PROJET
DEEP LEARNING

β-VAE : Learning Basic Visual Concepts with a Constrained Variational Autoencoders

EVRAERD Gauthier, SULEJMANI Armen

M2 CNS SA 2023-2024

Table des matières

1 Introduction **3**

2 Présentation **3**
2.1 Base de donnée 3
2.2 β-VAE 4

3 Architecture du VAE **8**
3.1 Architecture de l'article 8
3.2 Notre architecture 8

4 Résultat **9**

5 Visualisation de l'espace latent **11**

6 Capture des facteurs de variation **12**

7 Comparaison de la capture des facteurs de variation avec l'architecture de l'article **15**

8 Conclusion **18**

1 Introduction

L'article sur lequel nous utilisons propose d'introduire un paramètre β dans un Variational Autoencoder (VAE). Cette modification a conduit à la création d'une nouvelle approche appelée β-VAE, qui vise à améliorer la séparation des facteurs de variation dans des données non supervisées. Cette approche a été développée pour résoudre les problèmes de performance de la méthode VAE originale sur des ensembles de données plus complexes. La représentation latente apprise par un VAE peut souvent regrouper plusieurs facteurs de variation dans une seule dimension de l'espace latent. Les VAE standard peuvent avoir du mal à séparer ces facteurs, ce qui peut entraîner des représentations latentes moins interprétables et utilisables.

Dans ce projet, nous allons explorer les détails de cette approche et l'adapter à une nouvelle base de donnée qui n'a pas été utilisée dans l'article. Pour ce faire, nous allons remplacer la base de données d'entrée par celle de MNIST, en ajoutant toutefois une complexité en fournissant les images avec une rotation aléatoire. Le but est de permettre au modèle d'apprendre à reconstruire et générer des chiffres avec ou non une rotation et de suffisamment séparer les facteurs de variations pour les capturer, notamment la rotation.

2 Présentation

2.1 Base de donnée

Nous avons utilisé le dataset MNIST pour entraîner notre VAE. Ce dataset est constitué de 60 000 images pour la base d'entraînement et de 10 000 images pour la base de test. Chaque image est initialement de taille 28x28 en niveau de gris. Nous avons décidé de les redimensionner en 64x64 pour pouvoir réutiliser l'architecture présenté dans l'article et s'appuyer sur celle-ci pour en réaliser une nouvelle.
Pour créer le dataset Rotated MNIST, nous avons effectué sur chaque image une rotation aléatoire entre -90° et 90°. Nous avons trouvé pertinent de ne pas seulement utilisé Rotated MNIST pour l'entraînement mais de rajouter également le dataset MNIST de base pour l'entraînement. En faisant cela, nous pensons que notre VAE est capable de mieux apprendre à générer des chiffres avec ou non une rotation.

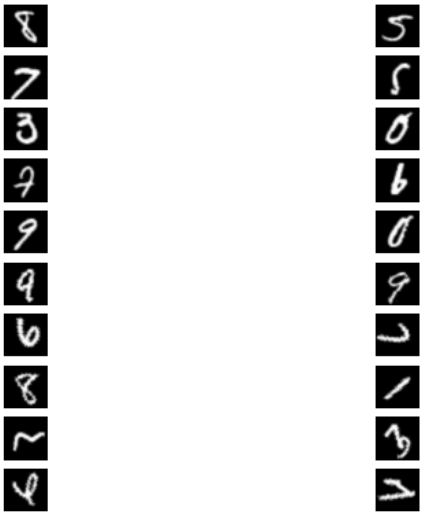

Figure 1 – Échantillon de notre dataset

2.2 β-VAE

Le terme β se réfère à un paramètre spécifique utilisé pour réguler la manière dont un modèle particulier, appelé Variational Autoencoder "VAE", apprend à représenter les données. Dans le contexte du VAE, ce paramètre, appelé "coefficient de régularisation bêta", contrôle l'équilibre entre la fidélité de la représentation des données et une contrainte qui encourage une représentation plus organisée.
Ce paramètre est ajouté dans la fonction de coût du VAE. L'idée est que le paramètre β influe sur la manière dont le modèle privilégie la séparation des facteurs de variation dans la représentation latente. Un β plus élevé force le modèle à apprendre une représentation plus démêlé des données, c'est-à-dire à mieux séparer les différents aspects des données dans la représentation latente.
L'objectif étant de trouver le bon compromis entre la précision de reconstruction et la bonne séparation des facteurs de variation en jouant sur le paramètre β.

Nous allons étudier l'impact du paramètre β en fonction de sa valeur pour réussir à trouver la valeur la plus intéressante pour notre ensemble de donnée. Sachant que β est compris dans [1 ; +inf].

