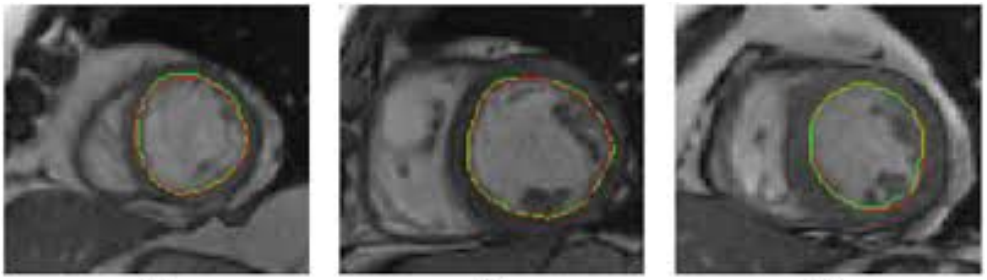

(a) (b) (c)

Variational Auto-Encoders for cardiac shape modeling

Juin 2023

Members :
Kelthoum Kerboua
Cécile Tillerot
Fatima Baldé
Manon Heffernan

Teacher :
Loic Le Folgoc, Elsa Angelini

Contents

I Introduction

Deep learning methods present remarkable results in the field of medical imaging, notably on diagnosis-related tasks such as classification. They can be a powerful tool to support diagnosticians in their work, by hinting at different interpretations and improving the speed – and perhaps even the quality – of the diagnosis.
Classification algorithms rely on feature extraction to accomplish the classification tasks, unfortunately those features may lack interpretability and therefore limit the diagnostician's understanding of the decision criteria. The stake here is to improve the interpretability of the features, to improve the clinical value of deep learning classification methods, beyond the classification output itself.
Depending on the architecture, a relevant option is to explore the feature space in order to find the most informative and insightful features, from a clinical point of view.
In this report, we focus on a variational autoencoder (VAE) architecture used for classification of cardiac diseases, proposed in Learning Interpretable Anatomical Features Through Deep Generative Models: Application to Cardiac Remodeling [1].

II Approach

This project was programmed in Python, using PyTorch as the deep learning framework. The data consists of cardiac MRI segmentation, from the ACDC Dataset from the MICCAI challenge (2017). The first step consisted in preprocessing the data, to align and focus on the regions of interest (ROIs).
Then, we decided to build the VAE architecture ourselves, in order to have a deeper understanding of the structure and better grasp the impact of the different elements influencing the final outputs.

II.1 Preprocessing

The ACDC Dataset consists of cardiac MRI images, split into training and test sets of 100 and 50 patients respectively. Each set is evenly divided into 5 classes, 1 healthy (NOR) and 4 pathological: previous myocardial infarction (MINF), dilated cardiomyopathy (DCM), hypertrophic cardiomyopathy (HCM) and abnormal right ventricle (RV).
For each patient, we focused on the segmentations at end-diastole and at end-systole. For each patient and stage of the cardiac cycle, we have a 3D image of the heart, consisting of a variable number n of slices of dimension 2. Each mask is represented by a number on our segmentations: (0: background, 1: left ventricle, 2: myocardia, 3: right ventricle). In this project, we are working on the reconstruction of these 2D slices using Variational Autoencoder. To achieve this, each slice goes through a number of pre-processing steps.

Although MRIs are standardized examinations, multiple external factors mean that our heart images do not have the same orientation from one patient to the next. So, in order to reduce variability at the input to our Variational Autoencoder, we decided to force this orientation. To do this, we calculate the barycenters of the left and right ventricle masks, and rotate the image so that the straight line passing through these two points is vertical. This results in a high number of segmentation sequences with the same orientation.

Another problem is the imposing presence of the background, which takes up more space than the region of interest, which is the heart. Keeping the entire background of the image is pointless, and the variational autoencoder would perform better if it were given images such that the region of interest covered it entirely. To achieve this, we perform a cropping operation to remove the background with a little extra padding, determining the smallest non-zero pixel index and the largest, and cropping the image accordingly.

The 2D images we obtain will be sent to the variational autoencoder. This requires us to set them to the same size, which is not the case (before and after cropping). We have chosen to set all images to 80x80.

Aligning and resizing the images involves interpolating the values of certain pixels. However, our image can only contain the values 0, 1, 2 and 3. We therefore specify that the interpolation chosen is the nearest neighbor interpolation. Any other interpolation leads to the appearance of new masks in previously empty areas (e.g. the boundary between myocardium (2) and background (0) becomes left ventricle (1, average of both)).

