Projet PRIM 2024–2025:
NVAE in cardiac imaging

Alexandre Myara

26/02/2025

Contents

1 Introduction

1.1 Motivation

Les maladies cardiovasculaires (MCV) constituent un enjeu majeur de santé publique, représentant l'une des principales causes de mortalité à l'échelle mondiale.

Une part importante de ces maladies est liée aux dysfonctionnements cardiaques et vasculaires, notamment aux infarctus du myocarde et aux accidents vasculaires cérébraux. La détection précoce des anomalies cardiaques joue un rôle clé dans la prévention et le traitement des MCV, permettant d'améliorer le pronostic des patients et de réduire la mortalité prématurée.

L'imagerie par résonance magnétique (IRM) cardiaque est la référence pour l'évaluation des structures du cœur, en offrant des images détaillées du myocarde et des cavités ventriculaires.

Une segmentation précise des structures cardiaques, notamment des ventricules gauche et droit ainsi que du myocarde, est essentielle pour quantifier les paramètres fonctionnels du cœur et identifier d'éventuelles pathologies. Toutefois, la segmentation manuelle reste une tâche complexe et chronophage, sujette à des variations intra- et inter-observateurs.

L'essor des techniques d'apprentissage profond a permis d'automatiser en partie la segmentation des images médicales. Des architectures de réseaux neuronaux comme U-Net [1] et ses variantes comme Attention U-Net [2] ou Swin UNETR [3] ont montré des performances prometteuses. La piepeline actuelle consiste en l'utilisation de volume 2D ou 3D utilisés pour produire des masques de segmentation en sortie de modèle. Cependant, ces modèles restent limités par leur sensibilité aux variations anatomiques et aux artefacts d'imagerie, ce qui nécessite encore des corrections manuelles de la part des experts.

Dans ce contexte, les autoencodeurs variationnels (VAE) ont émergé comme une approche intéressante pour apprendre des représentations compactes et structurées des formes cardiaques. Ils permettent non seulement d'améliorer la segmentation en capturant des structures latentes cohérentes, mais aussi de générer des formes plausibles pour la correction des prédictions erronées. Les modèles VAE hiérarchiques, et en particulier le Nouveau VAE (NVAE) [4], se distinguent par leur capacité à générer des représentations détaillées et cohérentes, même sur des données à haute résolution.

1.2 Base de code

Ce projet démarre sur l'utilisation de la base de code de M. Freddy Jiang. Nous nous inspirerons en particulier son implémentation du modèle Nouveau VAE ainsi que du formatage des données qu'il a réussi à mettre tout au long de son

projet.

A la suite de cette première base de code, les différentes implémentation nécessaires à la réalisation de ce projet se trouve quand à elle sur une seconde base de code disponible à l'adresse git `https://github.com/alexandremyara/prim-nvae-2024`. Le projet git s'organise en 2 dossiers:**shape** (base de code M. Freddy Jiang), **prim** (base de code développée sur ce projet).

1.3 Objectifs

L'objectif de ce travail est d'explorer l'intégration du Nouveau VAE (NVAE) dans un pipeline de segmentation cardiaque afin d'améliorer la qualité des segmentations obtenues à partir d'approches classiques basées sur des réseaux convolutifs. La segmentation des structures cardiaques, notamment les ventricules gauche et droit ainsi que le myocarde, est une tâche essentielle en imagerie médicale, en particulier pour l'analyse des IRM ciné cardiaques. Cependant, les méthodes de segmentation actuelles sont sujettes à des erreurs anatomiques et à des imprécisions, nécessitant des corrections manuelles.

Dans ce contexte, nous proposons d'évaluer le NVAE comme un modèle génératif capable d'affiner et de restaurer les segmentations imparfaites produites par des modèles de segmentation classiques. Cette étude repose sur les étapes suivantes:

1. **Entraînement initial du NVAE sur un ensemble de masques segmentés**

    - Objectif:Comprendre le comportement du modèle et optimiser ses hyperparamètres.

    - Entraînement du NVAE sur des masques binaires de segmentation extraits d'IRM cardiaques.

2. **Entraînement d'un ensemble de modèles de segmentation et génération d'un jeu de données de segmentations imparfaites**

    - Comparaison de plusieurs architectures de segmentation, notamment:

        - UNet

        - UNet avec ResBlock

        - Attention UNet

        - Swin UNETR

    - Génération d'un jeu de données de segmentations imparfaites à partir des prédictions de ces modèles, servant de données d'entrée pour le

NVAE en post-traitement.

3. **Intégration du NVAE en aval du pipeline de segmentation**

   - Exploration de deux approches:

     (a) **Pipeline classique**:Le NVAE est utilisé comme un correcteur des erreurs de segmentation.

     (b) **Pipeline avec 3-gram**:Implémentation d'une approche basée sur des contraintes locales pour guider la correction du NVAE.

   - Évaluation des améliorations apportées par le NVAE sur la cohérence anatomique des segmentations et sur la réduction des erreurs structurelles.

L'ensemble des expérimentations sera évalué selon des métriques de segmentation standard telles que le Dice Score et la validité anatomique des segmentations. L'objectif final est de démontrer si le NVAE peut apporter un gain significatif en précision et en robustesse par rapport aux méthodes de segmentation traditionnelles, et ainsi **réduire la nécessité de corrections manuelles en imagerie médicale**.

2 Presentation des architectures

Au long de cette dissertation nous nous appuyerons sur une succesion d'architecture. En particulier deux familles de modèles seront envisagés dans nos expériences. Les modèles génératifs profond avec notamment les architectures type Variational Auto-Encoders puis les modèles de segmentation en U.

2.1 Modèles génératifs

Afin d'introduire les forces et faiblesses de l'architecture principale de cette experience (le Nouveau VAE), nous explorons les architectures précédants ce modèle.

2.1.1 Auto-Encodeurs

Les réseaux Encodeur-Décodeur sont des réseaux de neuronnes adaptés aux tâches de transformation de séquences en séquences. Ils se composent de deux composants principaux:

1. **Encodeur**:traite l'entrée et crée une représentation de taille fixe (appelée un embedding) dans un espace latent, visant à capturer les informations essentielles.

2. **Décodeur**:utilise l'embedding comme entrée et génère une sortie de manière itérative à partir de celui-ci.

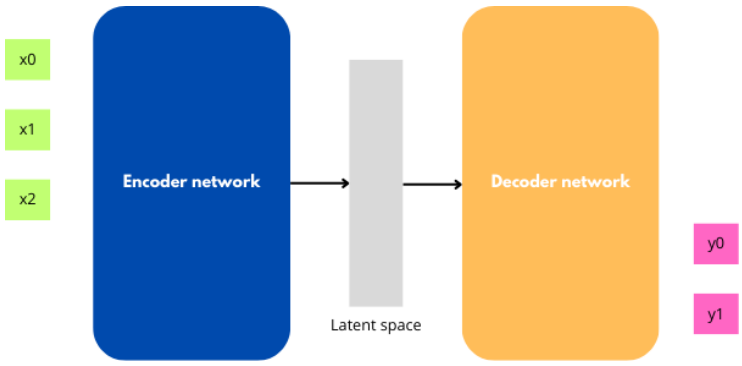


Figure 1: Une architecture encodeur-décodeur.

Par exemple, si nous avons une image de 28×28 pixels contenant un cercle, elle peut être compressée en un vecteur tridimensionnel représentant la position du centre et le rayon du cercle. C'est le rôle de l'encodeur.

Ensuite, en utilisant ces variables latentes, le décodeur peut déduire des informations, comme si l'aire du cercle dépasse un certain seuil.

L'Auto-Encodeur est essentiellement une architecture encodeur-décodeur, sauf que son objectif est de reconstruire l'entrée x de l'encodeur. Si on fournit x en entrée au modèle, on souhaite obtenir x en sortie du décodeur.

Dans l'exemple de l'image du cercle, le décodeur générerait un cercle avec le rayon et la position du centre appropriés.

Prenons un point aléatoire $z \in Z$ (Z étant l'espace latent) et passons le au décodeur pour tenter de générer une image. L'image produite sera probablement incohérente, car l'espace latent généré par un auto-encodeur est non structuré.

2.1.2 Variational Auto-Encoder (VAE)

L'Auto-Encodeur Variationnel (VAE) est une amélioration de l'auto-encodeur traditionnel obtenue grâce à l'Inférence Variationnelle Stochastique [6]. Il s'agit d'utiliser des méthodes d'optimisation stochastique afin de rendre l'espace latent adapté à de la génération de données. L'entraînement des VAE repose sur la contrainte de l'espace latent pour qu'il approxime une distribution fixé à l'aide de méthodes d'inférence variationnelle. Cela permet d'obtenir un espace latent plus continu et mieux organisé.

Nous modélisons le VAE [5] et les données de la manière suivante:

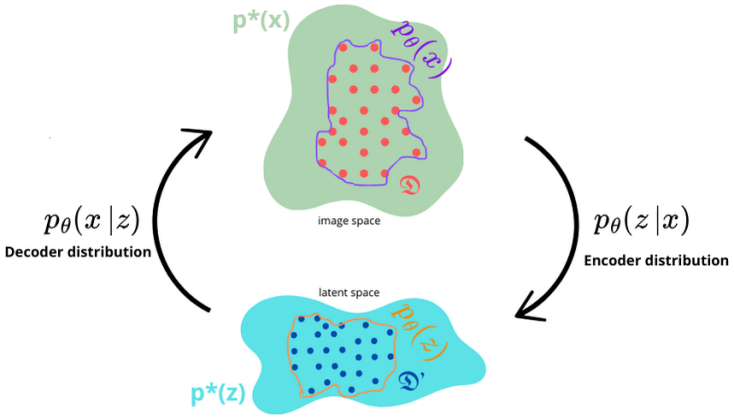


Figure 2: Modélisation du VAE avec les distributions postérieures et a priori.

1. L'ensemble de données d'entrée $\mathcal{D}$ et sa représentation correspondante dans l'espace latent, $\mathcal{D}'$.

2. $p_\theta(x)$:la distribution modélisée des images du jeu de données dans l'espace d'entrée, et $p_\theta(z)$:la distribution modélisée des variables latentes dans l'espace latent.

3. $p_\theta(z|x)$:la distribution reliant l'espace d'entrée à l'espace latent, et $p_\theta(x|z)$:la distribution reliant l'espace latent à l'espace de sortie.

En utilisant une notation similaire, nous désignons les distributions réelles par $p^*(x)$ et $p^*(z)$.

Notre objectif est d'approximer $p_\theta(z|x)$ par $q_\phi(z|x)$ à l'aide de l'inférence variationnelle, en cherchant à maximiser une fonction de perte appelée Evidence Lower Bound (ELBO).

$$\textbf{ELBO} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{\mathrm{KL}}(q_\phi(z|x)||p(z))$$

On a noté D_{KL} la divergence de Kullback-Leibler.

La nouvelle fonction de perte est décomposable en deux parties:

1. La perte de reconstruction $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$. En cherchant des distributions gaussiennes réduites, il est possible de montrer que la perte de reconstruction revient à la fonction L2.

2. La divergence de Kullback-Leibler $D_{\mathrm{KL}}(q_\phi(z|x)||p(z))$ qui elle gére la régularité de l'espace latent, son objectif est de rendre la distribution

$q_\phi(z|x)$ proche de $p(z)$, la distribution latente. En supposant cette dernière gaussienne centrée réduite, D_{KL} adopte une forme simple.

Le modèle VAE est donc un modèle capable de générer suite à un équilibre entre la qualité de reconstruction et la régularité de son espace latent.

Toutefois les VAE ont tendances à produire des images floues et donc peu adaptées à la résolution de problèmes réels. En essayant de générer des visages en entrainant un VAE sur le jeu de données Celeb-A [7].

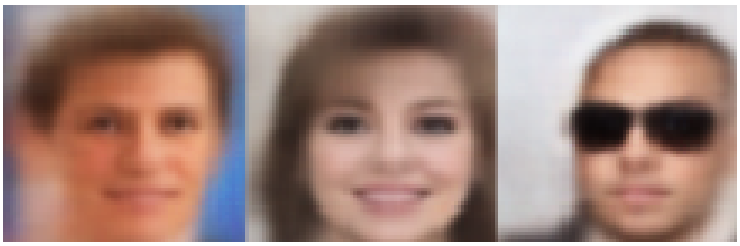


Figure 3: Generation d'image complexe avec le VAE

la fonction de perte ELBO est déséquilibrée entre la divergence de Kullback-Leibler (KL), qui tente d'ajuster l'espace latent à une distribution normale, et la perte de reconstruction, comme l'erreur quadratique moyenne (MSE).

