IMA 205 – Challenge: Report

Antoine Andurao

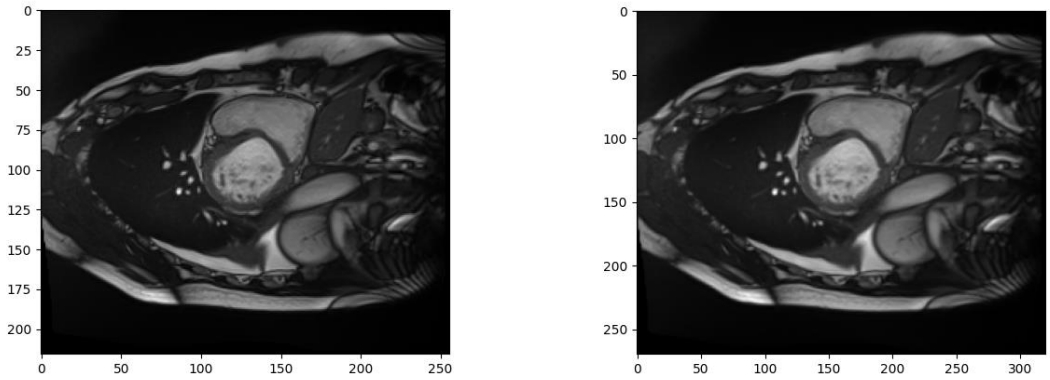
Introduction

This goal of this project was to find the best method for classification of MRI images of the heart among five different diagnostic classes: 1 healthy and 4 representing different diseases or abnormalities. I relied on the paper *Automatic Cardiac Disease Assessment on cine-MRI via Time-Series Segmentation and Domain Specific Features* [1] to find a workaround for the segmentation and the feature extraction.

Data preprocessing

As mentioned in [1], the Nifti1 images require a bit of preprocessing. I chose to apply the exact same preprocessing as the paper (for 3D UNet segmentation and feature extraction), i.e. normalizing and resampling (to match 1.25×1.25×10 mm voxel size) all Nifti1 images.

In the notebook, two functions have been used for preprocessing: *normalize_and_resample* for Nifti1 images of MRI images, and *resample* for Nifti1 images of segmentation masks (since there is no point in normalizing masks)



Comparison between MRI images before (left) and after preprocessing (right). The effect of normalization isn't very distinguishable in the pictures, but we can notice that image was indeed rescaled to match a certain voxel size.

Segmentation of left ventricle cavity

I tried to achieve the segmentation of the left ventricle cavity, following the works of [1]. I chose to implement the architecture of 3D UNet. This particular architecture was designed to handle the specificities of the dataset: upscaling and pooling operation are performed only on x-y axis because of the lower resolution on the z-axis.

Using the model as provided by the paper wasn't possible for me, since it required the optimization of more than 14 million parameters. I decided to try and scale down this model by removing symmetric upscaling and pooling layers. Getting down to three layers of pooling

(26 to 104 filters) and two layers of upscaling (52 to 26 filters), I managed to reduced the number of parameters to under 200 000. Nonetheless, as I was running out of time, I couldn't try this method on machines with adequate computational capacities. I decided to give up on the segmentation, and use only the partial segmentation given in the dataset.

Feature extraction

For feature extraction, I also relied on [1] to find meaningful features. For the right ventricle cavity and the myocardium, I extracted the following features: circumference, circularity, thickness and area. These features are local descriptors of the shape of the right ventricle and the myocardium: in the presence of abnormality and/or disease, I though that these features should vary enough to allow classification. It must be noted that those descriptors were computed on the x-y plane (thus on each "sub image" of one Nifti1 image) and aggregated over the z-axis. Those 4 descriptive features are computed on the MRI of the same heart at end diastole and end systole, giving a total of 16 features.

I decided to add on last feature that is mentioned in [1]: body surface. The body surface is a feature that resemble the body/mass index, and that makes use of the height and mass of the subject. Since we know that obesity increases the risk of heart and circulatory diseases, I thought that body surface could be a relevant feature (computed using Mosteller formula).

All these summed up to 17 features for each MRI (ED and ES combined).

Classification

Since the problem is a multiclass classification problem, I first thought of **k-NN**. I used GridSearch to find the best parameters, and got **0.65** as the **best validation score** (with 10 neighbors). I didn't try to submit a result to get a test score with k-NN, because I first wanted to see if other methods could yield a better validation score.

Thus, I tried **SVM** as separation technique. Since the data wasn't linearly separable, I used a **non-linear kernel**. SVM yielded a slightly better validation score than k-NN: **0.67**. Using SVM on the test data, I got an accuracy of 0.667 (public score, computed on approx.. 20% of the test data, since it was the only available at that time).

Finally, I wanted to try a more fine-tuned method for multiclass classification. Since separating the data (with SVM and k-NN) only got me to 0.667 of accuracy, one other multiclass method that came to my mind was Decision Tree. Considering the results I found on the previous lab on Decision Trees, I decided to skip ahead and try directly Random Forests.

Using GridSearch and Cross-validation to find the best parameters among: the maximum number of features that are considered for splitting, the minimum number of samples required to be at a leaf node and the number of decision trees in the random forest. Using RF with the parameters found by GridSearch and Cross-Validation, I got a validation score of **0.62** and a test score of **0.733** (public Kaggle score).

Since I ran out of time, I didn't try any classification method using ML and/or neural networks. Such methods could yield better results, yet the relatively small size of the dataset could be an obstacle without data augmentation.

Bibliography

[1]: F Insensee et al.: *Automatic Cardiac Disease Assessment on cine-MRI via Time-Series Segmentation and Domain Specific Features*