MEMOIRE DE MINI PROJET

BRUSCO : Modélisation et Structuration de l'espace latent par Adversarial Autoencoders pour l'analyse et génération d'images

Table de matière

Introduction générale

1. Contexte général

Le développement de représentations latentes robustes, organisées et exploitables constitue un défi majeur dans le domaine de l'apprentissage profond, en particulier pour les tâches de modélisation non supervisée et de génération de données. Les autoencodeurs classiques permettent de projeter des données de grande dimension dans un espace latent de dimension réduite afin d'en faciliter la reconstruction. Toutefois, ces modèles n'imposent aucune contrainte explicite sur la distribution de l'espace latent, ce qui conduit généralement à des représentations désorganisées, difficiles à manipuler et peu adaptées à la génération contrôlée de nouvelles données. Cette limitation restreint considérablement leur capacité à capturer des structures complexes ou à réaliser des interpolations pertinentes entre les données d'entrée.

Afin de surmonter ces insuffisances, les **Adversarial Autoencoders (AAE)** ont été proposés comme une extension générative des autoencodeurs traditionnels, combinant l'apprentissage de reconstruction avec une régularisation adversariale. Dans ce cadre, l'encodeur est entraîné non seulement pour produire des représentations fidèlement reconstructibles par le décodeur, mais également pour aligner la distribution de l'espace latent sur une distribution *a priori* définie, telle qu'une loi gaussienne. Cette contrainte est imposée par un discriminateur adversarial dont le rôle est de distinguer les codes latents issus de la distribution prior de ceux générés par l'encodeur. L'encodeur apprend ainsi à tromper le discriminateur, ce qui permet progressivement de structurer l'espace latent selon la distribution cible.

Bien que l'AAE permette d'obtenir un espace latent plus lisse et régularisé, il ne garantit pas une organisation explicite des représentations selon des informations sémantiques telles que les classes des données. Pour répondre à cette limitation, le **Conditional Adversarial Autoencoder (CAAE)** introduit une information conditionnelle, généralement la classe associée aux données, dans le processus d'encodage et de discrimination. Cette approche permet de structurer l'espace latent de manière plus discriminante, en favorisant une meilleure séparation inter-classes et en offrant un contrôle accru sur la génération des données.

Dans une perspective d'amélioration supplémentaire, le **CAAE hybride** combine les avantages de la régularisation adversariale, du conditionnement par classe et de mécanismes d'apprentissage renforcés afin d'accroître la robustesse du modèle et la qualité des

représentations latentes. Cette approche hybride vise à produire un espace latent à la fois bien structuré globalement et sémantiquement cohérent, tout en améliorant les performances de reconstruction et de génération.

Dans le cadre de ce travail, nous exploitons successivement un **AAE**, un **CAAE** et un **CAAE hybride** afin d'étudier et de comparer leurs capacités à structurer l'espace latent du jeu de données **MNIST**, composé d'images de chiffres manuscrits. L'objectif est d'imposer une distribution gaussienne à l'espace latent tout en analysant l'impact du conditionnement et de l'hybridation sur la qualité des représentations apprises, la séparation des classes et la génération contrôlée d'images. Cette étude ouvre la voie à des applications avancées en modélisation générative, en apprentissage non supervisé et en visualisation des manifolds latents.

2. Problématique :

Dans les systèmes de vision artificielle et d'apprentissage profond, l'efficacité des performances est étroitement liée aux représentations latentes que les modèles ont apprises. Malgré leur efficacité en termes de reconstruction des données, les autoencodeurs traditionnels ne définissent aucune restriction sur la configuration de l'espace latent, ce qui peut entraîner des représentations chaotiques et peu utilisables pour des tâches complexes telles que la génération contrôlée, l'interpolation ou encore la classification. Ce manque de structuration entrave aussi la faculté du modèle à faire preuve de généralisation et à générer des données crédibles.