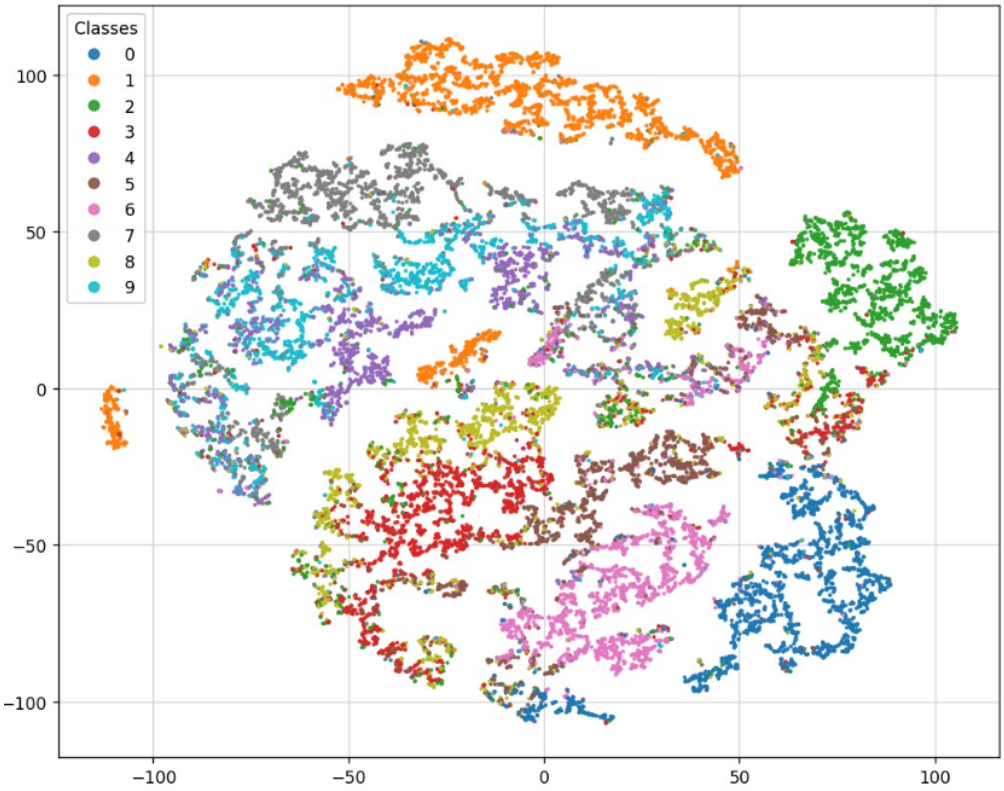

FIGURE 2 – Espace latent avec $\beta = 4$

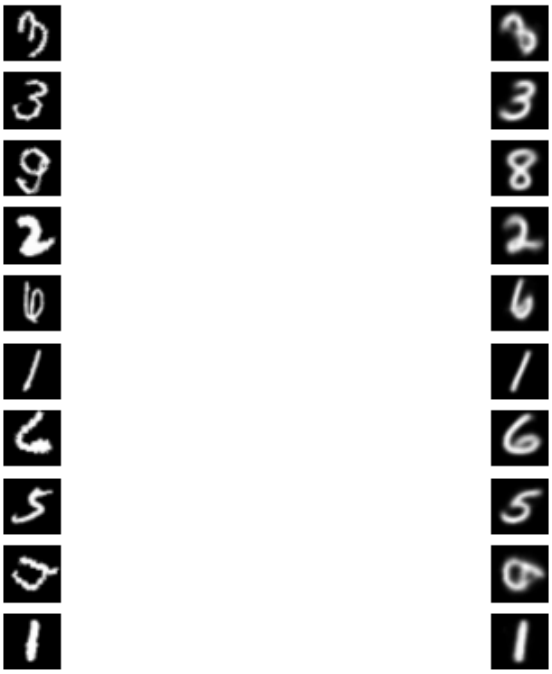

FIGURE 3 – Reconstruction avec $\beta = 4$

Nous pouvons voir que l'espace latent est assez prononcé avec un β "faible", il permet la fidelisation des données d'entrée. De plus, la reconstruction est très similaire avec l'image de base.

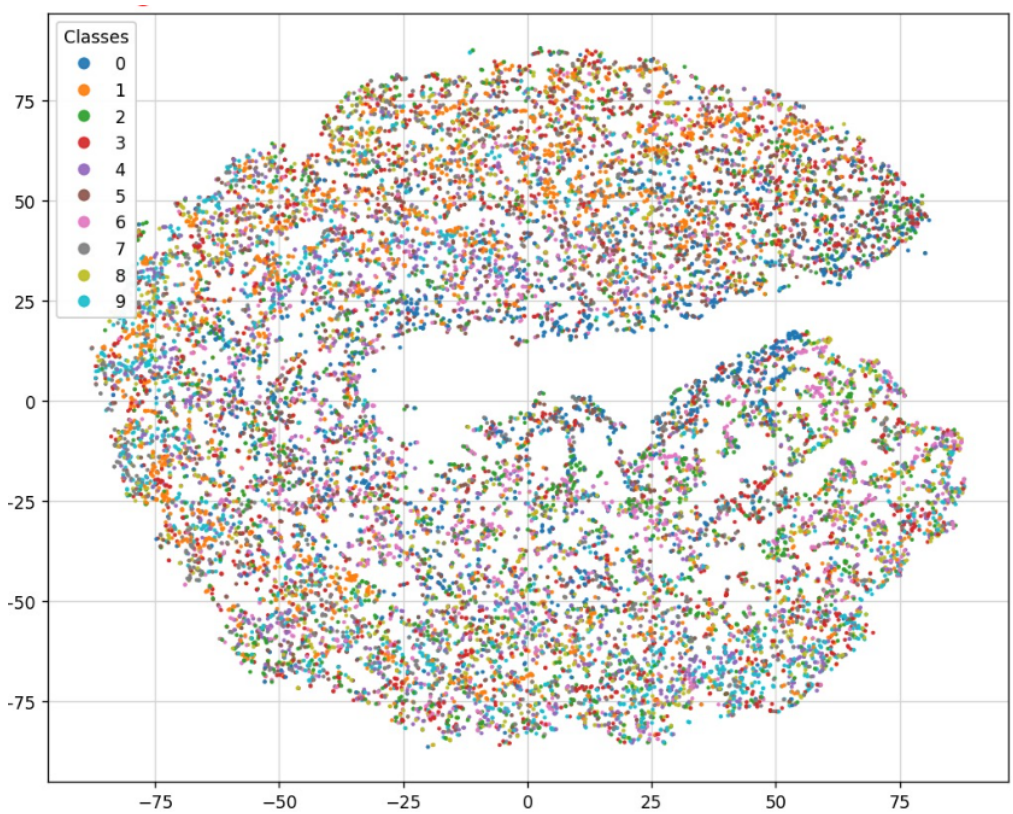

FIGURE 4 – Espace latent avec $\beta = 128$

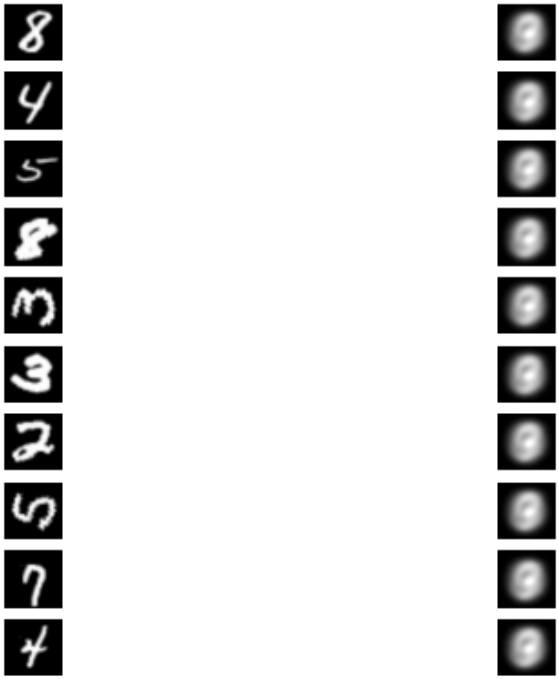

FIGURE 5 – Reconstruction avec $\beta = 128$

Ici, l'opacité de l'espace latent s'accentue avec un β particulièrement élevé pour le β-VAE, qui complexifie davantage la déchiffrabilité des représentations latentes. De plus, le modèle est incapable de reconstruire une image.