En raison de la stochasticité de l'espace latent pendant l'entraînement, un échantillon peut produire une distribution de valeurs légèrement différentes de z. À cause de cette variation, la perte de reconstruction tente de moyenner ces fluctuations, ce qui se traduit mathématiquement par un effet de flou dans les reconstructions. Par conséquent, le modèle a tendance à sous-ajuster les détails plus fins.

2.1.3 β-VAE

Une amélioration des VAE consiste à introduire un hyperparamètre β dans la fonction de perte afin d'équilibrer la perte de reconstruction et la divergence de Kullback-Leibler (KL). Cette idée a été proposée par Higgins et al. (2017) [8]. Ils proposent une nouvelle formulation de la fonction de perte ELBO:

$$L = L_{\text{reconstruction}} - \beta \cdot D_{\mathrm{KL}}(q_\phi(z|x)||p_\theta(z))$$

où la perte de reconstruction peut être une perte de type L_2.

Si nous choisissons $\beta > 1$, la distribution $q_\phi(z|x)$ est fortement contrainte pour correspondre à la distribution a priori $p_\theta(z)$, qui est une gaussienne centrée

réduite. Ainsi, la perte de reconstruction a un impact réduit sur l'optimisation, ce qui permet d'éviter les images floues lors de la génération.

Cependant, l'inconvénient majeur est que cela réduit l'information mutuelle entre un point z dans l'espace latent et un point x dans l'espace d'entrée.

Toutefois, augmenter le β-VAE favorise le disentanglement (désentrelacement) de l'espace latent, qui est une propriété utile en génération.

L'idée d'un espace latent est de créer un espace de dimension réduite par rapport à l'espace des données. Il permet d'encoder l'information avec un faible nombre de paramètres.

Par exemple, une image d'un cercle peut être encodée en trois dimensions:la position du centre et le rayon. Ici, l'espace latent a deux dimensions pour la position et une pour le rayon. Maintenant, si nous voulons générer un cercle, il suffit de choisir une position et un rayon.

Un espace latent est désentrelacé lorsque chaque dimension représente une caractéristique spécifique et que ces caractéristiques sont indépendantes les unes des autres.

Cependant, les VAE standards ne produisent pas naturellement des espaces latents désentrelacés. L'article Understanding disentangling in β-VAE explique pourquoi un $\beta > 1$ favorise la factorisation de l'espace latent [9] en reliant la fonction de perte du β-VAE à la notion de capacité de canal en théorie de l'information. Afin de garder les bénéfices d'une bonne reconstruction mais également du disentanglement (désantrelacement), des méthodes d'entrainement existent et sont décrites dans le papier Burgess, Higgins et al. [9].

2.1.4 Ladder VAE

En s'inspirant des modèles profond et afin d'apprendre des représentations multi-échelle des architectures VAE profondes (hierarchiques) ont été proposée comme le Ladder VAE de Sønderby, Raiko et al. (2016) [10].

La principale difficulté pour stabiliser l'entrainement est de résoudre le problème de partage d'information. Sønderby, Raiko et al. (2016) [10] mettent en évidence la nécessité pour un étage VAE donné d'avoir une partage entre la distribution décodeur et posterieur.

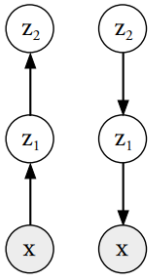


Figure 4: VAE profond sans partage d'information. (Fig.1-a du papier Ladder VAE)

Le Ladder VAE propose donc de partager les distributions pour des étages donnés.

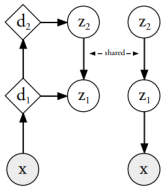


Figure 5: Ladder VAE. (Fig.1-b du papier Ladder VAE)

Chaque niveau i produit un espace latent Z_i. Ainsi Sønderby, Raiko et al. [10] $q\phi(z|x)$ et $p_\theta(x|z)$ définissent les distributions globales du modèles, elles même composées à partir des distributions auxiliaires par annexe. L'architecture proposée repose sur une paramétrisation précise des distributions:

$$q_\phi(\mathbf{z}|\mathbf{x}) = q_\phi(z_L|\mathbf{x}) \prod_{i=1}^{L-1} q_\phi(z_i|z_{i+1})$$

$$\sigma_{q,i} = \frac{1}{\hat{\sigma}_{q,i}^{-2} + \sigma_{p,i}^{-2}}$$

$$\mu_{q,i} = \frac{\hat{\mu}_{q,i}\hat{\sigma}_{q,i}^{-2} + \mu_{p,i}\sigma_{p,i}^{-2}}{\hat{\sigma}_{q,i}^{-2} + \sigma_{p,i}^{-2}}$$

$$q_\phi(z_i|\cdot) = \mathcal{N}(z_i|\mu_{q,i}, \sigma_{q,i}^2)$$

Ce partage d'informations entre les distributions est semblable au

fonctionnement du cerveau humain selon Sønderby, Raiko et al. [10], la perception est un aller-retour entre les signaux réels et ceux émulés par le cerveau.

Par ailleurs chaque étage est muni d'une couche de batch normalization et de warm-up. Ces étapes sont impactantes pour la régularité de l'espace latent produit. Voici un extrait des résultats sur l'impact des différents élements (extrait du papier Ladder VAE).

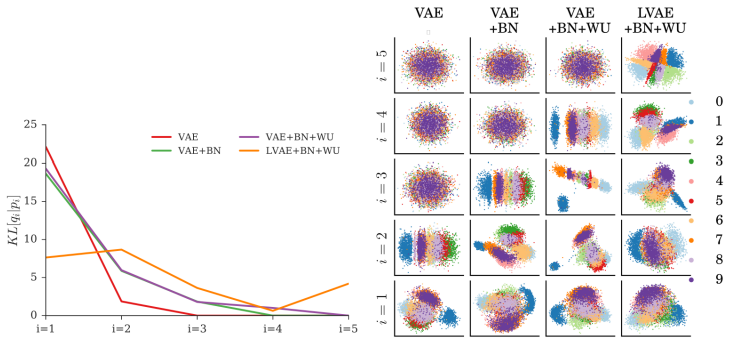


Figure 6: Entrainement de VAE et Ladder VAE sur MNIST. Graphiques PCA des échantillons de $q(z_i|z_{i-1})$ pour les modèles VAE à 5 couches et LVAE entraînés sur MNIST.

2.1.5 GAN, Normalizing flows, modèles autoregressifs

Si le Ladder VAE propose de bon résultats en génération, les jeu de données sur lequel il établit l'état-de-l'art sont des images simples comme MNIST. Les modèles type VAE bien qu'avantageux puisque donnant accès directement à la distribution latente sont souvent surpassés par d'autres familles de modèles profonds génératifs. En pratique, pour générer des images photoréalistes de haute résolution, d'autres architectures ont émergé, exploitant des approches différentes de celles des VAE. Parmi elles, trois grandes familles dominent aujourd'hui la génération d'images:

- Les Generative Adversarial Networks (GANs)
- Les Modèles Autorégressifs (PixelCNN, PixelSNAIL)
- Les Normalizing Flows

Generative Adversarial Networks (GANs) Les Generative Adversarial Networks (GANs), introduits par Goodfellow et al. [12], reposent sur une compétition entre deux réseaux neuronaux:

1. Un générateur $G(z)$, qui apprend à produire des échantillons réalistes à partir d'un bruit latent z.

2. Un discriminateur $D(x)$, qui apprend à distinguer les images réelles des images générées.

L'entraînement suit une formulation min-max où G essaie de tromper D, et D essaie de distinguer les vraies images des fausses:

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$$

Les GANs ont permis de générer des images photoréalistes avec des architectures comme StyleGAN [13] et BigGAN [14]. Toutefois, leur entraînement est instable et nécessite des techniques avancées pour éviter le mode collapse.

Modèles Autorégressifs Les modèles autorégressifs, comme PixelCNN [15] et PixelSNAIL [16], modélisent directement la distribution conjointe des pixels $p(x)$ comme un produit de distributions conditionnelles. Ces modèles génèrent chaque pixel séquentiellement, en tenant compte du contexte spatial. Bien qu'ils produisent des images de haute qualité, leur génération est lente, car chaque pixel doit être échantillonné individuellement.

$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2) \dots p(x_n|x_1, .., x_{n-1})$$

Normalizing Flows Les Normalizing Flows exploitent des transformations différentiables inversibles pour apprendre la distribution des données. Une transformation f est appliquée aux échantillons d'une distribution simple (ex:gaussienne) pour obtenir les données:

$$x = f(z), \quad z \sim \mathcal{N}(0, I)$$

L'apprentissage repose sur le changement de variable de la densité:

$$p(x) = p(z) \left| \det \frac{\partial f^{-1}(x)}{\partial x} \right|$$

Des architectures comme Glow [17] et RealNVP exploitent ces transformations pour générer des images de haute qualité, avec l'avantage de permettre une inférence exacte.

2.1.6 NVAE

Le Nouveau VAE est un modèle VAE hierarchique conçu dans un objectif de génération d'image. Au moment de sa parution, il achevait l'état-de-l'art en

génération parmi les modèles non-autoregressif sur les jeu de données MNIST [11] et Celeba-A [7].

Dans le NVAE, les variables latentes sont partitionnées en groupes disjoints:

$$z = \{z_1, z_2, \ldots, z_L\}$$

où L représente le nombre de groupes.

Le distribution latente est alors défini comme:

$$p(z) = \prod_l p(z_l|z_{<l})$$

et la distribution encodeur approximé par:

$$q(z|x) = \prod_l q(z_l|z_{<l}, x)$$

Nous pouvons alors écrire la forme hierarchique de la fonction de perte ELBO pour le NVAE:

$$\mathcal{L}_{\text{LVAE}}(x) := \mathbb{E}_{q(z|x)}\left[\log p(x|z)\right]$$

$$-D_{\text{KL}}(q(z_1|x)||p(z_1)) - \sum_{l=2}^{L} \mathbb{E}_{q(z_{<l}|x)}\left[D_{\text{KL}}(q(z_l|x, z_{<l})||p(z_l|z_{<l}))\right]$$

L'architecture utilise plusieurs méthodes pour stabiliser l'entrainement et les réponses du modèles.

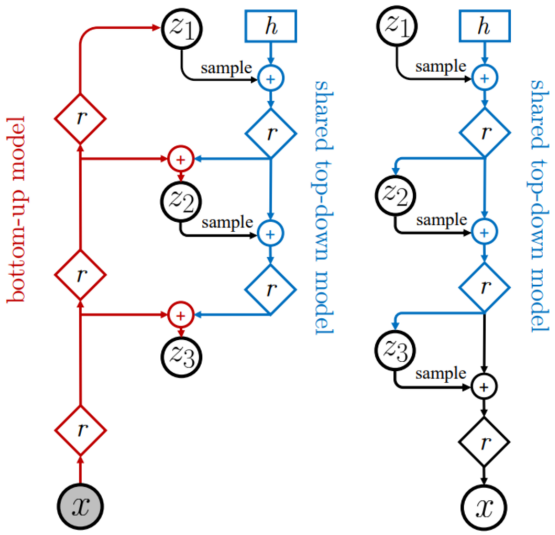


Figure 7: Architecture NVAE du papier original.

Des cellules résiduelles utilisant de la batch normalization (BN). Vahdat, A., Kautz, J. [4] proposent de coupler le BN avec la fonction d'activation Swish $f(u) = \frac{u}{1+e^{-u}}$.

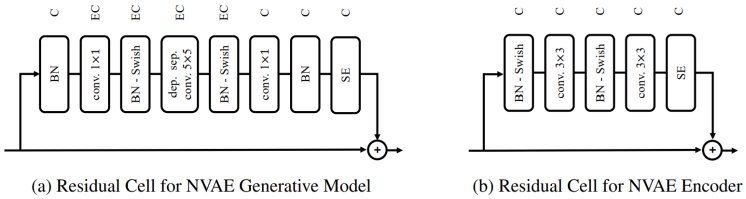


Figure 8: Architecture des cellules résiduelles dans le NVAE. Extrait du papier NVAE original.

Une partie du défi de l'architecture NVAE est la somme des divergences de Kullback-Leibler. Un entrainement de NVAE classique reposerait sur un décodeur minimisant la reconstruction et un encodeur générant des distributions latentes minisant KL.

Toutefois au vu du nombre de variables latentes, une grande différence entre la distribution encodeur et la distribution latente entrainerait un gradient élevé emenant la perte vers des zones instables. Une solution est de partager les

paramètres entre les distributions latentes et les distributions encodeurs. Lorsque la distribution latente (supposée gaussienne) évolue, on enregistre le delta entre ces paramètres. On utilise ces delta pour reparamétriser la distribution encodeur. Soit $p(z_l^i|z_{<l})$ la distribution latente de la i-ième variable à l'échelle l.