3. Objectif de Projet

Ce projet vise à concevoir et à mettre en œuvre un **Adversarial Autoencoder (AAE)** afin de structurer l'espace latent en le contraignant à suivre une distribution spécifique, notamment une distribution gaussienne. En s'appuyant sur le jeu de données **MNIST**, composé d'images de chiffres manuscrits, l'objectif est d'apprendre des représentations latentes compactes, continues et contrôlées, facilitant ainsi la génération d'images cohérentes et l'interpolation fluide au sein de l'espace latent. Afin d'enrichir cette approche, le projet étend l'AAE vers un **Conditional Adversarial Autoencoder (CAAE)**, qui intègre l'information de classe dans le processus d'apprentissage, permettant une structuration plus discriminante de l'espace latent et une génération conditionnée des images. Par ailleurs, un **CAAE hybride** est proposé afin de combiner les avantages de la régularisation adversariale, du conditionnement par classe et de mécanismes d'apprentissage renforcés, dans le but d'améliorer la robustesse du modèle et la qualité des représentations apprises. Ce travail vise également à analyser l'impact de la régularisation adversariale sur la qualité des représentations latentes, en comparant les

performances de l'AAE, du CAAE et du CAAE hybride avec celles d'un autoencodeur classique. Enfin, cette étude met en évidence l'intérêt de ces modèles pour des applications liées à la modélisation générative, à la visualisation de l'espace latent et à l'apprentissage non supervisé.

4. Organisation de projet

Notre travail s'articule autour de Cinque chapitres principaux et d'une conclusion générale, organisés de la manière suivante :

Chapitre 1 : Concepts fondamentaux

Ce chapitre se concentre sur les concepts théoriques fondamentaux indispensables pour comprendre le projet, y compris l'apprentissage profond, les réseaux de neurones, les autoencodeurs et les principes de l'apprentissage adversarial.

Chapitre2 : Revue de littérature (État de l'art)

Ce chapitre examine les études existantes concernant les autoencodeurs traditionnels, les **Autoencodeurs Variationnels (VAE), Autoencodeurs Adversariaux (AAE)**, **Conditional Adversarial Autoencoder (CAAE**) et **CAAE** hybride. Il souligne les bénéfices et les contraintes de chaque méthode, en argumentant la décision d'opter pour les AAE dans le contexte du projet BRUSCO.

Chapitre3 : Approche et élaboration du modèle

Ce chapitre présente la démarche suivie pour mener à bien le projet. Il décrit l'architecture des modèles **AAE**, **CAAE** et **CAAE hybride** (encodeur, décodeur/générateur et discriminateur lorsque applicable), la distribution *Prior* imposée à l'espace latent, le corpus de données **MNIST**, ainsi que les méthodes d'entraînement employées.

Chapitre4 : Mise en œuvre et expérimentations.

Ce chapitre expose la mise en œuvre concrète du modèle suggéré, les décisions techniques prises et les diverses expérimentations réalisées.

Chapitre5 : Analyse et comparaison des résultats

Ce chapitre analyse et compare les résultats des modèles implémentés afin d'identifier celui offrant les meilleures performances en reconstruction, structuration de l'espace latent et génération conditionnelle.

Conclusion générale :

Cette conclusion résume le projet BRUSCO en mettant en évidence l'efficacité du modèle proposé pour la structuration de l'espace latent et la génération contrôlée des images MNIST, tout en soulignant certaines limites liées à la simplicité du jeu de données et à la sensibilité de l'entraînement, et en ouvrant des perspectives pour des travaux futurs.

Chapitre 1 : Concepts fondamentaux

1.1. Apprentissage profond (Deep Learning)

L'**apprentissage profond** (*Deep Learning*) est une sous-discipline de l'apprentissage automatique (*Machine Learning*) qui permet à des **modèles computationnels composés de multiples couches de traitement** d'apprendre des **représentations des données à différents niveaux d'abstraction**. Ces méthodes utilisent des réseaux de neurones profonds pour découvrir des structures complexes dans de grands ensembles de données, ce qui a considérablement amélioré l'état de l'art dans des domaines tels que la reconnaissance vocale, la vision par ordinateur et l'analyse d'images.