En ajustant la valeur de ce paramètre β, on peut influencer la qualité de la représentation apprise par le modèle. Un β plus élevé peut conduire à une représentation plus structurée, tandis qu'un β plus bas peut permettre au modèle de se concentrer davantage sur la fidélité de la reconstruction des données d'entrée. En résumé, le β aide à trouver le juste équilibre entre une représentation précise des données et une représentation latente plus organisée.

3 Architecture du VAE

3.1 Architecture de l'article

Dans l'article, il est présenté 2 architectures différentes de VAE, une en MLP et une en CNN. C'est l'architecture utilisant un CNN qui nous intéresse vu que nous traitons des images. L'architecture en CNN comporte 4 couches de convolutions sous la forme : 32, 32, 64, 64 comme nombre de channel dans chaque couche. L'architecture est représentée dans les figures ci-dessous.

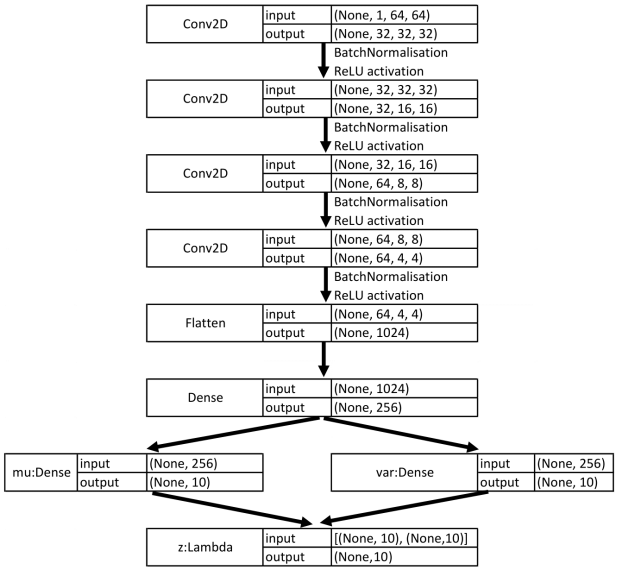

FIGURE 6 – Architecture de l'encodeur du VAE utilisé dans l'article

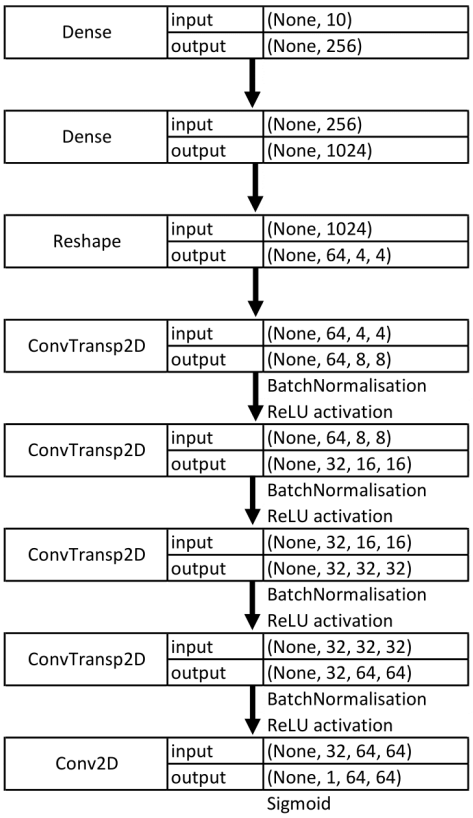

FIGURE 7 – Architecture du décodeur du VAE utilisé dans l'article

3.2 Notre architecture

Nous avons décidé de reprendre l'architecture de l'article utilisé pour un VAE avec un CNN que nous avons décidé de complexifier en rajoutant des couches de convolution. L'objectif est de tenter de capturer de meilleurs facteurs de variation qu'avec l'architecture de l'article. Entre chaque couche de convolution, nous mettons une couche de batch normalisation et une fonction d'activation linéaire. Notre encodeur est de la forme 32, 32, 64, 64, 128, 128, 256 comme nombre de channel dans nos différentes couches de convolution. Notre décodeur est principalement l'inverse de notre encodeur. A la dernière couche de notre décodeur, nous utilisons une fonction d'activation Sigmoid à la place de la linéaire.

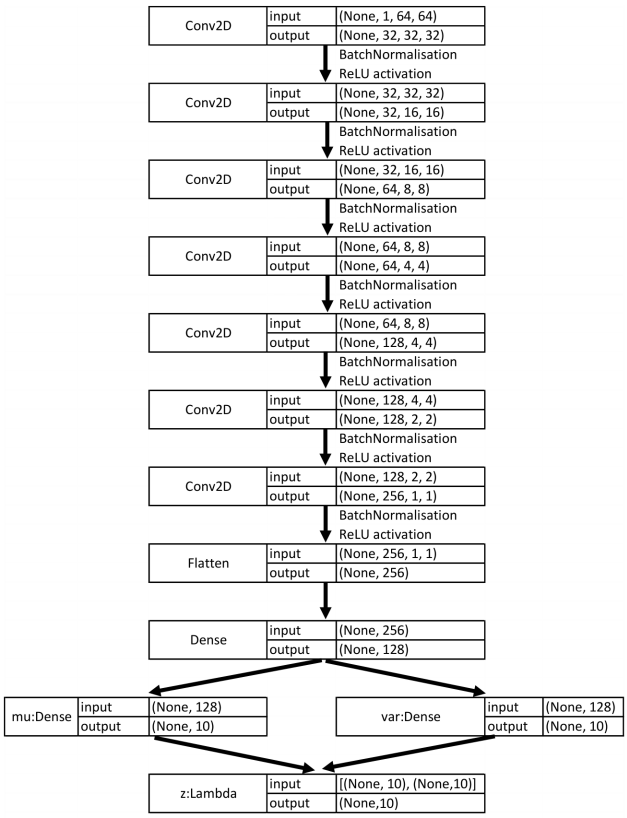

FIGURE 8 – Architecture de l'encodeur

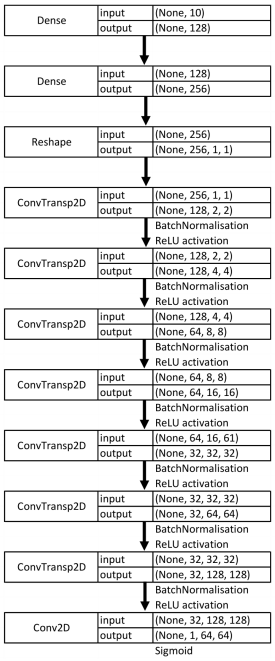

FIGURE 9 – Architecture du décodeur

4 Résultat

Pour entraîner notre modèle, nous avons effectué 15 epochs. Comme paramètre d'entraînement nous avons pris un batch size de 64, $\beta = 4$ et une dimension d'espace latent à 10. A la fin de notre entraînement nous obtenons les résultats suivants :