Nous avons:

$$p(z_l^i|z_{<l}) \sim \mathcal{N}(\mu_i(z_{<l}), \sigma_i(z_{<l}))$$

où $\mu_i(z_{<l})$ et $\sigma_i(z_{<l})$ sont les paramètres de la distribution latente.

Nous définissons ensuite $\Delta\mu_i$ et $\Delta\sigma_i$ comme des paramètres relatifs, qui mesurent la différence entre les paramètres de la distribution latente et ceux de l'encodeur.

Ainsi, la distribution de l'encodeur devient:

$$q(z_l^i|z_{<l}, x) = \mathcal{N}(\mu_i(z_{<l}) + \Delta\mu_i(z_{<l}, x), \sigma_i(z_{<l}) \cdot \Delta\sigma_i(z_{<l}, x))$$

où:

- $\mu_i(z_{<l})$ et $\sigma_i(z_{<l})$ sont les paramètres de la distribution latente.
- $\Delta\mu_i(z_{<l}, x)$ et $\Delta\sigma_i(z_{<l}, x)$ sont les ajustements relatifs de l'encodeur par rapport à la distribution latente.

La divergence Kullback-Leibler prend alors la forme:

$$D_{\mathrm{KL}}(q||p) = \frac{1}{2}\left(\frac{\Delta\mu_i^2}{\sigma_i^2} + \Delta\sigma_i^2 - \log\Delta\sigma_i^2 - 1\right)$$

Dans l'objectif de donner plus d'importance à la régularité de certains étages, il est également possible, comme pour le β-VAE, de mettre des coefficients devant les divergences. Les différents β_i deviennent alors des hyperparamètres essentiels.

Afin d'assurer que la divergence n'explose pas malgré tout il est parfois nécessaire d'ajouter une régularisation spectrale. Il s'agit de déterminer la plus grande valeur singulière de chaque couche. $L_{\mathrm{SR}} = \lambda \sum_i s(i)$ à $\mathcal{L}_{\mathrm{LVAE}}$, où $s(i)$ est la plus grande valeur singulière de la i-ième couche convolutionnelle. λ contrôle le niveau de lissage imposé par L_{SR}.

2.2 Modèles de segmentation classiques

Afin de générer des données issues de segmentation, nous utiliserons des modèles de segmentation dérivés de U-Net. Ces modèles ont démontré des performances solides en segmentation d'images biomédicales, notamment pour la segmentation des IRM cardiaques.

2.2.1 U-Net

Le U-Net, introduit par Ronneberger et al. [1], est une architecture basée sur un réseau de neurones convolutifs (CNN), conçue spécifiquement pour la segmentation d'images. Il repose sur une structure en U, caractérisée par:

- Un encodeur:capture le contexte de l'image via des convolutions et du max-pooling.
- Un décodeur:permet de récupérer les détails spatiaux pour une segmentation précise.
- Des connexions skip:relient directement les couches d'encodage aux couches correspondantes du décodage, assurant une meilleure préservation des détails.

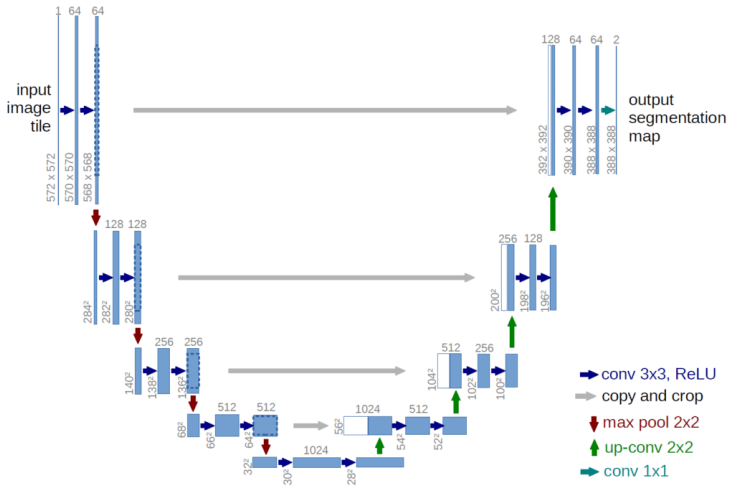


Figure 9: Architecture du U-Net [1].

Toutefois, U-Net reste limité face aux structures anatomiques complexes et aux artefacts présents dans les images médicales. Des variantes ont été développées pour améliorer ses performances.

2.2.2 Attention U-Net

L'Attention U-Net [2] est une extension de U-Net intégrant des mécanismes d'attention afin d'améliorer la segmentation. L'idée principale est de filtrer les features inutiles et de concentrer l'attention du réseau sur les régions pertinentes.

- Ajout de modules d'attention:ces modules pondèrent l'importance des

pixels en fonction du contexte global de l'image.

- Amélioration des connexions skip:les connexions entre l'encodeur et le décodeur sont modulées par l'attention, renforçant ainsi la précision des prédictions.

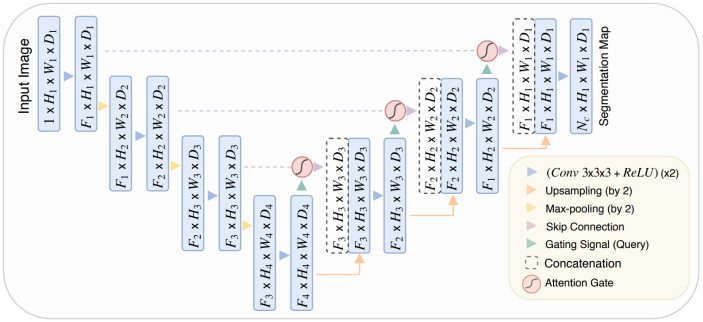


Figure 10: Architecture de AttentionUNet [2].

2.2.3 Swin UNETR

Le Swin UNETR [3] représente une avancée majeure dans la segmentation médicale, intégrant les Transformers dans une architecture inspirée de U-Net. Ce modèle repose sur:

- L'utilisation des Swin Transformers:au lieu d'utiliser des convolutions classiques, le Swin UNETR applique un mécanisme d'attention à fenêtres glissantes (shifted windows), permettant une modélisation plus efficace des dépendances inter-patch à l'aide de supperposition.

- Une architecture hybride:combine les avantages des Transformers pour capturer les relations à longue portée et des convolutions pour une meilleure exploitation des détails locaux.

- Une adaptation aux images médicales 3D: contrairement aux modèles classiques, Swin UNETR est conçu pour traiter des volumes 3D, ce qui le rend particulièrement efficace pour les IRM 4D cardiaques.

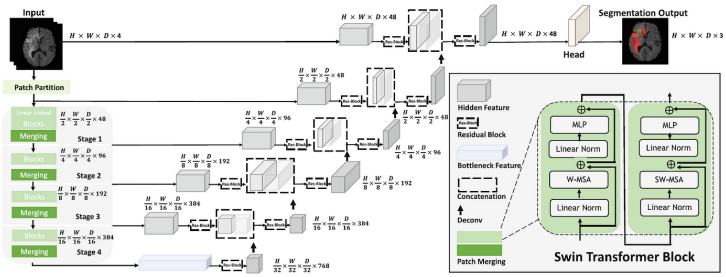


Figure 11: Architecture du Swin UNETR [3].

3 Exploration et travail sur les données

3.1 Jeu de données ACDC

Le jeu de données *Automated Cardiac Diagnosis Challenge* (ACDC) [18] est une base de données d'IRM cardiaques dédiée à l'analyse automatique des structures du cœur. Il contient des volumes 4D (3D + temps) acquis sur **151 patients** et annotés manuellement par des experts. Chaque volume est constitué de plusieurs coupes en *short-axis* couvrant le ventricule gauche (**LV**), le ventricule droit (**RV**) et le myocarde (**MYO**). Deux instants clés du cycle cardiaque sont retenus:

- **End-Diastole (ED)**: moment de remplissage maximal du ventricule gauche.
- **End-Systole (ES)**: moment d'éjection maximale du sang.

Chaque patient appartient à l'une des cinq catégories suivantes:

- **NOR**: sujets sains sans pathologie cardiaque.
- **MINF**: patients ayant subi un infarctus du myocarde avec fraction d'éjection altérée.
- **DCM**: cardiomyopathie dilatée.
- **HCM**: cardiomyopathie hypertrophique.
- **ARV**: anomalies du ventricule droit.

L'ensemble de données est divisé en **100 patients pour l'entraînement** et **50 pour les tests**, avec des annotations en format `nifti`. Chaque annotation segmente les trois structures principales: **LV, MYO et RV**, avec des valeurs respectives de `3, 2 et 1` dans le masque.

Un patient est défini par un fichier d'information contenant une pathologie éventuelle et notamment 2 volumes 3D *frame*, image arrêtée du volume 4D au moment de la fin de diastole et fin de systole. Chaque frame est un ensemble de tranches qui elles sont des images 2D.

3.2 Préparation des données

Les volumes 4D initiaux sont prétraités avant utilisation dans notre modèle. Les principales étapes du prétraitement sont:

Sélection des phases ED et ES Chaque volume étant une séquence temporelle couvrant un cycle cardiaque complet, nous extrayons uniquement les instants ED et ES pour réduire la redondance et nous focaliser sur les moments physiologiquement significatifs.

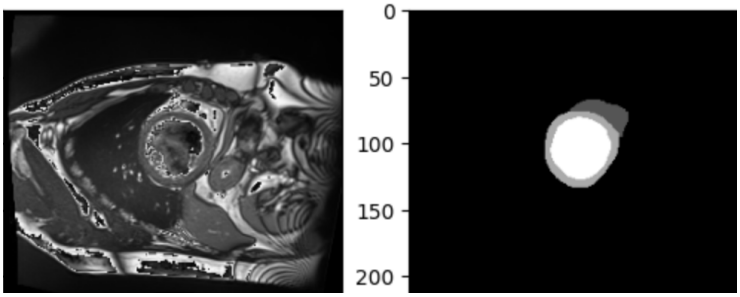


Figure 12: Coupe d'un volume 3D présents dans le jeu de données ACDC. A droite le masque annoté.

Conversion en coupes 2D Les volumes 3D sont ensuite découpés en coupes axiales 2D afin de s'adapter aux architectures CNN standard. Chaque volume étant constitué de 10 à 20 coupes en fonction du patient, cela génère un total de plusieurs milliers d'images annotées.

Redimensionnement et normalisation Les images originales ont une résolution variable (256×256 ou 192×192). Afin d'uniformiser les entrées du modèle, nous les redimensionnons à une résolution standard de 128×128 en appliquant une transformation `torchio.Resize` tout en préservant les proportions de l'organe segmenté. Une normalisation en intensité est également effectuée entre les percentiles 1% et 99% afin de réduire l'impact du bruit et des variations d'acquisition.

Alignement et correction des volumes Pour assurer la cohérence anatomique, un recentrage automatique sur la région du cœur est appliqué via une extraction de boîte englobante autour des masques de segmentation. Cette étape limite l'impact des différences de cadrage entre les patients.

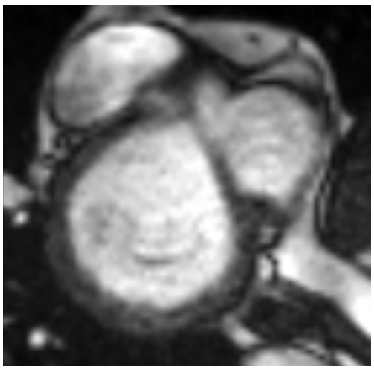


Figure 13: La coupe est centrée autour de la région d'interet.

Encodage des labels Les masques de segmentation sont convertis en une représentation **one-hot** avec 4 canaux:

- Canal 0: arrière-plan
- Canal 1: RV
- Canal 2: MYO
- Canal 3: LV

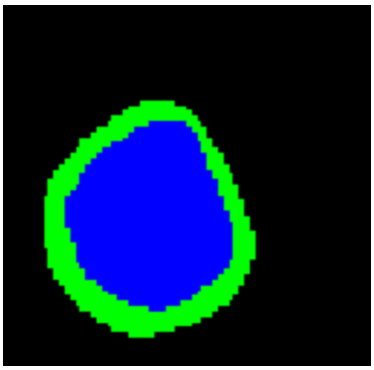


Figure 14: La vérité terrain est converti en format 4 canal. A l'affichage en vert le myocarde et le ventricule gauche en bleu.

Données finales Après ces étapes, le jeu de données prétraité contient des images 2D normalisées de taille 128×128, avec des annotations multi-classe et une séparation entre phases ED et ES. Ces données sont ensuite utilisées pour

l'entraînement et l'évaluation de notre modèle de segmentation. Le NVAE quand à lui ne sera entrainé que sur les masques de segmentations.