1.2. Les Réseau de neurones

Un réseau de neurones est constitué d'unités ou nœuds interconnectés, appelés ***neurones artificiels*** , qui modélisent de manière simplifiée les **neurones** du cerveau. Des modèles de neurones artificiels imitant plus fidèlement les neurones biologiques ont également fait l'objet de recherches récentes et ont démontré une amélioration significative des performances. Ces neurones sont connectés par *des arêtes*, qui modélisent les **synapses** cérébrales. Chaque neurone artificiel reçoit des signaux des neurones connectés, les traite et envoie un signal à d'autres neurones connectés. Ce « signal » est un **nombre réel** , et la sortie de chaque neurone est calculée par une fonction non linéaire de l'ensemble de ses entrées, appelée **fonction *d'activation*** . L'intensité du signal à chaque connexion est déterminée par un *poids*, qui s'ajuste au cours de l'apprentissage.

En général, les neurones sont regroupés en couches. Différentes couches peuvent effectuer différentes transformations sur leurs entrées. Les signaux circulent de la première couche (la *couche d'entrée*) à la dernière couche (la *couche de sortie*), en passant éventuellement par plusieurs couches intermédiaires (***couches cachées***). Un réseau est généralement qualifié de réseau neuronal profond s'il comporte au moins deux couches cachées.

Les réseaux de neurones artificiels sont utilisés pour diverses tâches, notamment **la modélisation prédictive , le contrôle adaptatif** et la résolution de problèmes en **intelligence artificielle**. Ils peuvent apprendre de l'expérience et tirer des conclusions d'un ensemble d'informations complexe et apparemment sans lien entre elles.

1.3. Autoencodeurs (AE)

Un auto-encodeur est un type de réseau de neurones artificiels utilisé pour apprendre des codages efficaces de données non étiquetées, généralement dans le but de réduire leur dimensionnalité. Il emploie un apprentissage non supervisé pour apprendre une représentation (encodage) d'un ensemble de données, généralement afin de réduire leur dimensionnalité. Le réseau est entraîné à compresser l'entrée en un code de dimension inférieure, puis à reconstruire la sortie à partir de cette représentation pour qu'elle corresponde le plus fidèlement possible à l'entrée originale, d'où son nom. Ils comprennent deux sections :

- **Encodeur :** Cette partie du réseau compresse les données d'entrée en une représentation dans un espace latent. Elle encode les données d'entrée sous forme d'une représentation compressée de dimension réduite. La couche d'encodage projette les données d'entrée sur la couche cachée.
- **Décodeur :** Cette partie vise à reconstruire les données d'entrée à partir de leur représentation dans l'espace latent. La couche décodeur associe la couche cachée à la reconstruction des données d'entrée.

Limites des autoencodeurs classiques :

Malgré leur utilité, les auto-encodeurs présentent des limitations et des défis. Ils peuvent parfois apprendre des solutions triviales, comme la fonction identité, qui ne fournissent aucune représentation utile des données. De plus, le choix de la taille de la couche cachée, de la méthode de régularisation et de la fonction de perte peut influencer considérablement les performances et le type de caractéristiques apprises par l'auto-encodeur.

De plus, les auto-encodeurs sont des modèles d'apprentissage non supervisé et, de ce fait, n'utilisent aucune information d'étiquetage. Cela signifie que les caractéristiques apprises peuvent ne pas être optimales pour toute tâche supervisée pour laquelle la représentation encodée est utilisée ultérieurement.