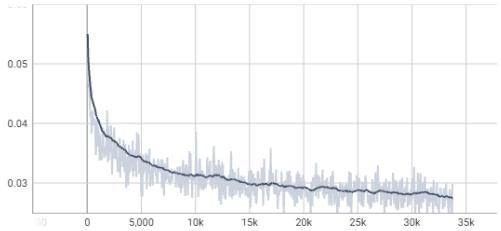

FIGURE 10 – Training Loss

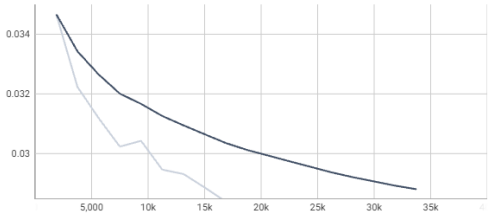

FIGURE 11 – Validation Loss

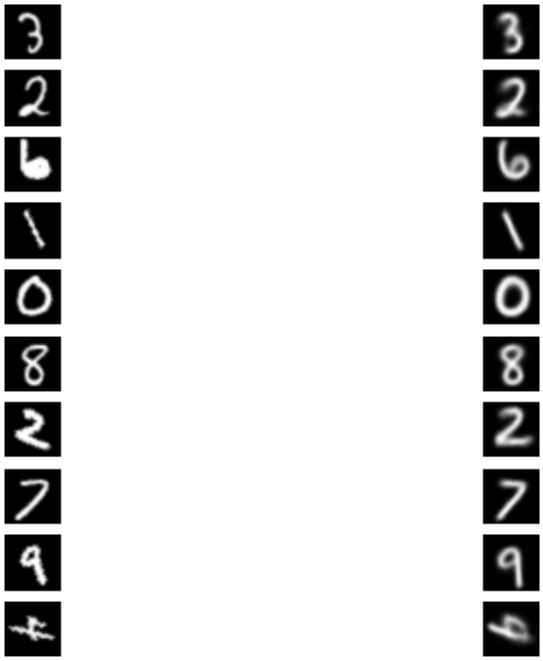

FIGURE 12 – Reconstruction (à droite) des images en entrée (à gauche) avec notre VAE

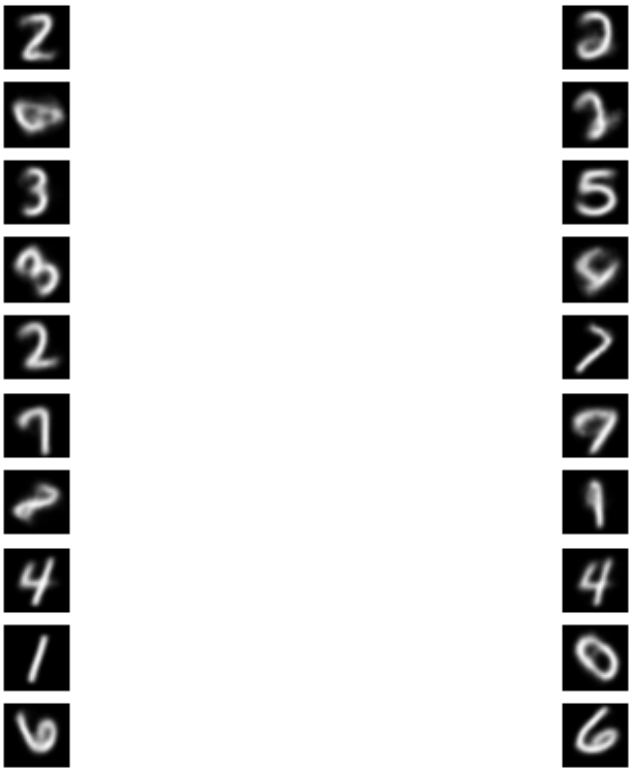

FIGURE 13 – Génération de chiffres à partir de points aléatoire dans l'espace latent avec notre VAE

5 Visualisation de l'espace latent

Pour visualiser notre espace latent ainsi que sa répartition, nous avons donné à notre VAE notre dataset de test et récupérer l'encodage qu'il produit pour chaque image. Mais il nous est impossible d'afficher l'encodage de chaque image dans un graphe à cause du grand nombre de dimensions. En effet, notre espace latent comporte 10 dimensions et ce n'est pas compréhensible de faire un graphique à 10 dimensions. Nous devons d'abord réduire les dimensions de notre l'espace latent. Pour cela, nous pouvons appliquer une PCA ou TSNE. Le principe est de réduire la dimension des données en préservant autant que possible l'information originale. Après l'utilisation de TSNE, pour réduire notre espace latent à 2 dimensions, nous obtenons ce graphique suivant :