4 Précédents travaux

4.1 Nathan Painchaud

La segmentation automatique des structures cardiaques à partir d'IRM est un défi majeur en raison des contraintes anatomiques et physiologiques propres au cœur. Une segmentation erronée peut mener à des incohérences anatomiques, ce qui limite l'applicabilité clinique des modèles de deep learning. Dans ce contexte, **Painchaud et al.** ont proposé une approche intégrant des contraintes anatomiques fortes au sein d'un modèle de segmentation basé sur un **Variational Autoencoder (VAE)**, garantissant ainsi la plausibilité des formes cardiaques reconstruites.

L'idée centrale de leur approche repose sur l'utilisation d'un **Constrained Variational Autoencoder (C-VAE)**, qui apprend un espace latent contraint aux formes cardiaques valides. Cela permet de projeter toute segmentation générée dans cet espace latent et de contraindre la sortie à respecter des critères anatomiques stricts.

4.1.1 Contributions principales

1. **Apprentissage d'un espace latent valide**: Le **C-VAE** est entraîné sur des masques annotés par des experts, garantissant ainsi que les échantillons issus de l'espace latent correspondent à des formes anatomiquement plausibles.

2. **Correction des formes invalides**: Une fois la segmentation générée, un post-traitement basé sur un **VAE de correction anatomique** permet de projeter une segmentation incorrecte dans l'espace latent et de récupérer la forme anatomiquement valide la plus proche.

3. **Méthode d'augmentation des données par rejection sampling**: Le modèle est entraîné avec un mécanisme d'**anatomically-constrained rejection sampling**, qui permet de générer de nouveaux vecteurs latents valides, assurant ainsi une meilleure couverture de l'espace latent et limitant les artefacts.

4. **Interpolation linéaire dans l'espace latent**: Le modèle apprend une fonction de régularisation linéaire via un MLP noté y_θ, qui favorise l'organisation linéaire du manifold latent et améliore la stabilité des interpolations.

5. **Optimisation du problème de projection**: La recherche de la forme

valide la plus proche repose sur une minimisation de la distance L_2.

4.1.2 Résultats et implications

Les expérimentations menées par Painchaud *et al.* montrent que cette approche permet une segmentation robuste et anatomiquement plausible, avec une réduction significative des erreurs de structure.

4.2 Reprise des résultats

Cette méthode repose sur le principle de sampling de l'espace latent. Dans le cas du NVAE l'espace latent est hierarchique et donc difficile de savoir quel niveau on choisit de considérer. Nous faisons donc le choix d'essayer d'entrainer un NVAE à reconstruire les masques de segmentations seulement à partir de son encodeur et de son décodeur en le contraignant sur les masques du jeu de données ACDC.

5 Expériences

Dans l'objectif d'étudier l'apport du NVAE sur les pipelines de segmentation d'image médicale, ce projet décide de raisonner de la manière suivante:

1. Entrainement d'un NVAE sur des masques correctes afin de trouver un groupe d'hyperparamètre de référence.

2. Sélection et entrainement de modèles de segmentations pour constituer une base de données de masque à améliorer.

3. Intégration du NVAE dans une pipeline modèle de segmentation couplé à un NVAE.

La pipeline finale aura une allure comme décrite ci-contre.

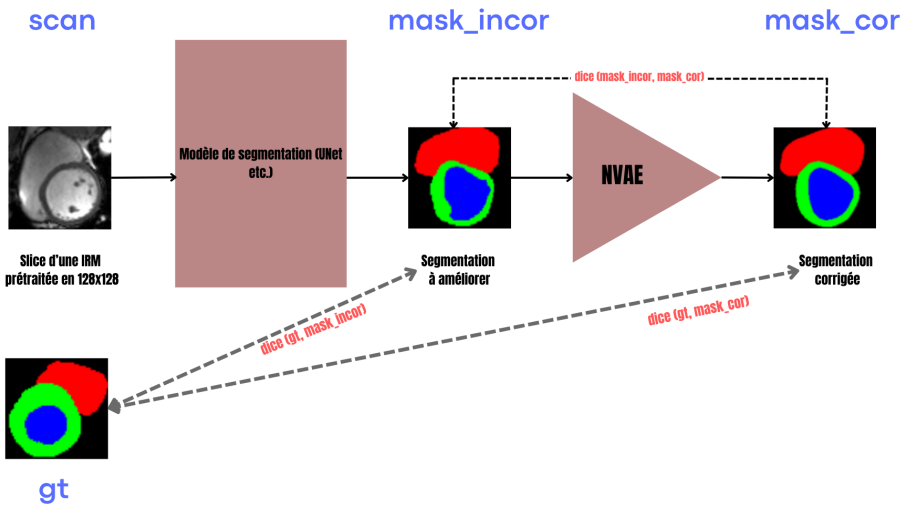


Figure 15: Vue gloable de notre objectif

Toutes les experiences sont réalisées sous PyTorch Lightning avec CUDA sur des GPU Nvidia P100 16GB.

5.1 Premier entrainement du NVAE sur ACDC

Pour ce premier entrainement on utilisera uniquement les masques de segmentations du ACDC. Il s'agit de tenseur de format $[B, 4, 128, 128]$ où B désigne la taille de batch. L'objectif est que le NVAE arrivent à reconstruire ces masques de segmentation. Nous nous interesseront à sa capacité à reconstruire chaque classe et à reconstruire le masque au global. Nous cherchons également à avoir des images anatomiquement valides.

Pour cela on fait le choix d'un modèle NVAE avec une profondeur de 3 niveaux. On rappelle que les modèles NVAE hierarchiques peuvent amener à des valeurs de divergence KL qui explosent. Nous faisons le choix de pondérer l'importance des divergences dans la fonction de perte. Nous cherchons des valeurs optimales pour $\beta_1, \beta_2, \beta_3$. Pour voir en détail l'architecture NVAE utilisée voir Annexe B.

Hyperparamètre	Valeur
Learning rate initial	1×10^{-2}
Optimiseur	AdamX ($\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 1 \times 10^{-3}$)
Pénalité L2	3×10^{-4}
Scheduler	CosineAnnealingLR ($\eta_{\min} = 1 \times 10^{-4}$)
Warm-up	6420 itérations
Nombre maximal d'époques	100
Taille du batch	8

Table 1: Liste des hyperparamètres d'entraînement.

Pour le choix des hyperparamètres du modèles nous avons démarré avec les hyperparamètres par défaut de l'architecture:

Hyperparamètre	Valeur
Nombre de canaux d'entrée	4
Nombre de canaux après stem	64
Nombre minimal de canaux	0
Nombre de canaux latents	20
Groupes par couche	[4, 2, 1]
Facteur de réduction des canaux	8
Nombre maximal d'époques	50
Coefficients β par couche	[1.0, 1.0, 1.0]
Etapes de warm up KL	500
Régularisation spectrale (SR)	False

Table 2: Hyperparamètres par défaut du modèle.

On affiche la perte en entrainement 16 et en validation 17.

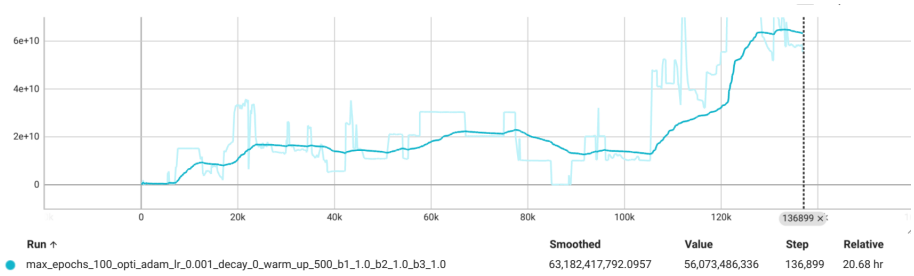


Figure 16: Fonction de perte en entrainement.
En abscisses le numéro de batch, en ordonée la valeur de la perte.

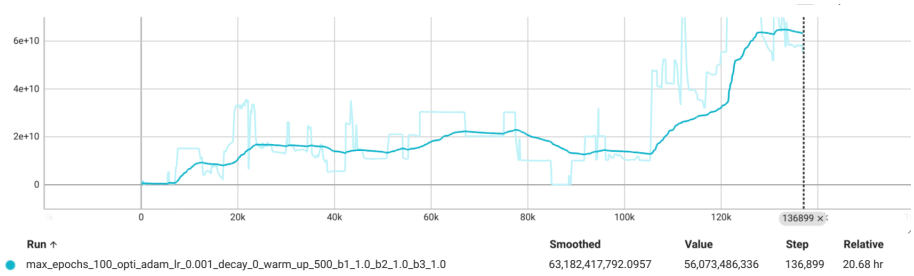
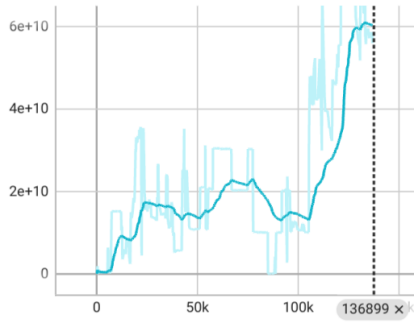
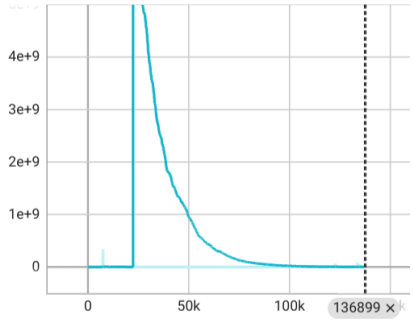


Figure 17: Fonction de perte en validation.
En abscisses le numéro de batch, en ordonée la valeur de la perte.

Le temps d'entrainement est anormalement élevé (plus de 20h). Le modèle ne semble par ailleurs pas apprendre avec des pertes croissantes. La partie de la fonction de perte responsable de la croissance est la divergence KL. La partie reconstruction de la fonction de perte décroit bien.



(a) divergence KL



(b) perte de reconstruction

Figure 18: Décomposition de la fonction de perte.
En abscisses le numéro de batch, en ordonée la valeur de la perte.

L'hyperperamètre *Nombre de canaux après stem* représente l'augmentation de canaux faites dès l'entrée de l'encodeur. Nous décidons alors de ne pas appliquer d'augmentation de canal et d'observer l'effet sur le temps d'entrainement. Par ailleurs afin de stabiliser la perte KL nous augmentons le nombre d'étape de warm-up. Une fois la divergence KL stabilisée nous rechercherons le meilleur compromis entre divergence KL et reconstruction par une recherche d'hyperparamètres β, valeurs qui pondèrent les divergence KL dans la perte globale.

Bien que le warm-up ait pour effet de faire travailler la divergence petit à petit, l'augmentation du warm-up ne résoud pas le problème d'entrainement. Les pertes ne sont toujours pas stabilisées. Le véritable changement se fait avec le passage du *Nombre de canaux après stem* de 64 à 4. Outre une réduction du temps d'entrainement attendu (passage de 20h à 6h), la divergence est stabilisée et les fonctions de pertes en validation et entrainement convergent.

Recherche des β Nous retenons les plages de valeurs proposée par M. Freddy Jiang dans son rapport. Il propose de faire une recherche de $\beta \in \{8, 9, 10\}^3$. Nous réalisons des essais d'entrainement avec les valeurs de warm-up et *Nombre de canaux après stem* retenues. Nous essayons les 27 combinaisons de β possible. Pour chaque combinaison nous entrainons notre modèle de manière à retenir la configuration avec la plus petite perte en validation ainsi que la configuration à la dernière epoch. On évaluera chacune des 27*2=54 configurations à l'aide du dice score et de la validité anatomique.

Beta 1	Beta 2	Beta 3	Dice Score	Anatomical Validity
11.0	11.0	10.0	0.9412	0.8992
11.0	11.0	11.0	0.9390	0.8783
11.0	11.0	9.0	0.9393	0.8691
11.0	8.0	11.0	0.9367	0.9136
11.0	8.0	9.0	0.9423	0.8940
8.0	**10.0**	**11.0**	**0.9491**	**0.9738**
8.0	10.0	9.0	0.9451	0.8154
9.0	10.0	11.0	0.9450	0.9241
9.0	10.0	9.0	0.9463	0.8887
9.0	8.0	11.0	0.9454	0.9424

Table 3: Extrait de la recherche des β optimaux.

Le β optimal est donc la combinaison $(8, 10, 11)$. Sous ces conditions le NVAE reconstruit les classes avec un dice score par classe supérieur à 0.92. Nous notons le dice score par classe (où LV tient pour le ventricule gauche, RV le droit et MYO pour le myocarde).