1.4. Variational Autoencoders (VAE)

Un auto-encodeur variationnel est un modèle génératif doté d'une distribution a priori et d'une distribution de bruit. Ces modèles sont généralement entraînés à l'aide du méta-algorithme **EM (par exemple, l'ACP probabiliste** ou le codage parcimonieux (spike & slab)). Ce type

d'approche optimise une borne inférieure de la vraisemblance des données, ce qui est généralement très difficile à calculer et nécessite la découverte de q-distributions, ou distributions **a posteriori** variationnelles. Ces q-distributions sont normalement paramétrées pour chaque point de données individuel lors d'un processus d'optimisation distinct. Cependant, les auto-encodeurs variationnels utilisent un réseau de neurones comme approche amortie pour optimiser conjointement les données sur l'ensemble des points. Ainsi, les mêmes paramètres sont réutilisés pour plusieurs points, ce qui permet de réaliser d'importantes économies de mémoire. Le premier réseau de neurones prend en entrée les points de données eux-mêmes et fournit en sortie les paramètres de la distribution variationnelle. Uneréseau, qui effectue une projection d'un espace d'entrée connu vers un espace latent de faible dimension, est appelé l'encodeur.

Le décodeur est le second réseau de neurones de ce modèle. Il s'agit d'une fonction qui transforme l'espace latent en espace d'entrée, par exemple en moyenne de la distribution du bruit. Il est possible d'utiliser un autre réseau de neurones qui transforme en variance, mais on peut l'omettre par souci de simplicité. Dans ce cas, la variance peut être optimisée par descente de gradient.

Pour optimiser ce modèle, il est nécessaire de connaître deux termes : «l'erreur de reconstruction » et la **divergence de Kullback-Leibler** (KL-D). Ces deux termes sont dérivés de l'expression de l'énergie libre du modèle probabiliste et diffèrent donc selon la distribution du bruit et la distribution a priori des données, ici appelée distribution p. Par exemple, une tâche VAE standard telle qu'IMAGENET est généralement supposée avoir un bruit gaussien ; cependant, des tâches telles que MNIST binarisé nécessitent un bruit de Bernoulli. La KL-D issue de l'expression de l'énergie libre maximise la masse de probabilité de la distribution q qui chevauche la distribution p, ce qui peut malheureusement entraîner un comportement de recherche de mode. Le terme de « reconstruction » correspond au reste de l'expression de l'énergie libre et nécessite une approximation par échantillonnage pour calculer sa valeur moyenne. [1]

1.5. Les Autoencodeurs Adversariaux (AAE)

Les Autoencodeurs Adversariaux (AAE) sont des autoencodeurs probabilistes qui fusionnent la perte de reconstruction classique avec une formation adversariale mise en œuvre dans l'espace latent. Ce dispositif permet d'accorder la distribution postérieure agrégée des codes

latents avec une distribution a priori définie par l'utilisateur, en s'appuyant sur le principe des réseaux antagonistes génératifs (GAN). Cette régularisation permet aux AAE de proposer un apprentissage de représentations structurées et logiques, facilitant la modélisation générative, la classification semi-supervisée, le clustering non dirigé et la diminution de la dimensionalité. Toutefois, leur apprentissage exige un ajustement méticuleux des buts de reconstruction et d'adversaire afin d'assurer la stabilité et la performance du modèle.

1.6. Conditional Adversarial Autoencoder (CAAE)

Le Conditional Adversarial Autoencoder (CAAE) est un modèle de génération qui fusionne les concepts des autoencodeurs adversariaux et des réseaux antagonistes génératifs conditionnels afin d'apprendre une représentation latente régie par des conditions spécifiques telles que des étiquettes ou des attributs. Dans ce processus, l'image initiale est d'abord convertie en un vecteur latent par un encodeur, puis ce vecteur est orienté sur un manifold latent en fonction d'une condition spécifique (comme une classe ou une caractéristique) via un décodeur/générateur. Le modèle emploie deux réseaux antagonistes — l'un pour l'encodeur et l'autre pour le générateur — dans le but d'inciter le générateur à créer des images photoréalistes tout en organisant de manière efficace l'espace latent conditionné.