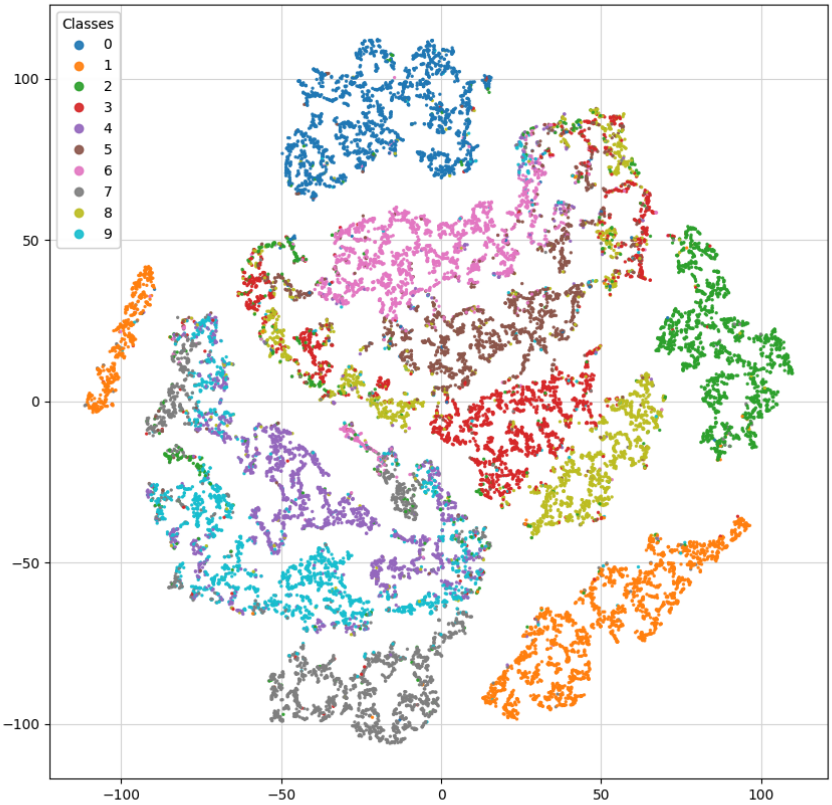

FIGURE 14 – Visualisation en 2 dimensions de l'espace latent

On peut observer que les différents chiffres sont réunis en clusters de taille plus ou moins grande. On remarque que notre VAE réussit à bien répartir chaque chiffre dans l'espace latent. Chaque partie de l'espace latent correspond à un seul chiffre. De plus, on peut observer plusieurs cluster à différentes positions de l'espace latent pour un même chiffre. On peut expliquer ces différents clusters par l'introduction de la rotation des images dans la base de données. Enfin, on peut remarquer que les clusters avec des chiffres similaires sont très proche dans l'espace latent. Le cluster principal du 1 et du 7 sont à côté, en effet le 1 et le 7 sont très similaire au niveau de leur forme. Également pour le cluster principal du 3 et du 8 qui sont juste à côté comme le 3 et le 8 sont aussi très similaires dans leur forme globale. Enfin, il semble que les chiffres possédant des arrondis ou une boucle dans leur forme sont d'un côté de l'espace latent et ceux plus linéaires, sans boucle, dans leur forme d'un autre côté de l'espace latent. Il y a que le chiffre 9 qui se trouve du côté du 1, 4 et 7.

6 Capture des facteurs de variation

Nous nous avons tenté de comprendre notre espace latent et de capturer certains facteurs de variation. Un facteur de variation fait référence à une dimension spécifique de l'espace latent associé à une caractéristique telle que la rotation ou l'épaisseur de trait par exemple.
En générant plusieurs images à partir de points aléatoires et en les affichant, nous avons pu nous rendre

compte que notre VAE était capable de reconstruire n'importe quel chiffre mais surtout qu'il était capable de jouer sur la rotation des chiffres ainsi que sur leur forme globale.
Nous avons commencé par essayer de déterminer les dimensions de notre espace latent qui sont responsable de ces variations lors de la génération. Pour cela, nous avons pris un point aléatoire dans l'espace latent et pour chaque dimension du point que l'on génère, modifié sa valeur. Pour vraiment se rendre compte de l'effet d'une dimension sur la génération, nous avons fait une interpolation entre les valeurs -2 et 2, pour observer l'effet de cette dimension à intervalles réguliers.

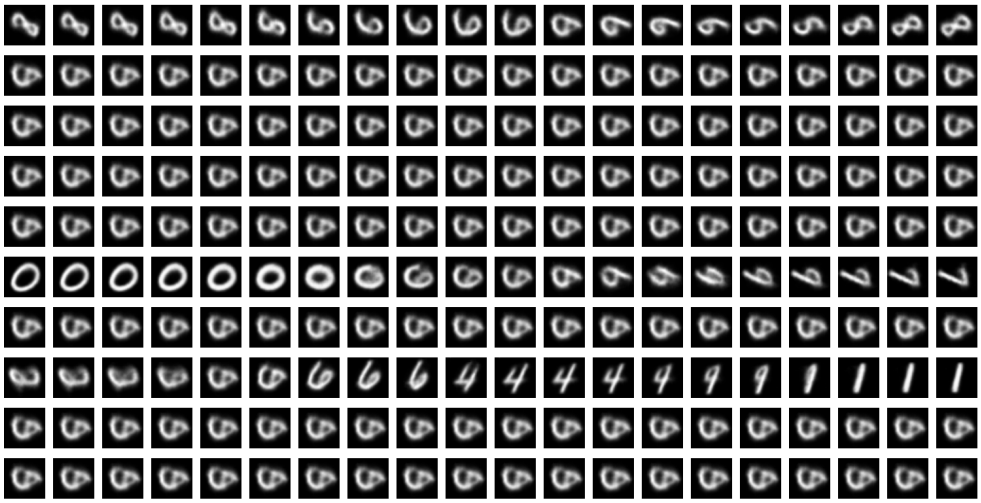

Figure 15 – Visualisation des variations pour chaque dimension à partir d'un point aléatoire dans l'espace latent

Sur cette image ci-dessus, chaque ligne représente une dimension et les colonnes les différents intervalles sur une dimension. En effectuant cette méthode à plusieurs reprises avec des points différents, on se rend compte que surtout 3 dimensions impactent grandement la génération des chiffres. Ceux sont les dimensions 1, 6 et 8.

Comme en prenant un point aléatoire, on n'obtient pas à chaque fois une génération qui ressemble à un chiffre. Nous avons décidé de reprendre cette méthode mais en utilisant cette fois ci l'encodage d'un chiffre du dataset dans notre espace latent à la place d'un point aléatoire. Nous avons décidé de prendre tous les chiffres différents soit de 0 à 9 et de refaire notre méthode pour observer principalement l'effet des dimensions 1, 6 et 8 sur la reconstruction. Nous avons ensuite affiché dimension par dimension le résultat de l'interpolation de chaque dimension pour chaque chiffre. On retrouve donc sur chaque ligne les chiffres de 0 à 9 et sur les colonnes les différents intervalles.

