. Having the input of straightened arteries, the developed RCNN will be able to detect the stenosis regions and classify their types.

For our approach to detect and classify the stenosis two literature review papers were followed to allow us to implement such techniques. [1] was followed to implement the CNN model that allows us to extract the centerline. This paper clearly explained the idea of how to obtain direction vectors from each point on the centerline to the other and how the CNN model was implemented. The second Literature review paper followed [2] clearly explained that the idea of the centerline extracted from the former paper approach was used to construct stretched arteries using MPR and a R-CNN model was further discussed that allowed such approach to detect stenosis and its type.

II. Materials and Methods

Two different datasets are used, the first dataset for the centerline extraction part was collected by the company, it is basically the CT scan of 8 patients, including different modalities, each patient has a folder containing the DICOM volume along to other 4 folders corresponding to the 4 coronary vessels which are right coronary artery (RCA), left coronary artery (LCA), left anterior descending artery (LAD) and left circumflex artery(LCX). Each vessel file contains points annotated by doctors along with a reference file containing coordinates of points on the centerline of the whole vessel and an extra column including the radius. The other dataset used for the stenosis detection consists of 18 folders representing the patients and for every patient the following information is provided, per centerline point: (1) the (xCTA,yCTA, zCTA) world coordinates, (2) the AHA-segment number snrCTA (between 1 and 17), (3) the patient’s lesion number lnrCTA (lnrCTA = 0 if the point does not belong to a stenosis, lnrCTA > 0 if the point belongs to a stenosis, lnrCTA 6 N, N being the number of lesions present in the CTA reference standard), (4) the stenosis type t, and (5) the CTA diameter percentage stenosis gCTA.

The whole project is implemented using python.

A. *The Implementation process*

Our process begins with using the previously annotated centerline points to build a convolutional neural network model that is able to extract the centerline points in the future from any given point on a CT volume. These extracted centerline points are then used to straighten the artery to avoid working on the complicated vasculature, this is done using the multiplanar reconstruction technique. Following these steps, a recurrent convolutional neural network is built to detect the stenosis regions and types.

1) *Centerline Extraction:*

a) Pre-Processing of the data

The centerline points available were in millimeters so we had to convert them to pixel values, to be able to view their locations on the given DICOM volume.

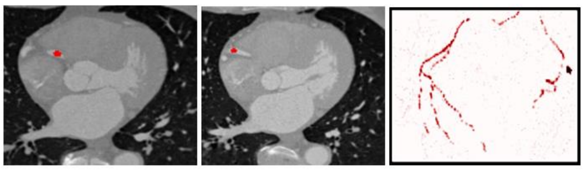


Fig. 1. Visualing Artries

b) Input

The receptive field of the CNN determines the part of the image that is taken into account to estimate the local orientation and radius. The choice of the receptive field size may affect the CNN's performance and based on previous work and experiments, a 19x19x19 volume is the most suitable receptive field for this problem. The center of the input volume is the current point on the centerline.

The proposed CNN learns to identify the coronary centerline orientations and lumen radius directly from image data only, without the need for intermediate hand-crafted vesselness representations. The network combines two tasks: direction classification and radius regression. The possible directions are distributed on a sphere, where each point corresponds to a class

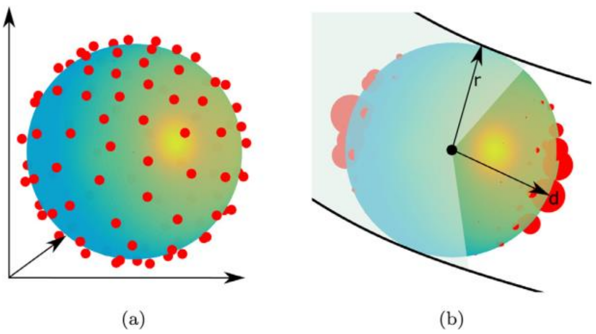
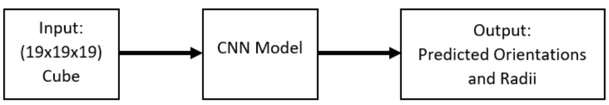


Fig from [1] The set of possible directions is distributed on a sphere.