β	Dice LV	Dice MYO	Dice RV
(8, 10, 11)	0.9751	0.9298	0.9425

Table 4: Dice Scores pour différentes valeurs de Beta

Nous proposons donc le récapitulatif des hyperparamètres optimaux:

Hyperparamètre	Valeur
Nombre de canaux d'entrée	4
Nombre de canaux après stem	4
Nombre minimal de canaux	0
Nombre de canaux latents	20
Groupes par couche	[4, 2, 1]
Facteur de réduction des canaux	8
Nombre maximal d'époques	50
Coefficients β par couche	[8.0, 10.0, 11.0]
Etapes de warm up KL	6420
Régularisation spectrale (SR)	False

Table 5: Hyperparamètres séléctionnés du modèle.

Voici quelques exemples de masque donné au NVAE choisi et voici le résultat en sortie du modèle.

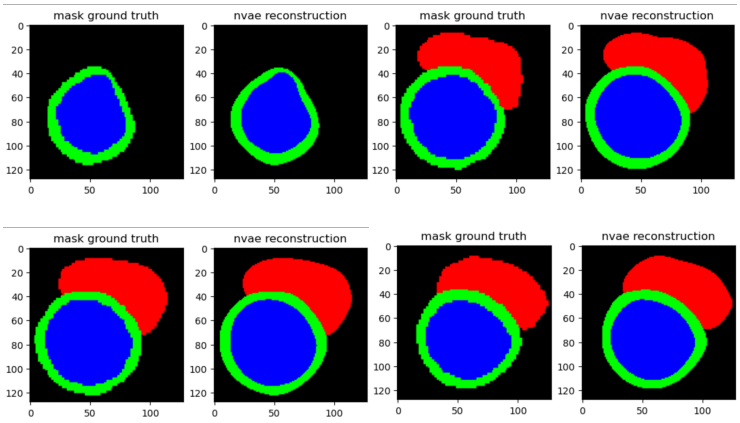


Figure 19: Exemples de reconstructions générées par NVAE

5.2 Génération des segmentions incorrectes

Pour compléter la pipeline 15 nous faisons le choix de 4 modèles de segmentation. Ces modèles prendront en entrée les scans 2D de taille 128×128 issus du jeu de données ACDC. Nous faisons le choix de prendre des modèles

de segmentations populaires en imagerie médicale. A la suite d'une recherche bibliographique nous faisons le choix de prendre un UNet [1], ResUNet (un UNet avec des connexions residuelles à la place de convolutions simples), Attention UNet [2] et Swin UNetr [3]. Les implémentations utilisées sont inspirées des implémentations de la bibliothèque MONAI [19]. Chacun des scans 2D segmentés (sous forme de masque à 4 canaux) et stockés dans un tenseur. A l'issue de cette expérience tout les scans 2D de ACDC ont été ségmentés chacun avec les 4 modèles. On obtient donc un ensemble de segmentation de 4 fois la taille de ACDC. On répartit ces segmentations en 3 jeu de données:un pour l'entrainement de 80% la taille totale puis 10% pour la validation et 10% pour le test.

Hyperparamètre	**Valeur**
Learning rate	1×10^{-3}
Optimizer	Adam
Max epochs	100

Table 6: Hyperparamètres commun aux modèles de segmentations

Pour les modèles à attention Swin UNetr et AttentionUNet nous avons également essayé des configurations avec et sans warm-up. Toutefois le warm-up n'apportait pas de point sur le dice score global. Pour le ResUNet nous faisons le choix d'utilisation de 2 blocs résiduels par branche.

A l'issue de l'entrainement nous obtenons des dice scores en test pour chaque modèle comme suit:

Architecture	**Best Dice Score**
Attention UNet	0.9162
Swin UNet	0.8952
UNet	0.8866
UNet	0.9037

Table 7: Meilleurs Dice Scores pour différentes architectures

Nous observons la distribution générales des dice scores sur l'ensemble des images segmentées. A l'aide d'une représentation en violon, nous observons un lobe principal autour de 0.9 points de dice scores. Un lobe secondaire autour de 0.6, l'objectif de la pipeline serait de ramener ce lobe secondaire vers ce lobe principal. Enfin des lobes tertiaires vers 0.3 et 0.0 sont à notés et sont éventuellement améliorables bien qu'il semble compliquer de corriger une segmentation si incorrectes.

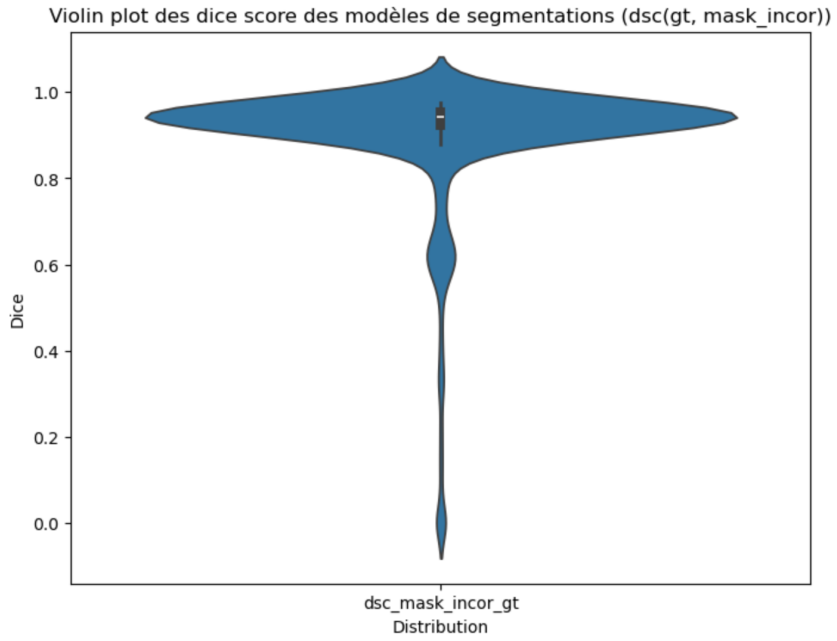
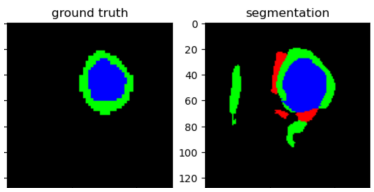
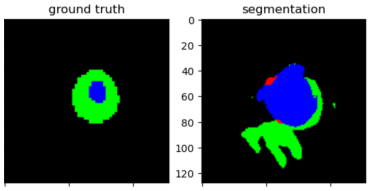


Figure 20: Distribution des dice scores pris entre le masques annotés (gt, ground truth) et les masques issues de la segmentation par les modèles dit 'classiques'.

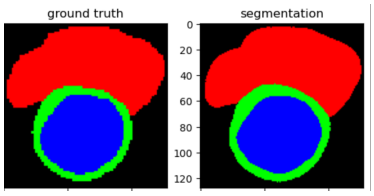
Voici quelques exemples de segmentations produites:



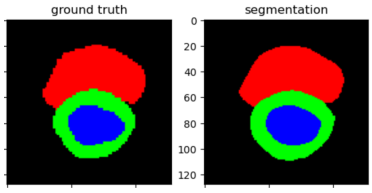
(a) Issue du lobe secondaire



(b) Issue du lobbe tertiaire

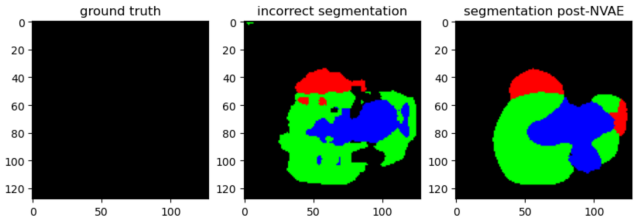


(c) Issue du lobbe principal



(d) Issue du lobbe principal

Figure 21: Segmentation issues des modèles de segmentations



5.3 Pipeline Segmentation + NVAE

Le jeu de données résultant de la précédente expérience est appelé **ACDCSeg**. C'est ce jeu de données qui sera fourni au NVAE afin d'en améliorer le dice score. En utilisant le NVAE entrainé précédemment nous souhaitons améliorer le dice score et l'anatomic validity entre les masques annotés (**gt**) et les masques de ACDCSeg (**mask_incor**). Cette métrique nous permettra d'évaluer l'impact du NVAE sur l'amélioration ou non des segmentations déjà éffectuées par les modèles classiques. Par conséquence nous porterons une attention particulière au cas suivant:

1. Observation des exemples "mal segmentés" par les modèles de segmentation (i.e avec un dice score entre **gt** (ground truth) et **mask_incor** bas au regard des autres valeurs.) On portera une attention sur l'allure de ses segmentations une fois passée dans le NVAE.

2. Observation des exemples fortement modifiés par le NVAE (i.e avec un dice score entre **gt** (mask_cor) et **mask_incor** élevé au regard des autres valeurs.) Nous évaluerons l'impact du NVAE sur ces cas, le NVAE a-t-il été capable d'augmenter la qualité de la segmentation ou au contraire a t-il diminuer cette dernière?

3. Observation des exemples les plus améliorés et détriorés par le NVAE. Nous regarderons les exemples avec une grande différence entre $\text{dsc}(gt, mask\_incor)$ et $\text{dsc}(gt, mask\_cor)$. Cette évaluation pourrait qualifier si les exemples les plus touchés sont altérés ou améliorés mais également si il s'agissait d'exemple initalement bien segmentés ou non.

Observation des exemples "mal segmentés" Sur les exemples mal segmentés dès le modèle de segmentation le NVAE ne peut que difficilement reprendre la segmentation. Toutefois il lisse les formes et les rends plus cohérentes.

Le dice score n'augmente pas puisqu'il est à quasi 0 des deux côtés mais l'effet du NVAE est bien visible.

Au global nous observons observer que le lobbe secondaire à 0.6 s'est amaigri mais que certains lobbes tertiaires sont légérement plus épais. En

moyenne et sur le lobbe principal il y a peu de changement. Une étude plus appronfondie sur cette dynamique statistique est écrite plus bas.

Figure 22: Figure en violon des distributions dice scores avant et après NVAE.

5.4 Pipeline avec contexte

Dans un second temps nous regardons l'influence du contexte sur la qualité de la reconstruction par le NVAE. Nous faisons le choix de ne plus seulement donner au NVAE une image 2D mais de lui donner 3 images 2D. Pour chaque slice d'IRM ségementée nous enregistrons la slice segmentée précédente et la slice segmentée suivante à la slice segmentée considérée. Nous concacétenons alors ces 3 masques pour n'en former plus qu'un qui sera donné au NVAE en tant que tenseur à non plus 4 canals mais 12 canals. Il est alors possible de donner du contexte au NVAE afin d'évaluer l'influence du contexte sur la recontstruction.

Cette méthode s'inspire directement des méthodes n-gram en traitement automatique du langage. Donner du contexte au modèle peut aider à inférer la suite de la phrase ou à la reconstruire.

Nous avons donc reconstruit un jeu de données en revenant à l'étape de séparation des volumes 3D en volumes 2D. Nous avons porté une attention particulière à ne créer des 3-grams connexes. Dans un même triplet, il ne faut que des slices du même patient. En s'inspirant du traitement du langage qui utilise un token *EOS* (end of sequence) en début et fin de phrase, nous avons initalisé un masque *EOS* ne contenant que la classe *background* et ayant pour vérité terrain un masque ne contenant que cette même classe. Nous avons intercalé cette image en début et fin d'IRM pour chaque patient. Nous avons

ensuite adapté l'architecture NVAE à l'acceuil de tenseurs à 12 canaux.

En résumé dans cette nouvelle architecture le NVAE acceuil en entrée 12 canaux et en ressort 4 canaux. Les vérités terrains elles sont toujours à 4 canaux. Ceci ayant pour conséquence qu'il n'est pas nécessaire de changer la fonction de perte.

Toutefois la question de changer le *nombre de canaux après stem* de 12 à 4 s'est posée. La configuration à 12 canaux a comme précédemment fait explosé les valeurs de la divergence KL.

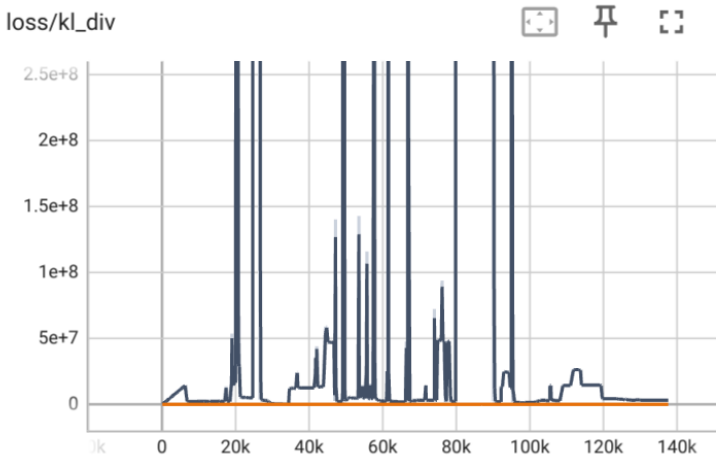


Figure 23: Le nombre de canaux après stem à 12 ne permet pas de stabiliser la divergence.
En bleu la divergence avec 12 canaux et en orange avec 4 canaux. En abscisses le nombre de batch analysés et en ordonnées la valeur de la perte.