We then choose to transform our 1-channel 2D images into a 4-channel image. Channel number i takes the value 1 if the pixel has value i, 0 otherwise. So, for each pixel, the sum of the channels is equal to 1, and each channel takes on a binary value.

Once we've obtained our list of images with the right shape, we redefine and create a DatasetMRI class, inherited from the torch.utils.data.Dataset superclass, which will later enable us to define two dataloaders (for the training and testing sets), which return batches of 50 randomly chosen images.

II.2 Variational Autoencoder

The variational autoencoder architecture described in the article Learning Interpretable Anatomical Features Through Deep Generative Models: Application to Cardiac Remodeling [1] is the following:

We started by reimplementing this architecture, using the indicated kernel sizes,

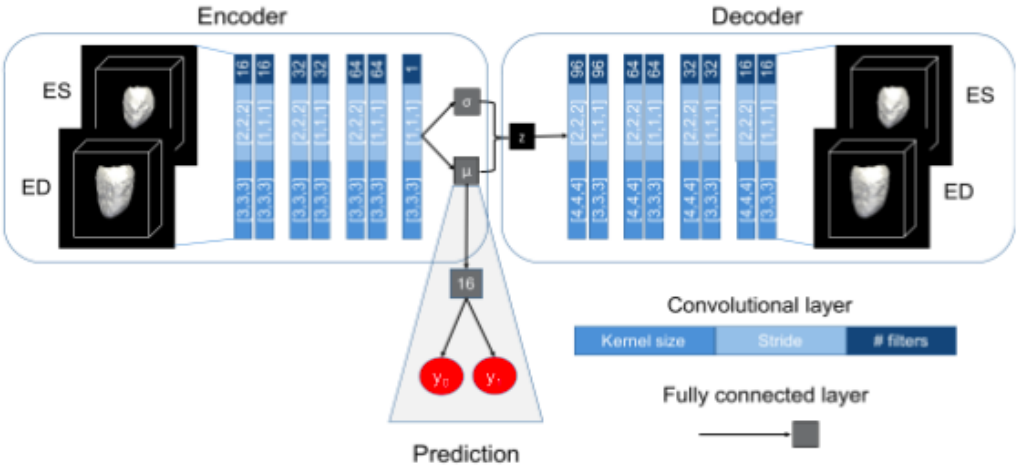

Figure 1: VAE model

strides and number of filters indicated, adapting their dimension to that of our input (slices instead of 3D images as in the article). The MLP layer, which is used for the classification, was left out at first, and we solely focused on the VAE itself.

The architecture being asymmetrical, we adapted the structure to reestablish a symmetry for the sake of simplifying the dimension reshaping for the encoder output and decoder input. The encoder consists of successive convolutional layers using Conv2d for the encoder, ConvTranspose2d for the decoder, both coming from the PyTorch library. Padding was manually added to keep the dimensions at the right value throughout the different convolutional layers. We determine the output size of the various layers from the input size, core size, stride and padding using a very simple formula for Conv2D:

$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

n_{in}: number of input features
n_{out}: number of output features
k: convolution kernel size
p: convolution padding size
s: convolution stride size

Figure 2: Formula for image size computation

The architecture of the variational autoencoder we have built is described in the figure below. We use ReLU activation between convolutional layers and then Softmax on the different channels, giving us the probability for each pixel that it belongs to each region.

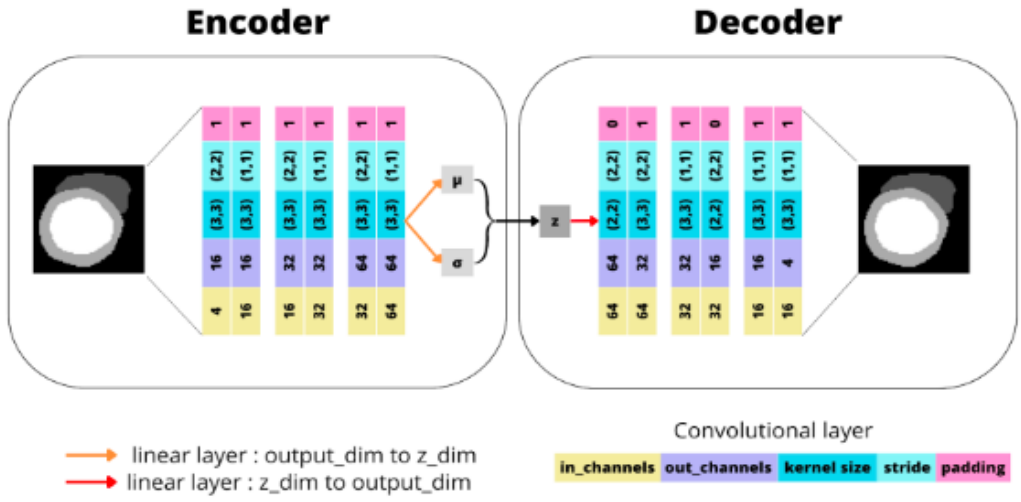

Figure 3: VAE model

III Loss function

We have defined a loss function for our variational autoencoder, composed of two terms:

- KLD (Kullback-Leibler divergence):

  This is a measure of dissimilarity between two probability distributions. Here, we attempt to approximate the encoder outputs μ and σ to 0 and 1. Thus, this term of the loss function guarantees that at the end of training, sampling latent variables with a centered reduced Gaussian distribution, and passing the variable through the decoder, we obtain a realistic image of a heart. The KLD term is directly related to the generative power of the variational autoencoder.

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \simeq \frac{1}{2} \sum_{j=1}^{J} \left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right)$$
$$\text{where} \quad \mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)} \quad \text{and} \quad \boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I})$$

Figure 4: KLD loss

- Dice Loss: This is a statistical indicator developed by Sorensen and Dice, which calculates the similarity between two samples, here of masks. You can find the formula for the Dice Loss function below. As you can see, the denominator considers the number of pixels from a global point of view, while the numerator

evaluates the overlap from a local point of view. This provides a more accurate comparison than cross-entropy, which only provides a local evaluation. This is very important, as our three masks are of very different proportions. Forgetting the overall shape of each mask may introduce a bias and prioritize masks of greater proportion (e.g. the background).

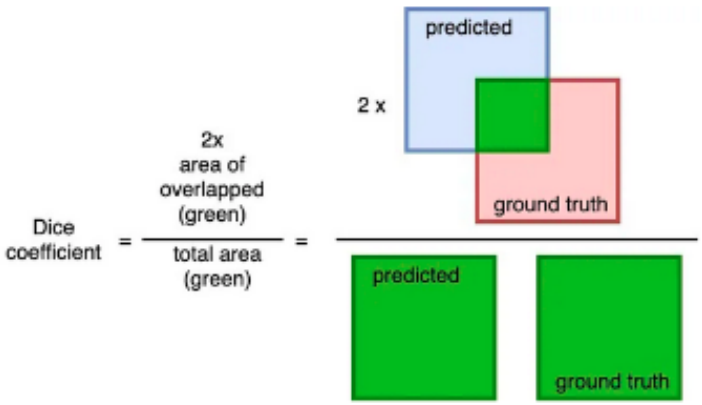

Figure 5: Dice loss

Care is taken to ensure that the reduction (sum or average) of each of the terms in the loss function is identical. We have chosen to reduce according to the sum. To improve reconstruction at the output of the variational autoencoder, we add a coefficient β in front of the KLD that is less than 1. If it is too low, we risk losing the generative power of the variational autoencoder. A good compromise would be 0.03 (for training with image batches of size 50).

IV Training

The latent space dimension is 32. We have chosen to reduce the dimension in relation to that of the article. In fact, the dimension of the article was 64, for 3D images. We decided that it wasn't necessary to keep such a large dimension for 2D images, and that, on the contrary, there was a risk of overfitting.

Our model is trained over 200 epochs with batches of 50 images. We used bigger batches as in the article (16) because we had 2D images, so less pixels to reconstruct. We used Adam optimization, a stochastic gradient descent method, with a learning rate of 10-4 and a weight decay of 0.001. We ran our project on Google Colab to gain access to the platform's GPUs, which enabled faster execution. Below are the curves that show the convergence of the difference losses for our parameters :