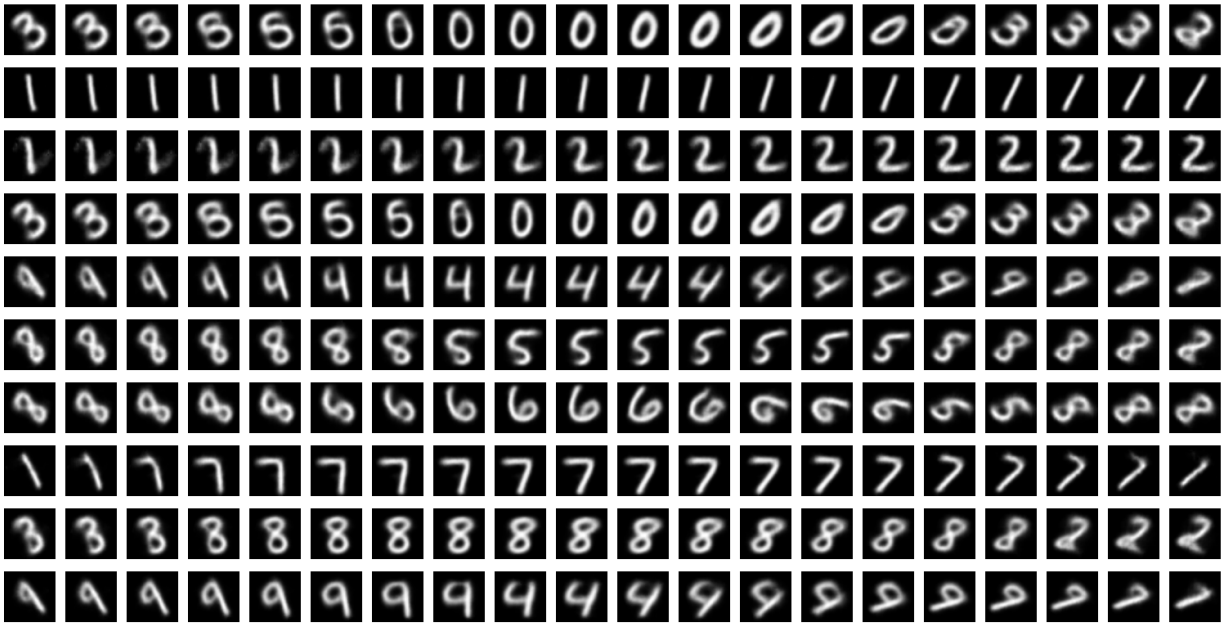

Figure 16 – Visualisation de la dimension 1

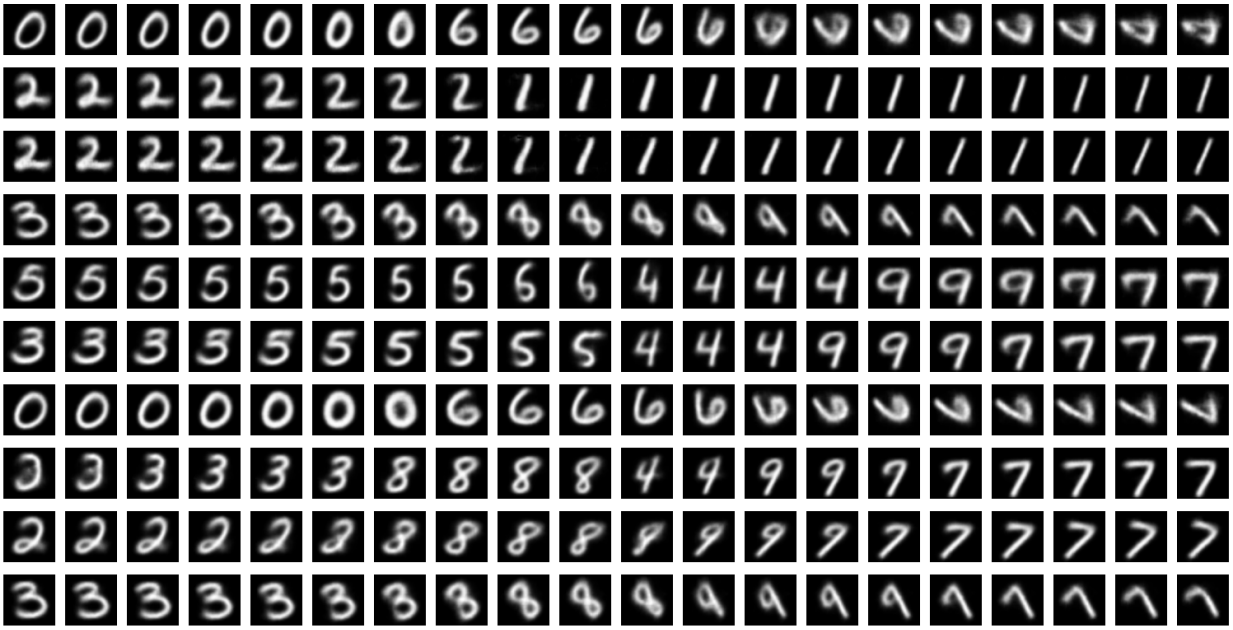

Figure 17 – Visualisation de la dimension 6

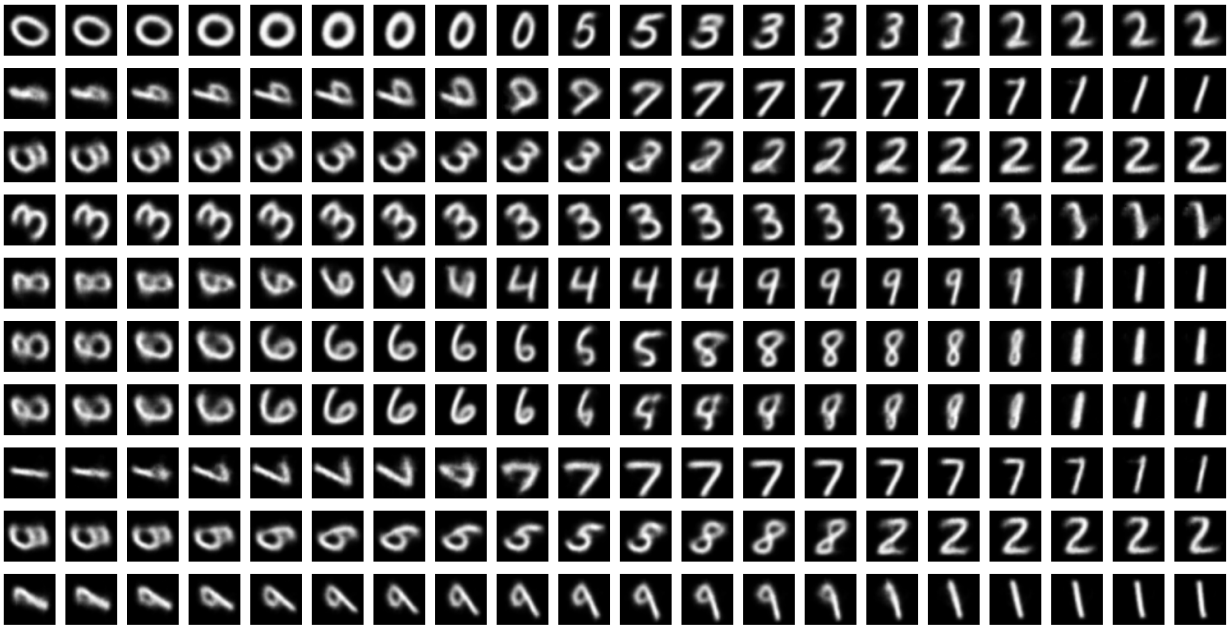

Figure 18 – Visualisation de la dimension 8

Pour commencer, on peut observer que la dimension 1 (Figure 16) est principalement responsable de la rotation lors de la génération d'un chiffre. De plus, comme on interpole entre -2 (à gauche) et 2 (à droite), on peut en déduire qu'une valeur négative dans cette dimension a tendance à créer une rotation dans le sens contraire des aiguilles d'une montre et inversement. On se rend très bien compte de son effet pour la ligne du 1 ou du 9, à chaque extrémité, on obtient sa symétrie. Mais il semble également que cette dimension ne soit pas que responsable de la rotation sinon on ne passerai pas du 3 au 0 puis au 3 sur la première ligne.

Ensuite, on peut remarquer que la dimension 6 (Figure 17) est responsable d'arrondir plus ou moins la forme générée. On se rend bien compte qu'avec des valeurs négatives, notre VAE arrondi par exemple les extrémités du 1, ou crée des boucles plus ou moins fermées. Enfin, avec des valeurs positives sur cette dimension, notre VAE supprime totalement les boucles et les arrondis, il ne génère plus que des lignes droites.

Finalement, on se rend bien compte que les facteurs de variation sont encore beaucoup liés entre-eux et peu séparés comme le β-VAE est censé le permettre. La visualisation de la dimension 8 (Figure 18) le montre bien. On retrouve la caractéristique des deux précédentes dimensions qui sont la rotation et l'ajout de boucle ou de ligne droite. Notre tentative n'est donc pas très convaincante.