Ce cadre autorise la production d'images réalistes et maîtrisées en fonction de la condition désirée, sans avoir besoin d'échantillons appariés pour l'apprentissage.

1.7. CAAE hybride

Le modèle proposé est un **Conditional Adversarial Autoencoder hybride (CAAE hybride)**, une architecture générative qui combine les principes des **autoencodeurs variationnels**, des **adversarial autoencoders** et de la **génération conditionnelle**. Son objectif est d'apprendre une représentation latente continue, structurée et discriminante, tout en permettant une génération contrôlée et réaliste des données.

Il repose sur un **encodeur convolutionnel probabiliste** qui projette les images d'entrée dans un espace latent gaussien paramétré par une moyenne et une variance, avec un échantillonnage réalisé via le *reparameterization trick*. La régularisation de cet espace latent est assurée par un **discriminateur adversarial latent**, contraignant la distribution apprise à se rapprocher d'une loi a priori gaussienne. La reconstruction et la génération des images sont effectuées par un **générateur conditionnel**, recevant conjointement le vecteur latent et l'information de classe, tandis qu'un **discriminateur d'images** garantit le réalisme visuel des données produites.

Grâce à l'intégration conjointe du conditionnement par classe, de la régularisation adversariale et de la contrainte variationnelle, le modèle permet d'obtenir des reconstructions de haute qualité, un espace latent bien structuré et une génération conditionnelle stable et contrôlée.

Conclusion

Ce chapitre a présenté les concepts fondamentaux nécessaires à la compréhension des modèles génératifs profonds, en commençant par les bases de l'apprentissage profond et des réseaux de neurones artificiels. Les autoencodeurs classiques et leurs limites ont ensuite été abordés, conduisant naturellement à l'introduction de modèles plus avancés tels que les autoencodeurs variationnels et adversariaux, qui permettent une meilleure structuration de l'espace latent. Enfin, les modèles conditionnels, en particulier le Conditional Adversarial Autoencoder et sa version hybride, ont été introduits comme des solutions performantes pour la génération contrôlée de données.

Chapitre 2 : État de L'arte

2.1. Introduction

Ce chapitre présente une synthèse des travaux existants portant sur les autoencodeurs et leurs principales extensions, en mettant l'accent sur les autoencodeurs classiques (AE), les Autoencodeurs Variationnels (VAE), ainsi que les Autoencodeurs Adversariaux (AAE) et leurs variantes conditionnelles, notamment le **CAAE** et le **CAAE hybride**. Il décrit les architectures, les principes d'apprentissage et les avantages ainsi que les limites de chaque approche. L'objectif est de retracer l'évolution de ces modèles et de justifier le choix des AAE et de leurs extensions comme base du projet **BRUSCO**, en particulier pour la structuration contrôlée de l'espace latent et la génération d'images du jeu de données **MNIST**.

2.2. Comparaison des approches et justification du choix pour BRUSCO

2.2.1. Comparaison des modèles AE, AAE, CAAE, CAAE hybride

Modèle	**Architecture**	**Avantages**	**Inconvénients**
Autoencodeur classique (AE)	Encodeur+ décodeur pour reconstruire les données	Simple, rapide à entraîner, bonne réduction de dimension	Espace latent non structuré, génération de nouvelles données limitée
Variational Autoencoder (VAE)	AE probabiliste : encodeur génère une distribution latente (μ, σ) + rééchantillonnage	Espace latent régularisé, interpolation fluide, génération de nouvelles données	Qualité de génération parfois floue, effondrement postérieur, sensible aux hyperparamètres
Adversarial Autoencoder (AAE)	AE + apprentissage adversarial sur l'espace latent	Contrôle explicite de la distribution latente, interpolation et génération de haute qualité, applications semi-	Entraînement plus complexe, nécessite un réglage précis du discriminateur et de l'encodeur

		supervisées et non supervisées	
Conditional Adversarial Autoencoder (CAAE)	AAE conditionné sur des labels ou caractéristiques	Génération dirigée, contrôle latent conditionnel, interpolation et génération de haute qualité	Entraînement complexe, nécessite une information conditionnelle fiable
Conditional Adversarial Autoencoder hybride (CAAE hybride)	Extension de l'AAE intégrant une information conditionnelle (classe) dans l'encodeur et le discriminateur, tout en conservant la régularisation adversariale.	- Meilleure séparation inter-classes - Génération conditionnelle et contrôle - Espace latent plus discriminant	Complexité accrue du modèle - Dépendance à la qualité des labels - Séparation parfois imparfaite entre classes proches