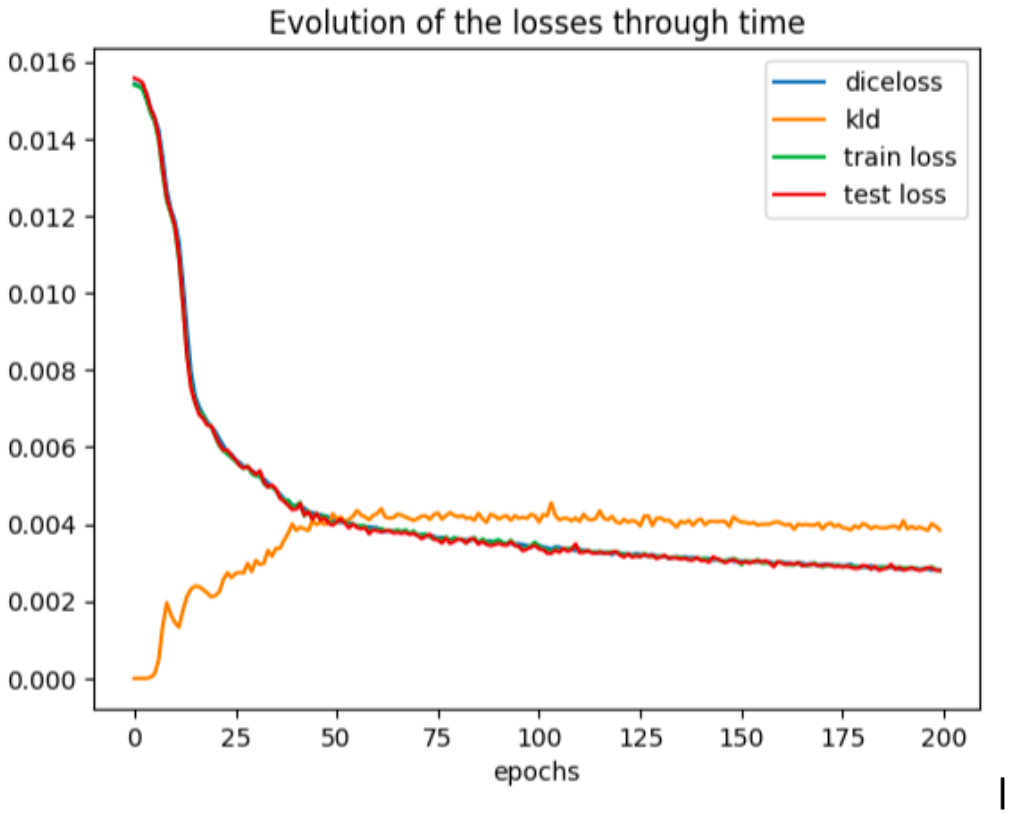

Figure 6: Evolution of the losses through time

V Results

V.1 Reconstitution after Variational Autoencoder

Below, on the first line, you can see the test set images supplied to the variational autoencoder, and on the second line the matching outputs.

The outputs look a lot like the input to the human eye, although they are smoother, because they did not go through the realignment steps that included interpolation. The most important features for disease diagnosis, the left and right ventricle volume and the myocardium thickness seem to be preserved after going through the VAE.