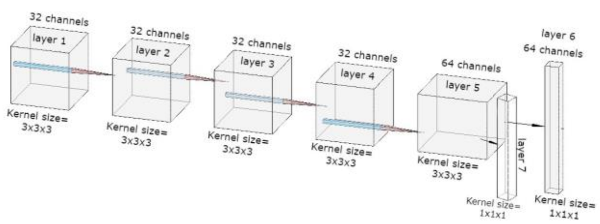
c) Output

The output during testing is posterior probability distribution over directions and a workflow of centerline tracking begins from this point, where we determine the most probable two directions that accompany the given point, convert them to unit vectors and check that the angle between them is more than 60˚ to make sure these directions are opposite. Each branch is then handled separately where we start passing each point in the 2 branches to the model, get a new direction, convert it to a vector and check that the angle between the current point and the new point is far by more than 60˚ than the vector from the current point and the previous point. This allows us to make sure that each new point is not in the same direction as the previous point. The process keeps iterating to get new points for the two branches until it terminates if a point was found to be out of the artery based on the entropy calculation.

Our data was divided into 3 sets
Training data set which consists of 6 patients – 6556 cubes
Validation dataset consists of 1 patient – 987 cubes
Testing dataset consists of 1 patient – 1107 cubes



d) CNN Architecture



For each layer, the convolution kernel width is listed, as well as the dilation level, the number of output channels, and the receptive field at the layer. All operations are performed in 3D. The number of output channels is equal to the number of potential directions in D, plus one channel for radius estimation.

Layer	1	2	3	4	5	6	7
Kernel width	3	3	3	3	3	1	1
Dilation	1	1	2	4	1	1	1
Channels	32	64	64	64	64	180	$\|D\|+1$
Field width	3	5	9	17	19	19	19

2) Artery Straightening

After being able to reslice the volume from different directions to achieve the different views and to trace the centerline itself, we wanted to be able to straighten the vessel to be able to easily detect stenosis regions using the model that will be developed in the next phase . To do so we had to follow the curved MPR technique to obtain the Normal, binormal, tangent vectors to be able to straighten the vessel. Normal vector represents the direction pointing directly "out" from a surface, meaning it is orthogonal (at 90 degree angles to) any vector which is coplanar with (in the case of a flat surface) or tangent to (in the case of a non-flat surface) the surface at a given point.

A Tangent vector is typically regarded as one vector that exists within the surface's plane (for a flat surface) or which lies tangent to a reference point on a curved surface (ie. if a flat plane were constructed with the same normal from the reference point, the tangent vector would be coplanar with that plane).
The concept of a Binormal vector is a bit more complex; in computer graphics, it generally refers to a Bitangent vector (reference here), which is effectively the "other" tangent vector for the surface, which is orthogonal to both the Normal vector and the chosen Tangent vector.
This technique is similar to a camera movement that tracks an object so these three vectors are needed to track the vessel as we change slices as the vessel is curved not straight and needs these vectors to determine the path of movement. After doing so and combining all the tasks assigned above we were able to construct a straightened vessel in different views to be able to detect stenosis easily.

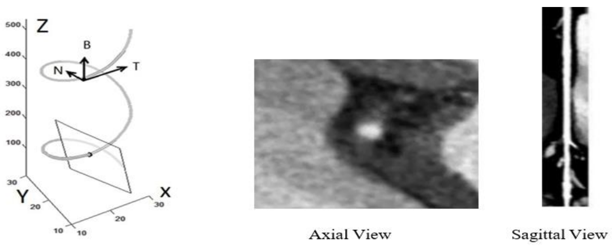


Fig. 2. MPR

3) Stenosis Detection & Classification:

a) Network Input

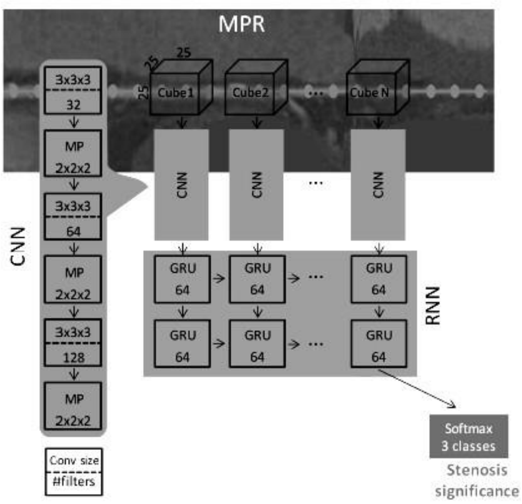
As explained previously, the dataset contains segments, each dataset is read separately then segments are combined to arteries in this manner:

- Artery1= segments [1,2,3,4]
- Artery2= [5,6,7,8]
- Artery3= [5,6,7,10]
- Artery4= [5,6,9]
- Artery5= [11,12]
- Artery6= [from 11 till the end of segments]