7 Comparaison de la capture des facteurs de variation avec l'architecture de l'article

Pour finir, nous avons décidé de comparer les facteurs de variation de notre architecture avec ceux capturés par l'architecture de l'article. Nous avons donc entraîné le VAE de l'article avec les mêmes paramètres que la nôtre : 15 epochs, $\beta = 4$ et une dimension d'espace latent à 10.
Pour capturer les facteurs de variation, nous utilisons les mêmes méthodes. Nous commençons par parcourir toutes les dimensions de l'espace latent à intervalle régulier à partir d'un point aléatoire dans celui-ci.

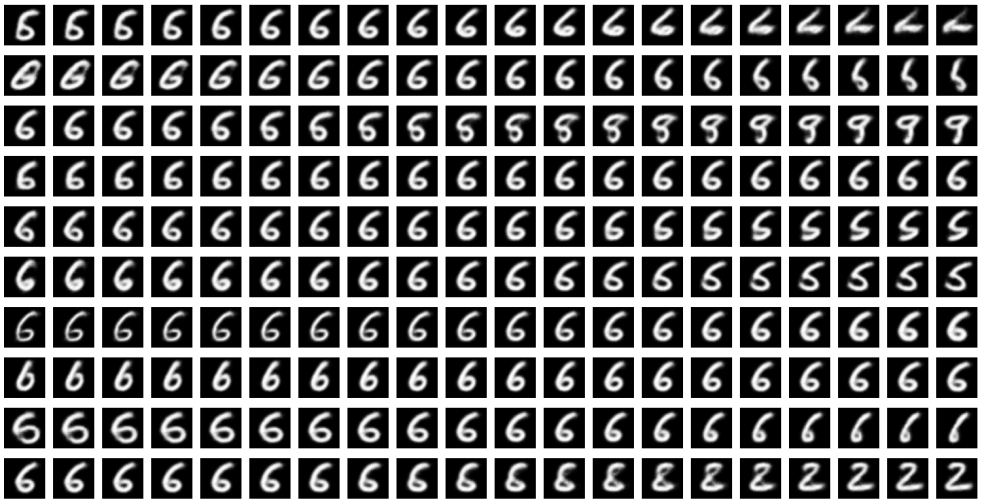

FIGURE 19 – Visualisation des variations pour chaque dimension à partir d'un point aléatoire dans l'espace latent

On peut observer que presque toutes les dimensions sont associées à un facteur de variation contrairement à nous. La séparation des facteurs de variation à mieux fonctionner ici malgré la même valeur de β et la même nombre de dimension d'espace latent. Ensuite, contrairement à nous on semble remarquer qu'il n'y a pas de dimension qui ai capturé la rotation. Par contre, on peut remarquer que de nouveaux facteurs de variation ont été capturés.

Pour mieux se rendre compte de ces nouveaux facteurx de variation, nous allons réutiliser la méthode avec tous les chiffres de 0 à 9 et parcourir chaque dimension.