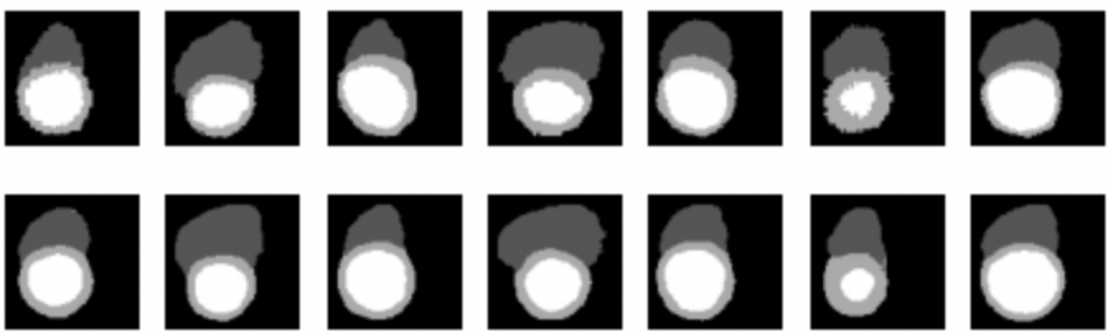

Figure 7: Test images and vae outputs

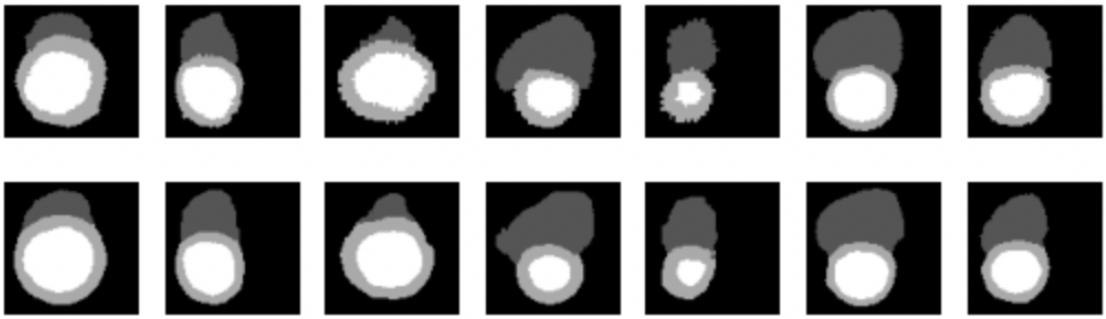

Figure 8: Test images and vae outputs

V.2 Generative power

Below, you can see the images generated by the variational autoencoder, based on latent variables randomly sampled according to a centered-reduced Gaussian distribution.

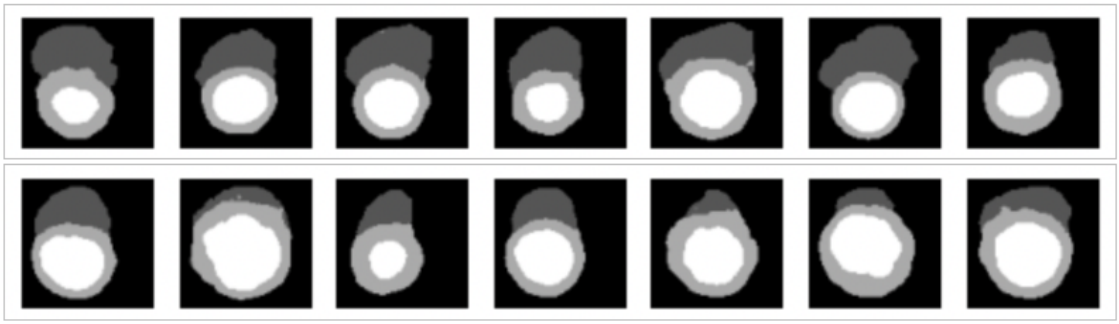

Figure 9: Latent vectors used as inputs for vae

All shapes seem realistic, and slices of different indexes can be generated (the sixth image would probably be at a different vertical index than the fourth one). There are a few artifacts, e.g. not being able to have one continuous shape for the myocardium on the sixth image, but they are minimal.

VI Analysis and comments

VI.1 Distribution of μ and σ and sampling

We checked that the encoder outputs μ and σ, were close to 0 and 1, and therefore that the distribution of latent variables approximates a Gaussian distribution of mean 0 and variance 1. We then displayed the histogram of each of the latent variable components, obtained from the test images. the results are similar for each of them, as follows:

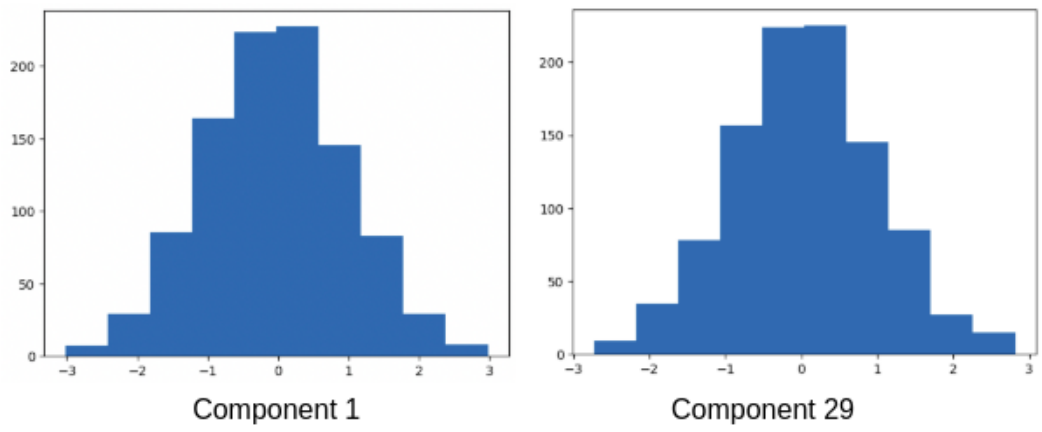

Figure 10: Sampling distribution

VI.2 Influence of the hyperparameter β

As explained earlier, we have introduced a hyperparameter β into our loss function, which takes the form: β*KLD + Dice Loss. Thus, when $\beta<1$, we reduce the importance of the KLD term (responsible of the distribution of latent variables and therefore of the variational autoencoder's generative power) and prioritize the Dice Loss term (responsible for input image reconstruction). When $\beta>1$, the opposite occurs. Below, we display the curves of the Dice Loss and KLD functions during the training of our model, for different values of β.