Some patients had missing segments, so we made a termination mechanism to exclude any messing segments. We then combined the segments into arteries. Arteries were combined along with their reference file so that each artery has its reference file. reference file includes the coordinates of the counterpoints of the artery with labels indicating the stenosis types. Since the distances were not equal we used a cubic spline interpolation to resample these center points. this is done by looping over the reference file to calculate the accumulated distance then creating an array with length equal to accumulated distance divided by the spacing we needed between each two points on the centerline. after creating an instance of cubic spline method we inputted the accumulated distance along with our original coordinates then applied the method over the array we created including points of equal spaces. Then we created an object from class cubic spline adding the accumulative distance and the actual centerline coordinates as arguments then applying the learned object on the set of points created. For the labels, we used the nearest neighbor interpolation with the same techniques used before. After We got the resampled centerline coordinates, we used them in creating the MPR of the DICOM volume. After straightening the MPR volume we divided this volume to set of cubes of sizes (25x25x25) using a stride of 5. We took the maximum label of these 25 frames and set it as a label representing the whole cube. Then we combined 10 cubes together to form a series where the maximum label of these 10 cubes represents the label of that series. This was done for each artery separately to preserve the entity of the artery. We then divided the dataset into: 12 patients as training dataset, 3 for validation, and 3 for testing.

b) Model Architecture

We used Hybrid model (RCNN) consisting of 3D CNN serving the feature extraction then RNN serving the classification task. The architecture is shown below:



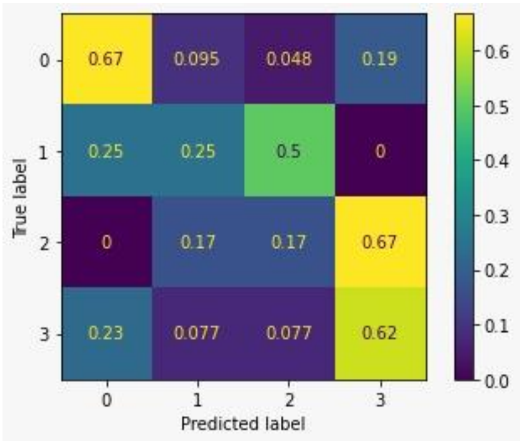
An MPR is obtained using the artery centerline points (dotted line). The input of the network is a sequence of cubes extracted from the MPR, along the artery centerline. A CNN extracts features out of 25 × 25 × 25 voxels cubes. The first convolution layer has 32 filters with kernel size of 3. The second layer has 64 filters and the third layer has 128 filters, both with kernel size 3. Subsequently, an RNN processes the entire sequence using gated recurrent units (GRUs). Each GRU layer had 64 units and we added a dropout of 0.5. The output of the RNN is fed into a softmax classifiers to characterize the stenosis. We have made a max-pooling layer after every convolution layer with pool size of 2. We also set the activation of every convolution layer to 'ReLu', and batch normalization was added after every layer of the convolution layer. The loss was categorical crossentropy and the optimizer used was Adam. As for the hyper parameters we have set the batch size equals 16 , the epochs equals 150, class weights equals: 0 for class zero, 1 for class one, 2 for class two, 3 for class three, and we also used early stopping with the val_loss as the monitor, patience equals 35, and setting the restore_best_weights to true.

c) Results

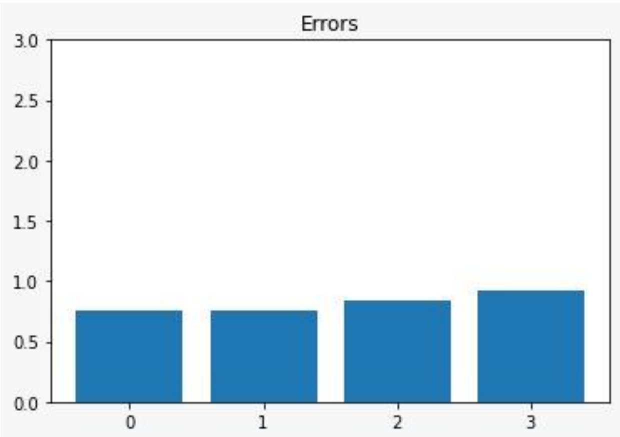
After training the model we calculated the error for each class through Getting the predicted labels and the ground truth labels. We then got the indices of ground truth representing a certain class, then we got the predicted labels at these indices. The error at this class is calculated by adding the absolute difference between each value in predicted labels and the class number (0, 1, 2, 3). This indicates how the model discriminate the classes. The error will be very high if the model was confused between 2 chosen classes. We also used 2 metrics which were accuracy and TopKcategorical. The metrics used were accuracy and TopKcategorical. TopKCategorical computes how often the targets are in the top K predictions where K is equal to 2 categories. Since we are having class in-balance we needed the classifier to heavily weight the few examples that are available, we determined the weights through counting the number of this class in the training labels then Using this formula:

(1/total amount of this class in the training labels) x (total amount of training labels/2)

We then displayed the confusion matrix to know how well our model did. Below is the confusion matrix:



The model scored 67% of class 0 correctly, 25% of class 1 correctly, 17% of class 2 correctly and 62% of class 3 correctly. the reasons behind such errors were our dataset was small, so our model wasn't able to train good enough on discriminating between class two and class one. We also computed the errors in each class as shown below:



After calculating the errors, we displayed the model's testing accuracy and loss as shown below

```
Test loss: 1.9492888450622559
Test accuracy2: 0.6590909361839294
Accuracy Score:  0.4772727272727273
```

the model had training accuracy of 0.53, loss of 2.7 , training top_k_categorical_accuracy: 0.9866, validation top_k_categorical_accuracy: 0.7317

We then tested the model's ability to label an artery according to our 4 classes and compared it with the ground truth and these were the results:

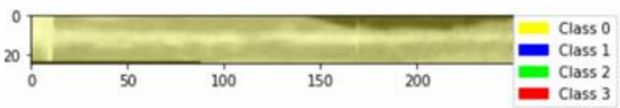


Figure 3: test_artery1,stenosis class - ground truth

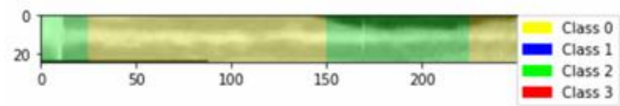


Figure 4: test_artery1,stenosis class - model prediction

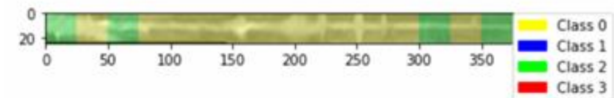


Figure 5: test_artery2, stenosis class - ground truth

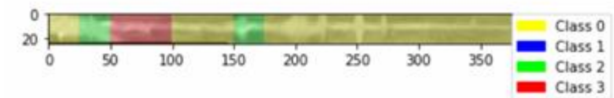
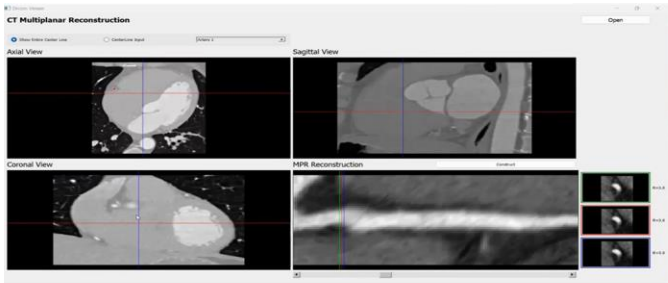


Figure 6: test_artery2, stenosis class - model prediction

III. GUI

To visualize the centerline and MPR a GUI was developed to allow the user to better interact with the different views and construct MPR. The GUI allows the user to upload a dataset generating different views of the dataset axial, sagittal, and coronal for better visualization. The GUI also has an option that allows the user to view the centerline of the arteries. It also lets the user to choose which artery to view and construct MPR for that artery allowing the user to visualize the straightened artery.



IV. CONCLUSION

A method for automatic detection and characterization of the anatomical significance of the coronary artery stenosis was presented. The method employs an RCNN that analyzes an MPR view of a coronary artery extracted from a CT scan using the coronary artery centerline. proposed method requires only the coronary artery centerline as an input along with the CT volume.

Since manual annotation is burdensome, we developed a CNN model to automatically computes the centerline without the Dr's annotations. Due to the GPU limitations, we couldn't reach the number of iterations that the papers we were inspired by have reached"50,000" we also lacked the sufficient amount of data since we only worked with 18 patients while the other papers have exceeded 146 patients. We had also suffered from poor quality of data.

We have constructed a good infrastructure where future improvements can be made to enable an automated triage of patients to those without coronary stenosis and those who with stenosis in need for further cardiovascular investigation.

V. REFERENCES

1] J. M. Wolterink, R. W. van Hamersvelt, M. A. Viergever, T. Leiner, and I. Išgum, "Coronary artery centerline extraction in cardiac CT angiography using a CNN-based orientation classifier.," Med. Image Anal., vol. 51, pp. 46–60, Jan. 2019.

2] Majd Zreik , Robbert W. van Hamersvelt, Jelmer M. Wolterink , Tim Leiner, Max A. Viergever, and Ivana Išgum " A Recurrent CNN for Automatic Detection and Classification of Coronary Artery Plaque and Stenosis in Coronary CT Angiography.," Ieee Transactions On Medical Imaging, Vol. 38, No. 7, July 2019