2.2.2. Justification du choix des AAE et CAAE et CAAE hybride pour le projet BRUSCO

Le choix des modèles AAE, CAAE et CAAE hybride dans le cadre du projet BRUSCO s'explique par leur capacité complémentaire à apprendre des représentations latentes structurées et exploitables pour des données visuelles. L'AAE constitue une base pertinente pour imposer une distribution contrôlée dans l'espace latent grâce à l'apprentissage adversarial, facilitant ainsi l'analyse et la génération des données. Le CAAE enrichit ce cadre en introduisant le conditionnement par classe, indispensable pour assurer une génération contrôlée et une meilleure séparabilité des représentations, notamment dans un contexte de reconnaissance et de génération de symboles. Enfin, le CAAE hybride a été retenu afin de dépasser les limites des deux premiers modèles, en combinant le conditionnement, la régularisation adversariale renforcée et des éléments variationnels, permettant d'obtenir un espace latent plus discriminant, une reconstruction plus fidèle et une génération conditionnelle

plus stable. Cette progression logique des modèles s'inscrit pleinement dans les objectifs du projet BRUSCO, qui vise à structurer l'espace latent tout en conservant un contrôle précis sur les représentations apprises.

Par conséquent, l'emploi conjoint de AAE, et CAAE hybride dans le cadre du projet BRUSCO aboutit à un modèle qui peut produire des images MNIST de haute qualité, tout en préservant un espace latent structuré, maîtrisable et conditionnel, conformément aux objectifs fondamentaux du projet.

Conclusion

Ce chapitre a proposé une comparaison des principaux autoencodeurs et modèles génératifs (AE, VAE, AAE, CAAE et CAAE hybride). L'étude a montré que les modèles adversariaux et conditionnels surpassent les approches classiques en termes de structuration de l'espace latent et de contrôle de la génération. En particulier, le CAAE hybride se distingue par une meilleure qualité de reconstruction et une séparation latente plus marquée. Ces observations motivent son choix pour le projet BRUSCO, garantissant un bon équilibre entre fidélité visuelle, contrôle latent et génération conditionnelle.

Chapitre 3 : Approche et élaboration du modèle

3.1. Introduction

Ce chapitre fournit une description approfondie de l'architecture retenue, du jeu de données exploité et de la tactique de formation du modèle Adversarial Autoencoder (AAE) élaboré dans le contexte du projet BRUSCO. La finalité est d'exposer de façon précise et structurée la méthode pour produire et examiner les images de signes ASL tout en maîtrisant la répartition de l'espace latent.