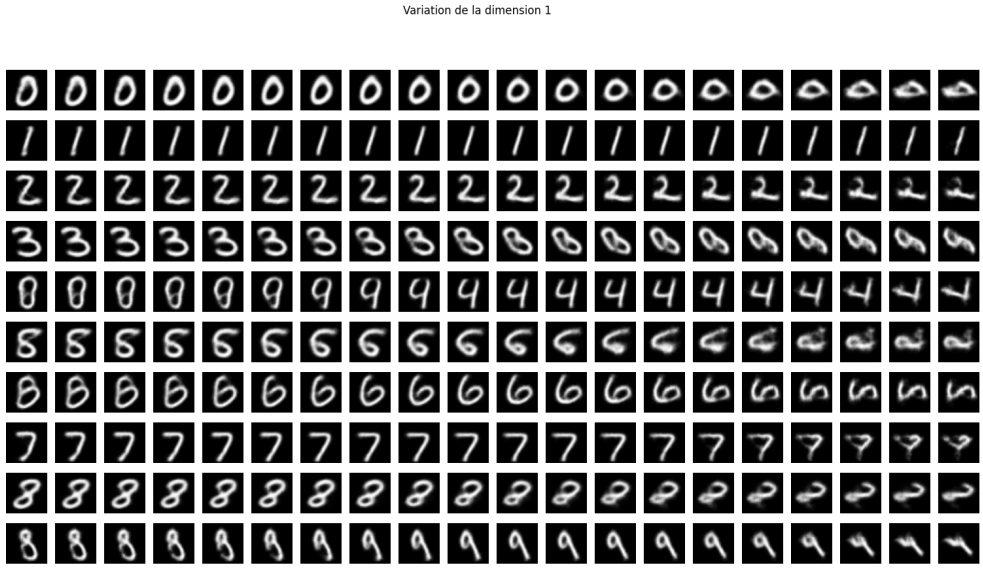

FIGURE 20 – Visualisation de la dimension 1

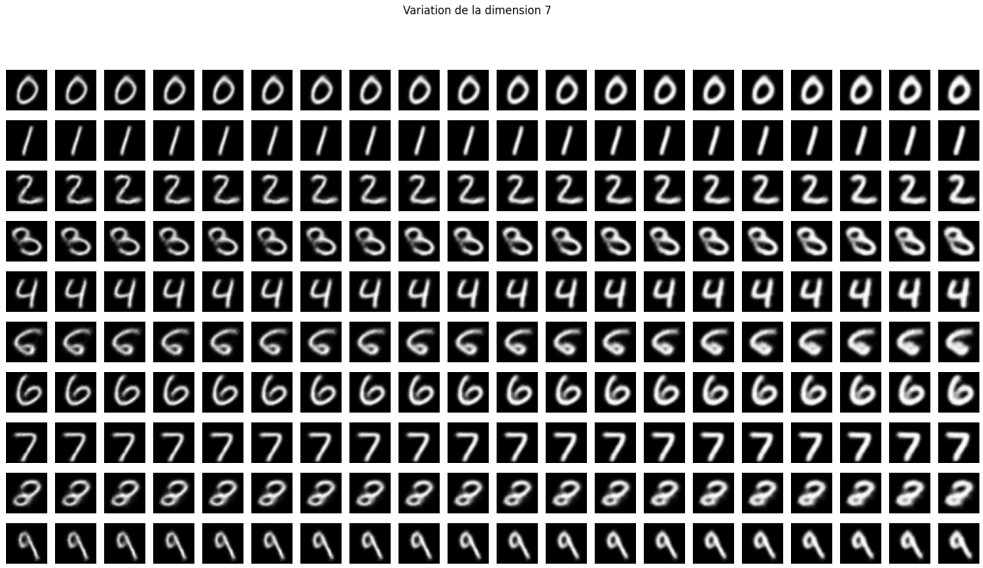

FIGURE 21 – Visualisation de la dimension 7

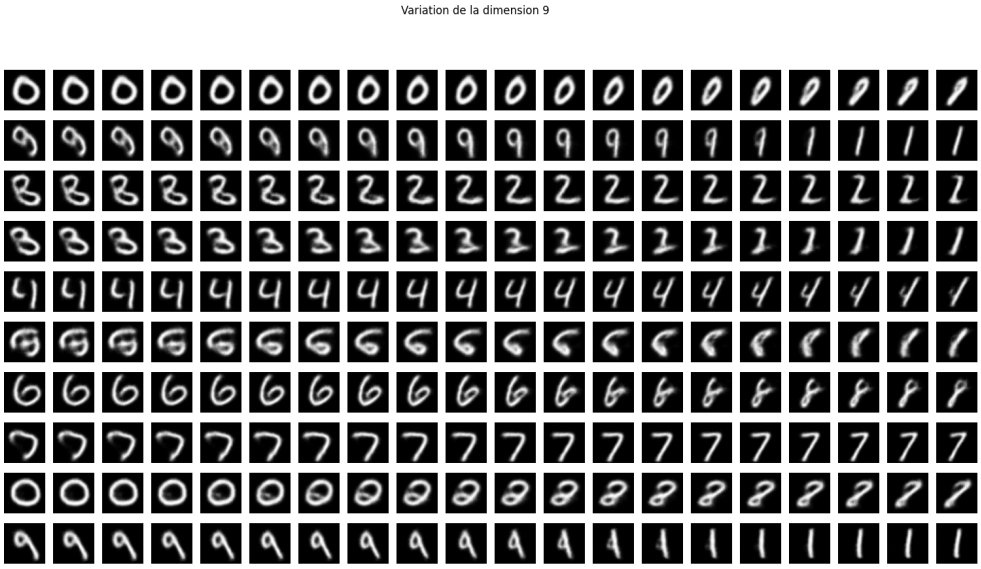

Figure 22 – Visualisation de la dimension 9

Pour commencer, la dimension 1 (Figure 20) est associée à la hauteur du chiffre lors de la génération. Elle a un impact sur la hauteur ou sur son ratio vertical. Ensuite, on retrouve un autre facteur de variation en lien avec le précédent. Il a été associé à la dimension 9 (Figure 22). Cette dimension permet de modifier la largeur du chiffre lors de la génération. Elle influe le ration horizontal cette fois ci. Enfin, le dernier facteur de variation interprétable est associé avec la dimension 7 (Figure 21). Cette dimension permet directement de jouer sur l'épaisseur du trait qui constitue le chiffre. On peut générer un trait tout fin ou un trait très épais par exemple.

On retrouve certes des facteurs de variation différents que ceux que nous avions trouvé avec notre architecture mais de nombreuses dimensions sont associées à des caractéristiques qui ne sont très difficilement interprétables ou compréhensibles pour un humain. De plus, elle n'a pas réussi à capturer la rotation qui était un des objectifs de ce projet.

8 Conclusion

Pour conclure notre projet de Deep Learning β-VAE met en lumière plusieurs points clés. Tout d'abord, l'approche β-VAE a été explorée en profondeur, mettant en évidence sa capacité à améliorer la séparation des facteurs de variation dans des données non supervisées. Ensuite, l'étude a permis de visualiser l'impact du paramètre β l'espace latent du VAE. Notre adaptation pour tenter de capturer de meilleurs facteurs de variation n'est pas concluante. Nous avons certes réussi à capturer la rotation sur une dimension mais les dimensions restent encore beaucoup trop lié entre-elles. Le paramètre β n'a pas réussi à bien séparer les facteurs de variation. Cependant, avec l'architecture présenté dans l'article, nous avons bien réussi à capturer plusieurs facteurs de variation et ces facteurs sont totalement indépendants des autres. Le paramètre β à cette fois bien fonctionné. Il est intéressant de remarquer que les deux modèles n'ont pas capturer les mêmes facteurs de variation. Celui de l'article n'a pas capturer la rotation. Enfin, les comparaisons approfondies entre notre modèle et celui de l'article soulignent l'importance de l'adaptation et de l'exploration approfondie des techniques d'apprentissage automatique pour mieux capturer et exploiter les facteurs de variation dans les données non supervisées.