Nous reprenons les hyperparamètres des précédents entrainements de NVAE. Nous réalisons une autre recherche pour trouver les β adaptés.

En sélectionnant le modèle avec la plus grande différence de dice score $\mathbf{dsc}(gt, mask\_cor) - \mathbf{dsc}(gt, mask\_incor)$ on fait finalement le choix de $\beta = (1, 1, 5)$.

6 Evaluations

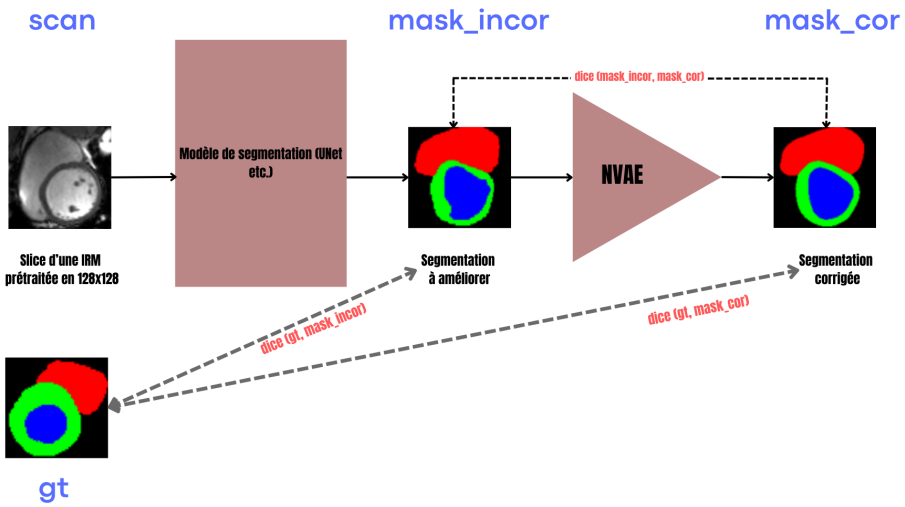


Figure 24: Pipeline et mesure des dice scores

Nous avons mener une étude statistique sur la répartition des dices scores avant et après utilisation du NVAE. Nous utilisons le dice score puis des estimateurs statistiques afin d'identifier les force et les faiblesses de cette pipeline sur la pipeline classique. Un travail similaire a été fait afin d'estimer l'apport du contexte.

6.1 Dice Score

Nous rappelons que le dice score est une mesure de supperposition de deux ensembles (ici ensemble de points). Une haute valeur (proche de 1) signifie une forte ressemblance et une faible valeur (proche de 0) signifie une faible ressemblance.

$$DSC = \frac{2|A \cap B|}{|A| + |B|}$$

où A désigne un canal considéré, B le même canal sur la vérité terrain.

6.2 Evaluation de la pipeline

Nous regardons dans un premier temps la distribution des dice scores avant et après NVAE (en comparaison à la vérité terrain).

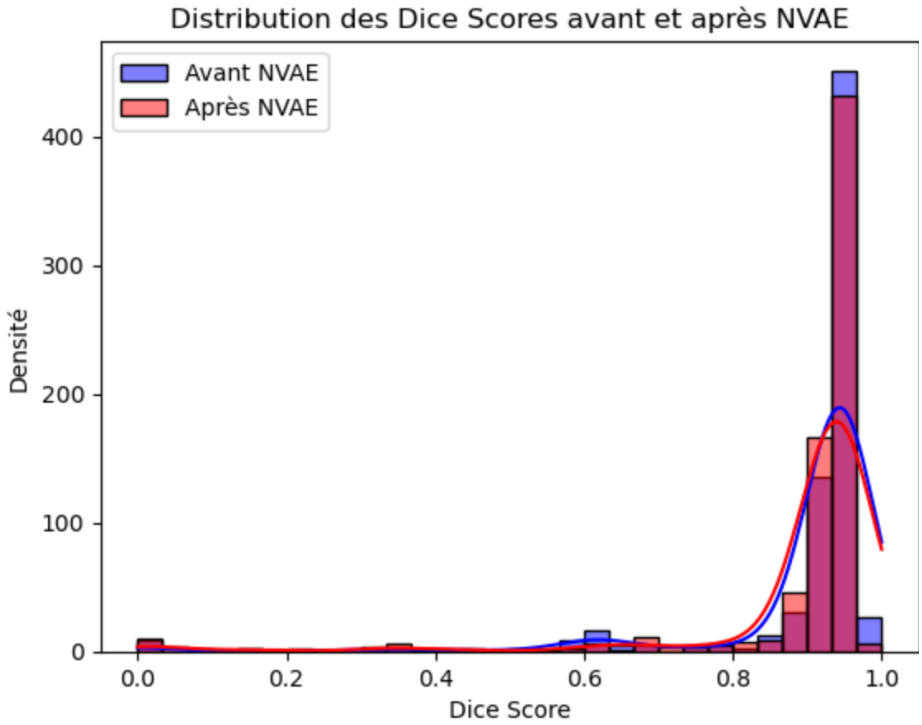


Figure 25: Effet du NVAE sur la distribution

Nous pouvons observer que le NVAE a tendance à ramener des des exemples dans la plage 0.9–0.93. Avant le NVAE nous en comptons 136 puis après le NVAE nous en comptons 166. Toutefois le nombre de dice score au-dessus de 0.93 a lui diminué avec le NVAE.

Nous cherchons donc à verifier la provenance de l'augementation de la zone 0.9–0.93, est-elle due a une dégradation des exemples ou a une amélioration des représentations.

Nous montrons la distribution différence entre les dice scores:
$\mathbf{dsc}(gt, mask\_cor) - \mathbf{dsc}(gt, mask\_incor)$.

Les valeurs positives sont des améliorations et les valeurs négatives des déteriorations. Nous montrons ensuite un histogramme afin de voir la répartition des valeurs. Une très grande majorité est très peu affectée par le NVAE. En excluant le pic central de 610 exemples du jeu de données test (sur les 728 exemples), nous comptons 75 exemples améliorés par le NVAE. Sur les 75 exemples améliorés, 18 le sont de plus de 0.3 points de dice score.

Cependant 42 exemples sont eux déteriorés. Toutefois sur les 42 exemples

déteriorés, 23 le sont de moins de 0.02 points de dice score.

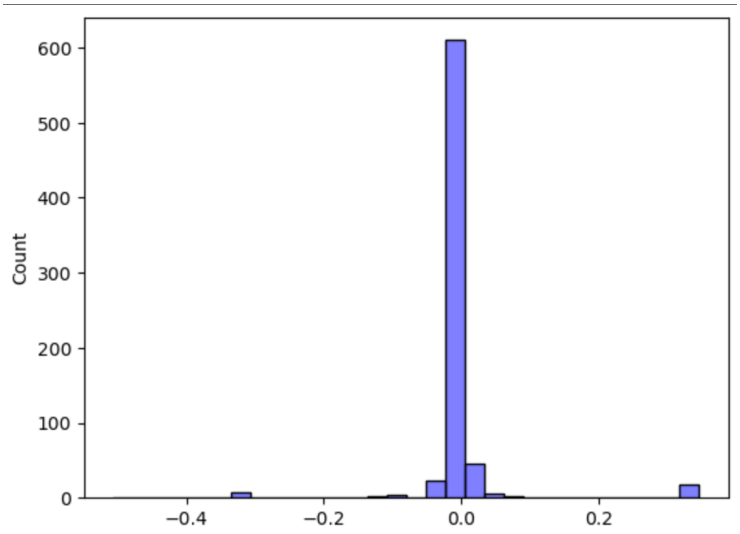


Figure 26: Différence de dice score après NVAE—avant NVAE

Nous visualisons le dice score après NVAE en fonction de celui avant en espérant voir si les dice scores bas avant sont à présent élevé ou si il s'agit d'une autre dynamique.

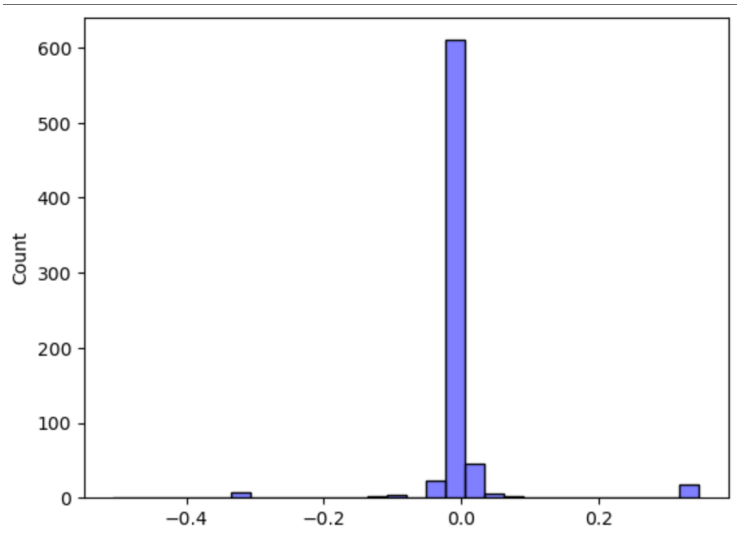


Figure 27: Différence de dice score après NVAE—avant NVAE

Après une régression linéaire on observe que le coefficient R^2 est à 0.775. Ce nombre explique qu'il n'y a pas d'inversion rare des dice scores, les exemples bas ne deviennent pas souvent élevé. Toutefois on assiste tout de même à un effet du NVAE. Par exemple un cluste secondaire négatif qui était initialement autour de 0.6 points de dice score est à présent au-dessus des 0.8.

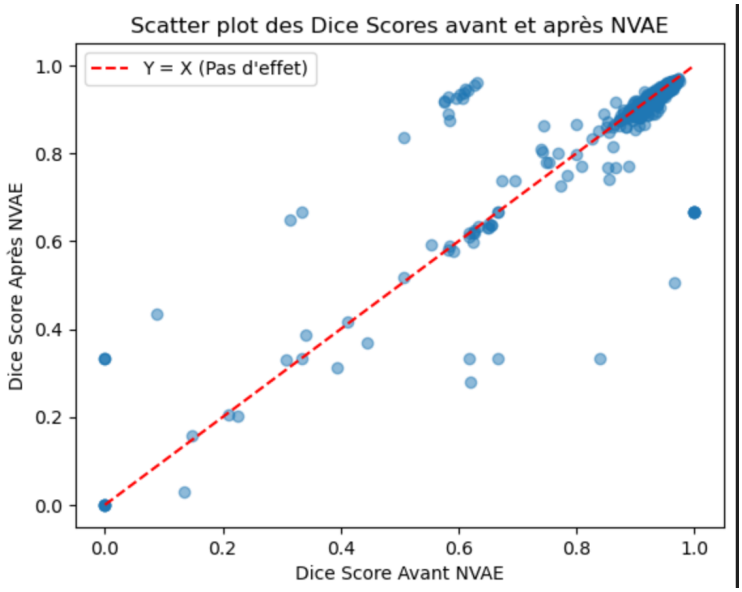


Figure 28: Différence de dice score après NVAE—avant NVAE

Parmis les plus mauvaises segmentations Lorsque l'on regarde les plus mauvaises segmentations nous pouvons voir que certaines sont très fortement améliorées d'un point de vue du dice score. En se penchant sur ce phénomène nous nous rendons compte que les exemples à faible dice score ($< 10^{-3}$) ont parfois un dice score améliorés car il s'agit en réalité de slice remplie uniquement de fond. Le NVAE ne change en réalité pas la validité de la segmentation mais le dice score lui augmente.

On en conclut que la pipeline présentée ne peut être adaptée lorsque le dice score avant NVAE est trop faible (proche de 0).

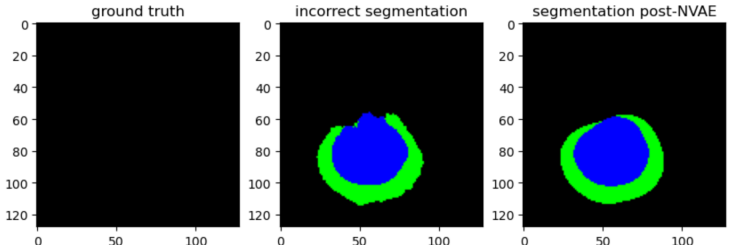


Figure 29: Le NVAE a une capacité à rendre les structures cohérentes même si il ne touche pas toujours le dice score.