We can see that when β is very small, Dice Loss is reduced as a priority, but KLD is not reduced enough, and ends up stagnating. Whereas when β is very large, the opposite situation appears: the KLD remains constantly zero but the Dice Loss is not reduced enough.

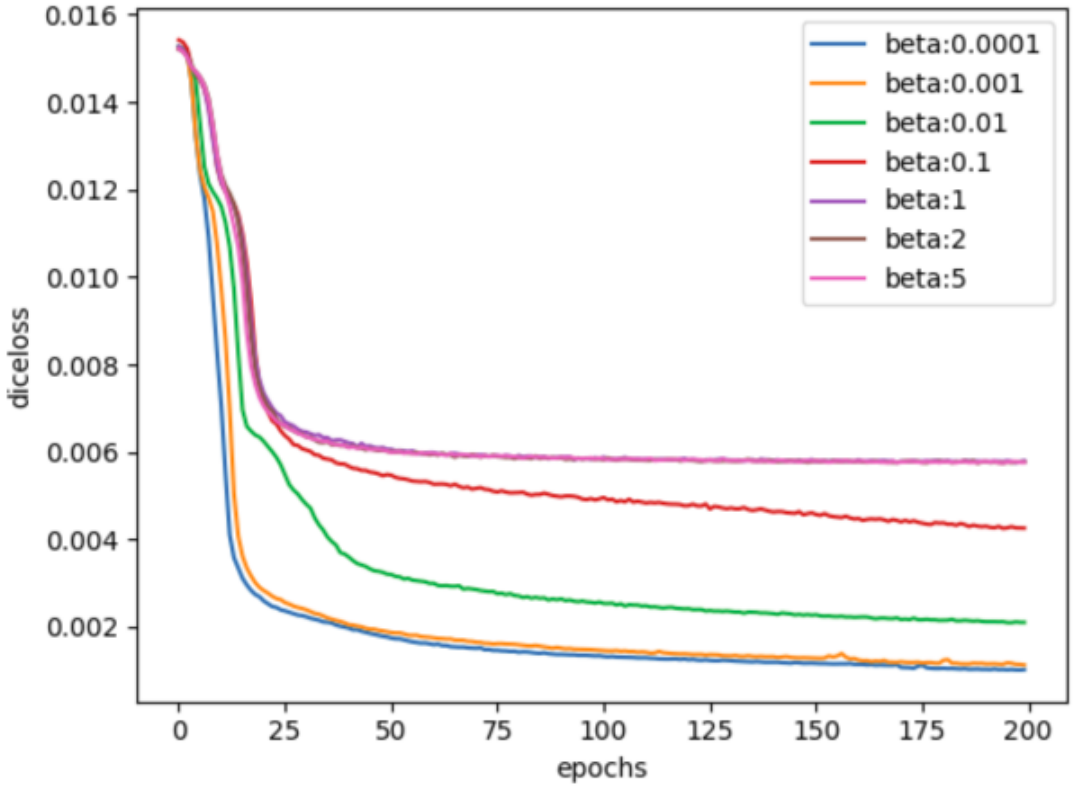

Figure 11: Dice loss

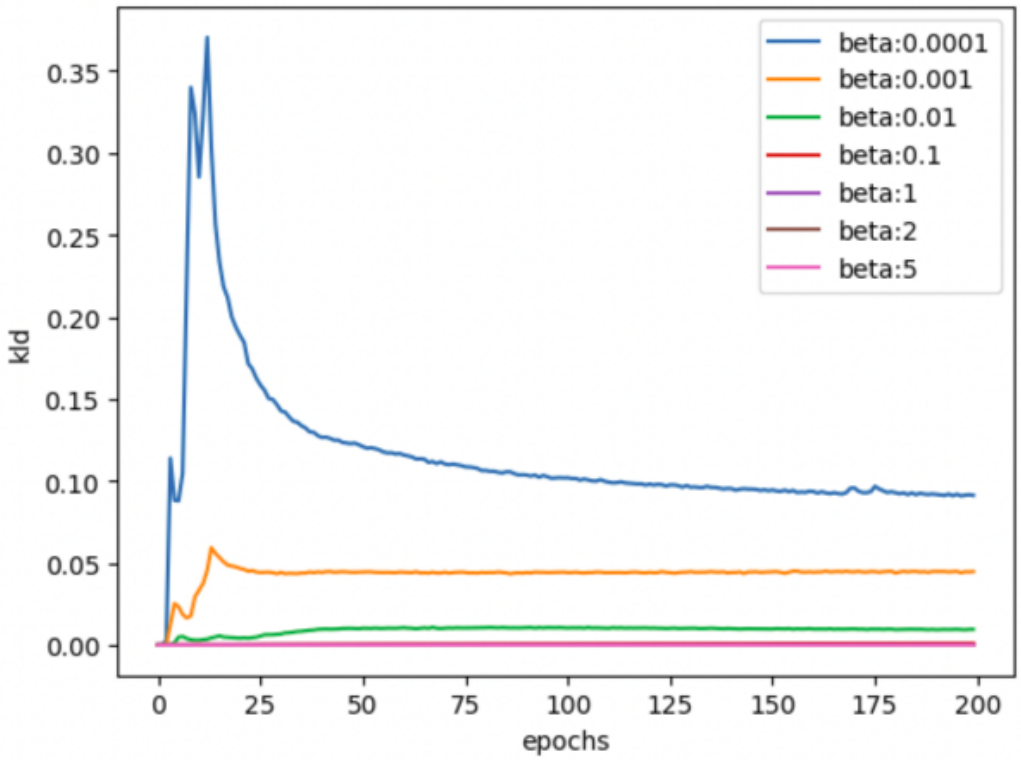

Figure 12: KLD loss

VI.3 Exploration of the latent space

Finally, we wanted to understand the signification of latent variables and the information they may contain, which, being essential for image reconstruction, could

potentially be used for medical purposes.

We can therefore focus on the volumes of the various regions of interest in the original 3D image of the heart, their areas on the selected slice as well as the slice index (between 0 and 1), which tells us which part of the heart we are looking at (end, center, etc.). To calculate the volumes of the various ventricles and myocardium, we used the images before preprocessing and the information contained in the headers of the Nii files, indicating the volume of the pixels etc... The information (volumes, areas, indices, etc.) is contained in the labels associated with the images, returned by the dataloaders created in pre-processing.

To understand the influence of this information on the latent variables, we aim to reduce their size from a 32 component vector to 2D points, which are easier to display and contain the most variability. Several well known dimension reduction algorithms were used: TSNE from the sklearn library, UMap from the umap library of the same name, and FastICA, also from the sklearn python library. For the TSNE algorithm, the parameters are defined as follows: perplexity=50, learning rate=10, ncomponents=2, and for the UMap algorithm: nneighboors=500, mindist=1, ncomponents=2.

Once we've obtained our 2D points, we display them with a color code associated with the value of the label we're studying. The latent variables contain more information and are more influenced by the area of the different regions on the slice than by the total volume of the region in the core, which is explained by the fact that the variational autoencoder only works on slices. However, the latent variables are not totally independent of volume, as shown in the display below, which is explained by the fact that the area of the ROIs in the slice is itself dependent on total volume. Slice index does not have a significant influence on latent variables. In fact, two slices with different indices can be very similar. However, the extremities of slices with indices close to 0 or 1 imply very small ventricle areas.

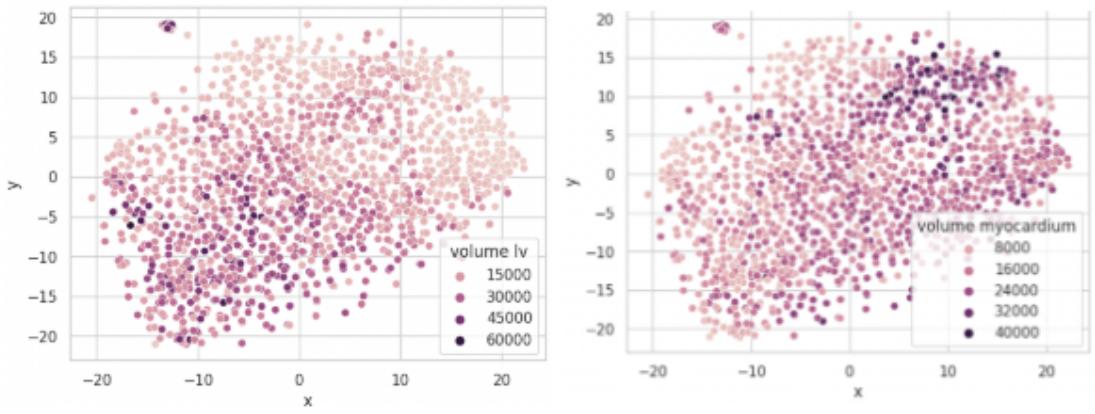

Figure 13: Volume LV and myocardium

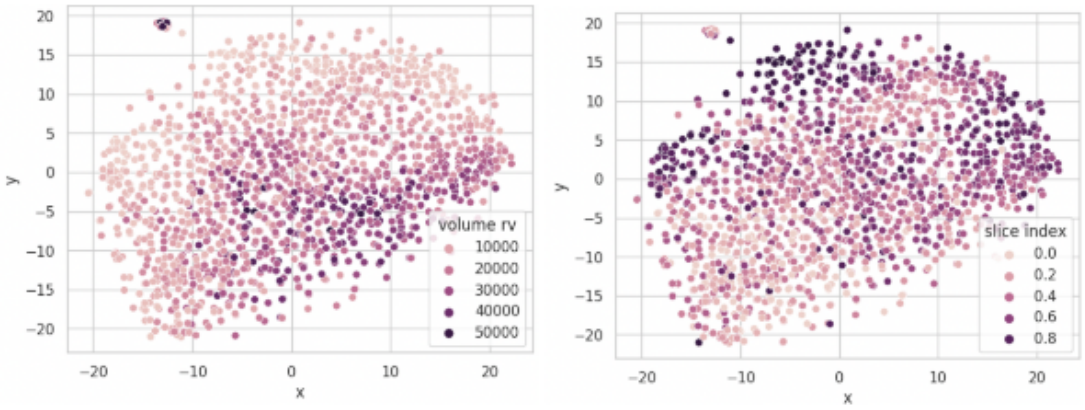

Figure 14: Volume Lv, Slice index

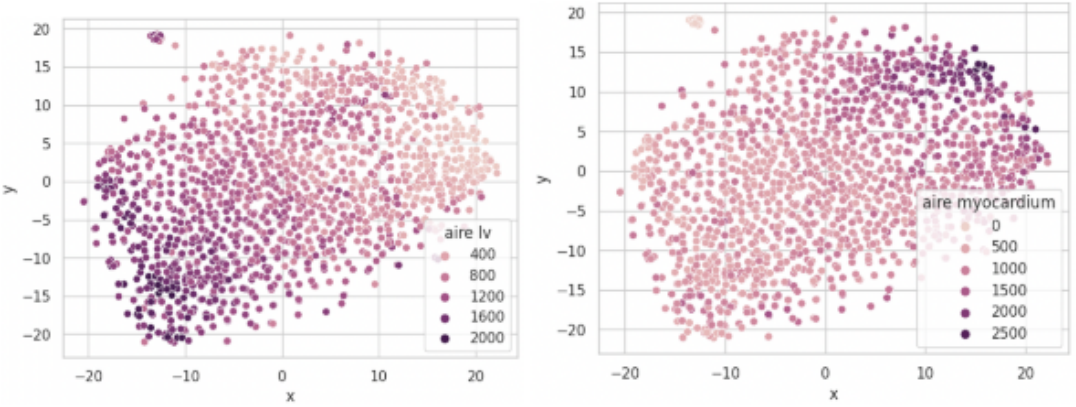

Figure 15: Aire Lv , Aire Myocardium

Figure 16: Aire rv

VII Conclusion

Our main objective was to build a variational autoencoder for 2D heart MRI segmentation slices, and subsequently to analyze its performance and explore the underlying latent space.

With a multi-stage pre-processing process, an architecture composed of several well-placed convolutional layers, a well-defined loss function and well-chosen training parameters, we obtained a variational autoencoder with remarkable image reconstruction and good generative power .

We were able to understand the influence of specific hyperparameters on its performance, as well as the importance of certain core and image characteristics on latent variables.

VIII Bibliographie

REFERENCES

1. Biffi, C., Oktay, O., Tarroni, G., Bai, W., Marvao, A. D., Doumou, G., ... Rueckert, D. (2018, September). Learning interpretable anatomical features through deep generative models: Application to cardiac remodeling. In International conference on medical image computing and computer-assisted intervention (pp. 464-471). Springer, Cham.

2. O. Bernard, A. Lalande, C. Zotti, F. Cervenansky, et al."Deep Learning Techniques for Automatic MRI Cardiac Multi-structures Segmentation and Diagnosis: Is the Problem Solved ?" in IEEE Transactions on Medical Imaging, vol. 37, no. 11, pp. 2514-2525, Nov. 2018. doi: 10.1109/TMI.2018.2837502