Qu'en est-il pour les segmentations modérées? Nous notons que ce phénomène de lissage mais sans changer en profondeur la structure de la segmentation se présente également sur les exemples de dice score aux alentours de 0.6.

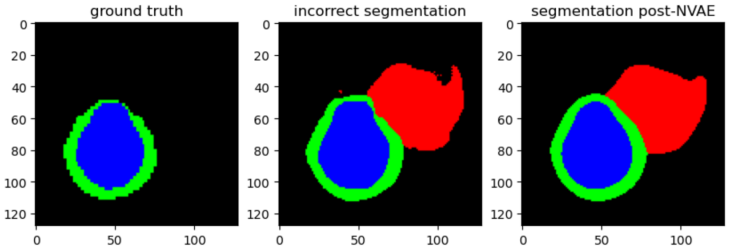


Figure 30: Le NVAE a une capacité à rendre les structures cohérentes même si il ne touche pas toujours le dice score.

Nous illustrons sa capacité à supprimer des structures anatomiques incorrectes:

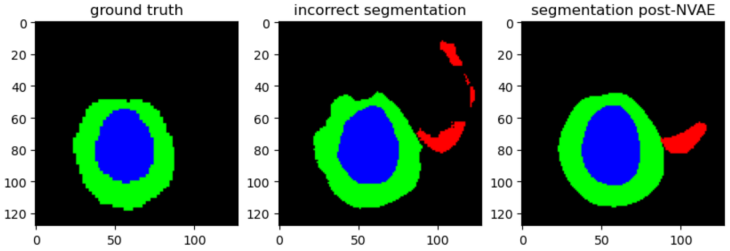


Figure 31: Bien que le dice score baisse de 0.01 points la structure gagne en cohérence.

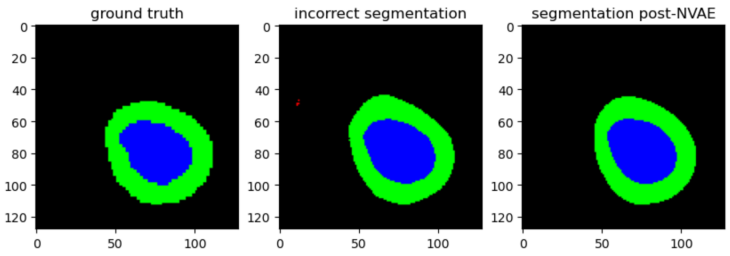


Figure 32: Qualité de suppression des artefacts

Visualisation des effets du NVAE des exemples fortement améliorés/déteriorés Nous cherchons à visualiser ici quelles sont les images les plus déteriorées et de combien le sont-elles. Nous affichons les 50 plus petites valeurs du tenseur $\mathbf{dsc}(gt, mask\_cor) - \mathbf{dsc}(gt, mask\_incor)$.

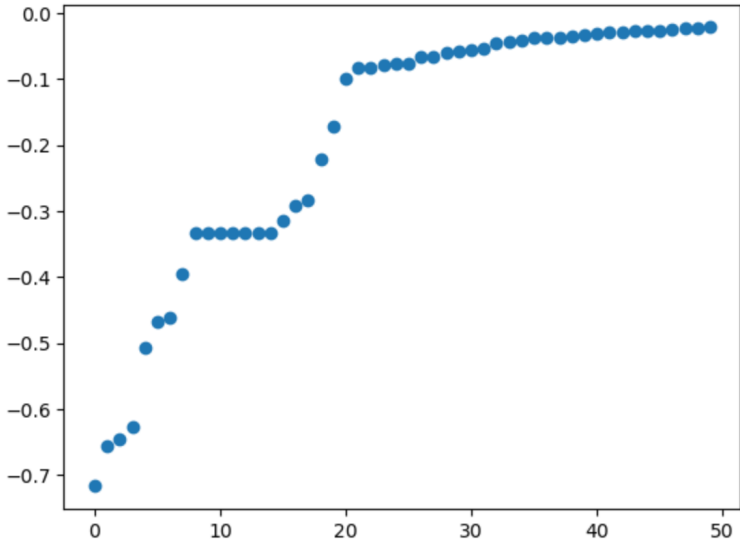


Figure 33: En abscisses le rang dans la liste des masques les plus détériorés. En ordonnées la perte de dice score.

Il arrive qu'une segmentation soit à 0.96 et qu'elle se retrouve à 0.2.

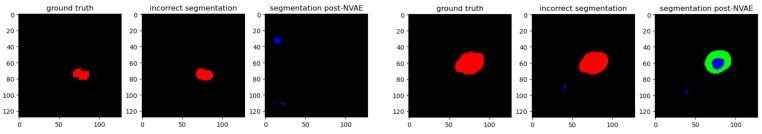


Figure 34: Masques les plus déteriorés par le NVAE.

Pour les plus améliorés nous faisons un raisonnement similaire. Nous affichons les 50 plus grandes valeurs du tenseur
$\mathbf{dsc}(gt, mask\_cor) - \mathbf{dsc}(gt, mask\_incor)$.

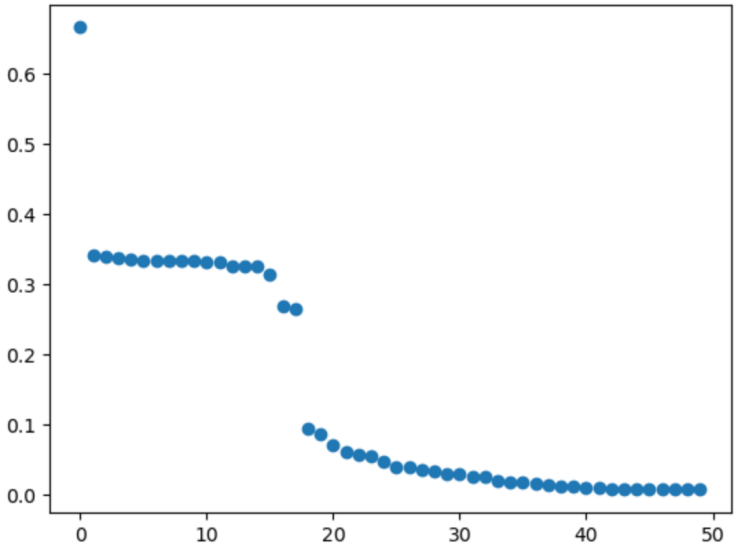


Figure 35: En abscisses le rang dans la liste des masques les plus améliorés. En ordonnées le gain de dice score.

Les meilleurs améliorations sont quelques suppressions d'artefacts mais certaines sont prometteuses.

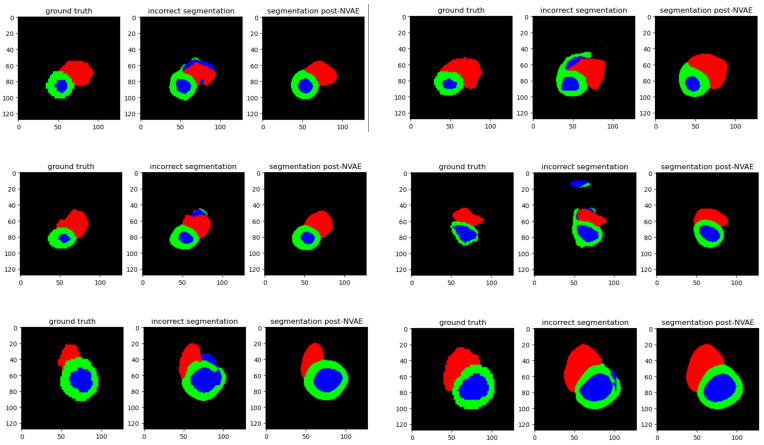


Figure 36: Masques les plus améliorés par le NVAE.

Qu'en est-il des cas à faible variation Lorsque la variation est quasi nulle les modifications sont généralement un simple effet de lissage.

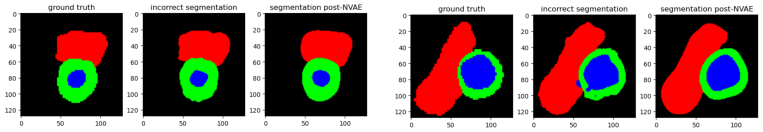


Figure 37: Masques les plus améliorés par le NVAE.

En moyenne le NVAE améliore seulement de 0.005 le dice score. Toutefois si nous regarons la moyenne sur les 100 meilleurs segmentations (représente 1/7 exemples) la moyenne des augementations de dice score passe à 0.10.

En revanche il n'est pas à ignorer que la moyenne des 100 plus grosses dégradations avec/sans NVAE est à 0.12.

On peut donc noter des effets paratagés pour le NVAE seul.

6.3 Evaluation de l'apport du contexte

Afin d'évaluer l'apport du contexte nous appliquons un raisonnement similaire à l'analyse.

Distribution générale des dice scores Nous remarquons que le contexte des masques a tendance à étaler la distribution autour de 0.93, à la différence

des autres pipelines où nous observions des pics autour de 0.95.

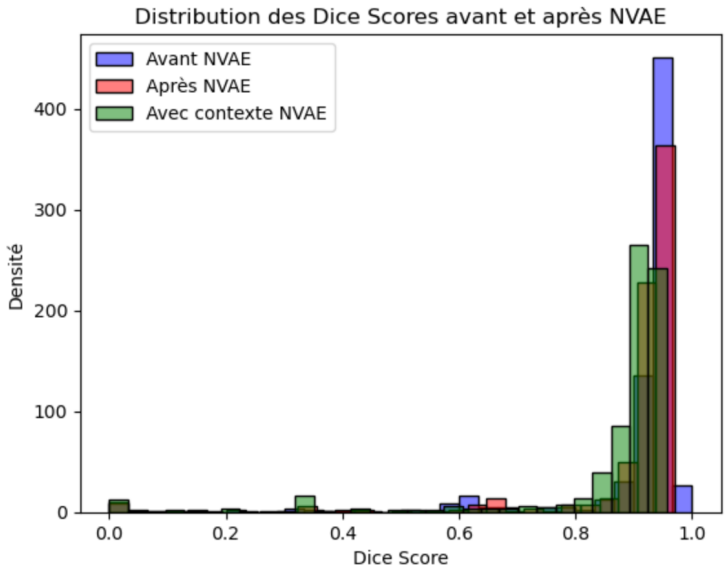


Figure 38: Distribution des dice scores dans les différentes pipelines.

En moyenne le contexte a tendance à ne pas apporter au dice score.

Dice Score	**Valeur**
dice score (gt, mask_cor)	0.8964
dice score (gt, mask_incor)	0.8973
dice score (gt, mask_context)	0.8577

Table 8: Dice score au regard de la vérité terrain pour les différentes pipelines

Nous montrons la distribution différence entre les dice scores:
$\mathbf{dsc}(gt, mask\_context) - \mathbf{dsc}(gt, mask\_incor)$.

Les valeurs positives sont des améliorations et les valeurs négatives des déteriorations. Nous montrons ensuite un histogramme afin de voir la répartition des valeurs.

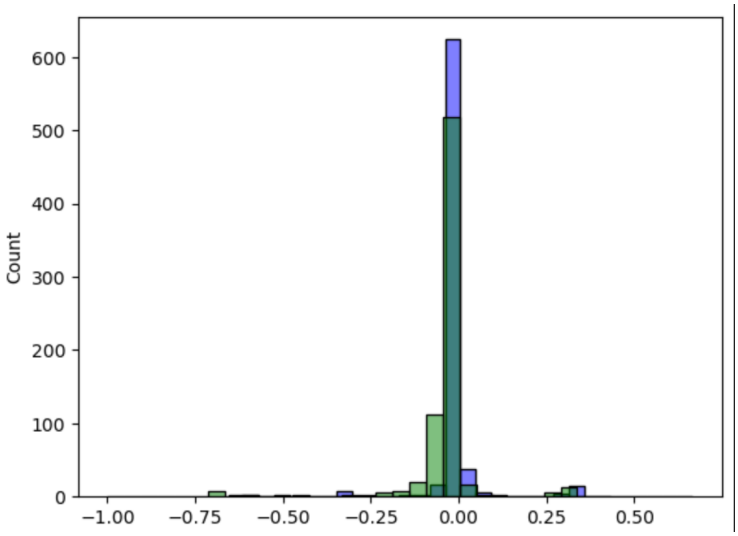


Figure 39: Différence de dice score après NVAE avec contexte —avant NVAE en vert.
Différence de dice score après NVAE sans contexte —avant NVAE en bleu.

Les exemples ont tendances à être plus déteriorés que sur la pipeline NVAE simple. En effet 112 exemples sont dégradés d'environ 0.05 points. Toutefois on note la présence de 41 exemples qui se retrouvent améliorés. Parmis ces 41 exemples, 18 récupérent 0.3 points de score.

Nous visualisons la légère tendance à la baisse sur la courbe en violon.

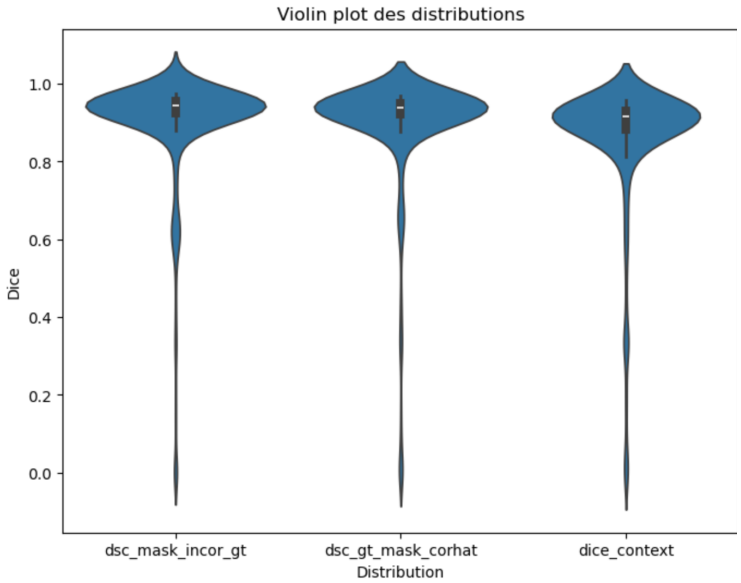


Figure 40: Courbe en violon des dice scores au regard de la vérité terrain des différentes pipelines

6.4 Cas limite: meilleurs et moins bonnes segmentations

Bien que le dice score n'augmente pas, le contexte apporte qualitativement pouvoir de correction qui est particulièrement visible.

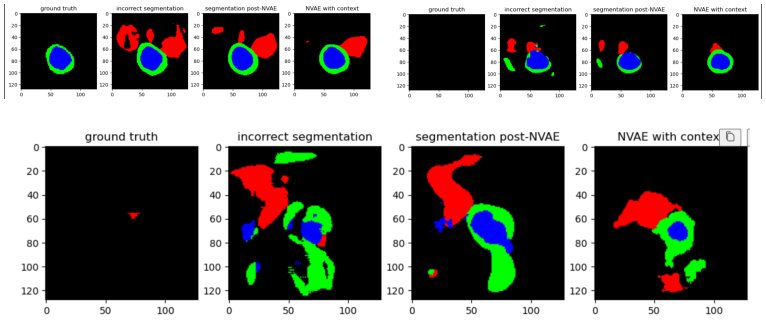


Figure 41: Correction au fil des modèles. Les images les moins bien segmentées, que deviennent elles?

Le contexte apporte une réelle connaissance des formes anatomiques et rend le NVAE plus puissant dans sa manière de rendre les formes plausibles. Toutefois

comme précédemment le modèle n'est pas en capacité de corriger des segmentations à trop faible dice score inital.

Sur les segmentations déjà fidèles, sa capacité à vouloir lisser la figure fait perdre quelques centièmes de points.

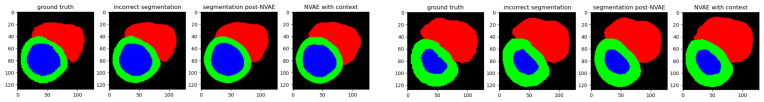


Figure 42: Correction au fil des modèles. Les images les mieux segmentées, que deviennent elles?

Les plus déteriorées Le contexte apporte qualitativement un réel apport dans la correction des formes. Toutefois il peut arriver qu'il amplifie la capacité du NVAE à "halluciner".

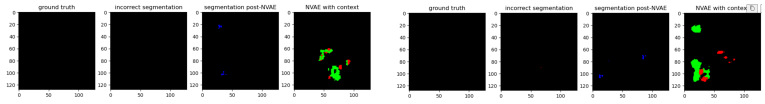


Figure 43: Hallucinations du NVAE

Les meilleures améliorations Comme vu plus haut, le contexte amplifie la correction anatomique qualitativement et également sur le dice score pour certains exemples comme le montre ces masques.

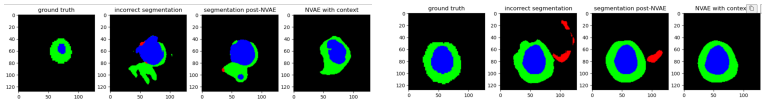


Figure 44: Correction au fil des modèles. Les images avec le plus d'améliorations

7 Conclusion

Pour conclure cette dissertation le NVAE est un modèle qui a des effets intéressants pour l'imagerie médicale de part sa capacité à se contraindre à des structures anatomiquement valides. Son apport sur la pipeline classique (sans ajout de contexte) est à nuancer. Il améliore un nombre conséquent d'exemples mais n'améliore ou déteriore que très peu la majorité des exemples. Il a toutefois tendance à quasi systématiquement lisser et à rendre cohérentes

les structures. L'apport du contexte quand à lui offre un apparentissage des formes anatomiquement valides amplifié mais à tendance à halluciner.

References

[1] Ronneberger, O., Fischer, P., Brox, T. (2015).
U-Net: onvolutional Networks for Biomedical Image Segmentation.
Disponible sur: urlhttps://arxiv.org/pdf/1505.04597

[2] Oktay, O., Schlemper, J., Le Folgoc, L. et al. (2018).
Attention U-Net: earning Where to Look for the Pancreas
Disponible sur: urlhttps://arxiv.org/pdf/1804.03999

[3] Hatamizadeh, A., Nath, V., Tang, Y. et al. (2022).
Swin UNETR: Swin Transformers for Semantic Segmentation of Brain Tumors in MRI Images
Disponible sur: urlhttps://arxiv.org/pdf/2201.01266

[4] Vahdat, A., Kautz J. (2021).
Disponible sur: `https://arxiv.org/pdf/2007.03898`

[5] Kingma, D., Welling, M. (2013)
Auto-Encoding Variational Bayes
Disponible sur: `https://arxiv.org/pdf/1312.6114`

[6] Hoffman, M., Blei, D., et al. (2013)
Stochastic Variational Inference
Disponible sur: `https://arxiv.org/pdf/1206.7051`

[7] Liu, Ziwei and Luo, Ping and Wang, Xiaogang and Tang, Xiaoou (2015)
Jeu de données Celeb-A, Deep Learning Face Attributes in the Wild,
Proceedings of International Conference on Computer Vision (ICCV)
Disponible sur:
`https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`

[8] Higgins, I., Matthey, L., et al. (2017)
β-VAE: Learning Basic Visual Concepts with a Constrained Varitional Framework Conference paper at ICLR 2017
Disponible sur: `https://openreview.net/pdf?id=Sy2fzU9gl`

[9] Burgess, C., Higgins, I., et al. (2019)
Understanding disentangling in β-VAE
Disponible sur: `https://arxiv.org/pdf/1804.03599`

[10] Sønderby, C., Raiko, T., et al. (2016)
Ladder Variational Autoencoders
Disponible sur: `https://arxiv.org/pdf/1602.02282`

[11] Y. LeCun, L. Bottou, Y. Bengio et P. Haffner (1998)
Gradient-based learning applied to document recognition,

Proceedings of the IEEE, vol. 86, no. 11, pp. 2278/2324, 1998.
DOI: 10.1109/5.726791.

[12] Goodfellow, I. et al. (2014)
Generative Adversarial Nets
Disponible sur: 1406.2661

[13] Karas, T., et al. (2019)
A Style-Based Generator Architecture for Generative Adversarial Networks

[14] Brock, A., et al. (2019)
Large Scale GAN Training for High Fidelity Natural Image Synthesis

[15] Van den Oord, A., et al. (2016)
Conditional Image Generation with PixelCNN Decoders

[16] Chen, X., et al. (2017)
PixelSNAIL: An Improved Autoregressive Generative Model

[17] Kingma, D., et al. (2018)
Glow: Generative Flow with Invertible 1×1 Convolutions

[18] Bernard, O., Lalande, A., et al. (2018)
Deep Learning Techniques for Automatic MRI Cardiac Multi-Structures Segmentation and Diagnosis:Is the Problem Solved?
International Journal of Computer Assisted Radiology and Surgery, 13(11), 1713–1720.
DOI: 10.1007/s11548-018-1812-1

[19] MONAI project
`https://docs.monai.io/en/1.3.0/`

8 Annexe A

La base de code a été organisée en 2 parties. Une première contenant le code de Freddy Jiang avec le preprocessing et la base d'implémentation NVAE. Une seconde partie est la base de code que nous avons mis en place au long de ces expériences.

8.1 Organisation shape

Cette partie de code contient les premiers scripts d'entrainement dans le dossier *scripts* ainsi que les implémentations de modèles comme le NVAE dans *arch*. Les DataLoaders eux sont implémentés à travers *data_module* et *datasets*. Un certain nombre de fonctions utiles à la visualisation se trouvent dans *utils*.

8.2 Organisation prim

La structure de *prim* est identique à celle de *shape*.

Au long de ce projet nous avons mis en place un certain nombre de classe. Les principales sont: ACDCSegSliceIndexDataModule qui implémentent le chargement du jeu de données ACDC segmentés. De même ACDC3GramSliceIndexDataModule implémente le jeu de données à 12 canaux en 3-gram. Ces classes sont trouvables dans le dossier data_modules/acdc_seg.

De plus, une classe génerale aux modèles de segmentations qui a permis l'entrainement des modèles de segmentation se trouve dans arch/.

Enfin un programme d'entrainement et de test trouvable également dans arch/ a été mis en place. Il est appelé par un script bash auquel il est possible de renseigner la nature du modèle, le jeu de données puis les hyperparamètres. Ce script enregistre les logs dans un Tensorboard.

Les scripts communiquent avec un cluster SLURM.

9 Annexe B

9.1 Notebooks découverte

Afin d'explorer les différentes classes, des notebooks trouvables dans notebooks/ ont été mis en place.

Un premier notebook permet d'explorer pas à pas le preprocessing des données puis le passage de ces mêmes données à travers chacune des couches du NVAE. Il est trouvable au nom de acdc_seg.ipynb.

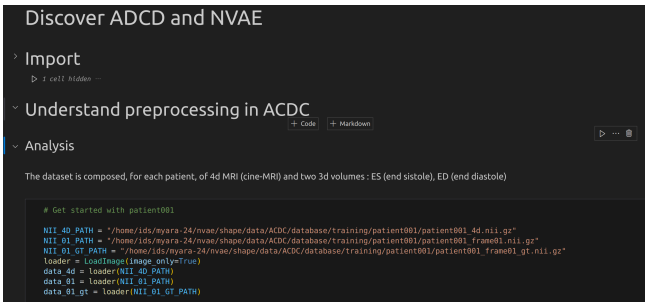


Figure 45: Notebook d'évaluation de la piepeline classique

Ensuite chacune des architectures et leurs étapes d'entrainement sont reproductibles à travers les notebooks/arch.

Enfin les deux pipeline *classique* et *3gram* sont elles accesibles dans les notebooks/seg. Il est alors possible de regarder les différentes courbes de l'évaluation des pipelines et de modifier les valeurs au besoin.

Figure 46: Notebook d'évaluation de la piepeline classique

9.2 Archtecture NVAE — version implémentée

Voici le schéma de l'architecture NVAE implémentée.

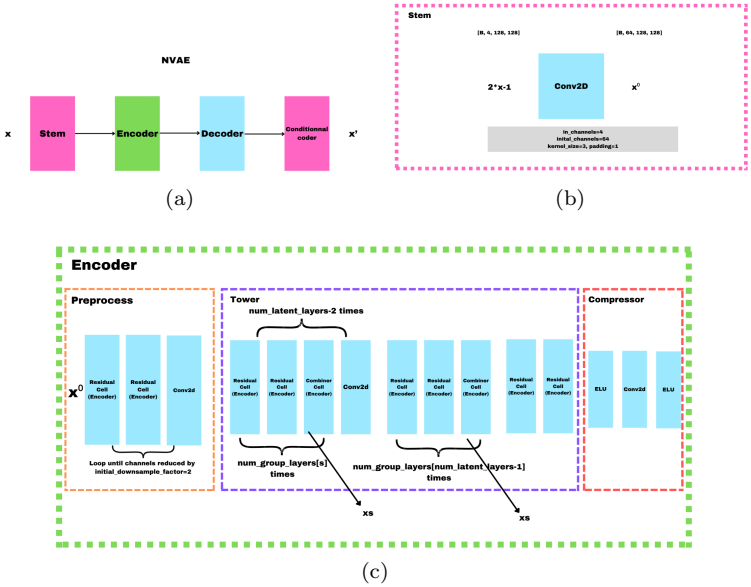


Figure 47: Schéma de principe de l'implémentation NVAE

Fin .