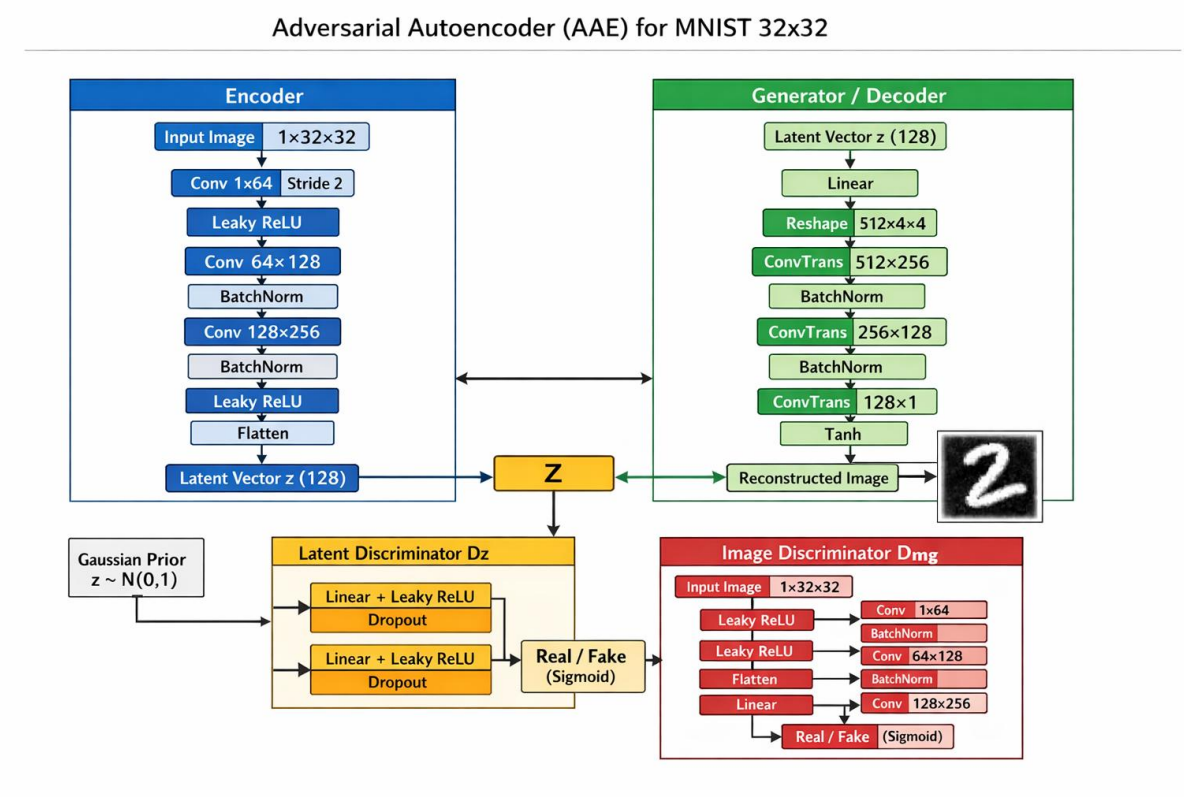
3.2. Architecture du modèle AAE

La structure d'un Autoencodeur Adversarial (AAE) peut être ajustée en fonction de la complexité des données et des buts de modélisation. Toutefois, la majorité des AAE s'appuient sur trois éléments fondamentaux, comme indiqué dans le schéma ci-dessus:

L'encodeur : qui transforme les données d'entrée en un espace latent compressé

Le décodeur : qui refait surface les données originales à partir du vecteur latent

Et **le discriminateur** : qui contrôle la distribution de l'espace latent en différenciant les codes générés des échantillons issus d'une distribution prior.



3.2.1. Explication de l'architecture de l'Adversarial Autoencoder (AAE)

Le diagramme illustre la structure traditionnelle d'un Adversarial Autoencoder (AAE), intégrant un autoencodeur avec un apprentissage antagoniste afin de réguler l'espace latent.

- **Encodeur (Encoder)**

L'encodeur a pour rôle de projeter une image d'entrée $x \in \mathbb{R}^{1\times32\times32}$ vers un vecteur latent continu $z \in \mathbb{R}^{128}$. Il est constitué de trois couches convolutionnelles successives qui permettent une réduction progressive de la résolution spatiale de l'image (de 32×32 à 16×16, puis 8×8 et enfin 4×4), tout en augmentant simultanément le nombre de canaux (64, 128 et 256).

Chaque couche utilise l'activation **LeakyReLU** afin d'éviter le problème des neurones morts, et intègre une **Batch Normalization** pour stabiliser l'apprentissage et accélérer la convergence. La sortie convolutionnelle de dimension $4 \times 4 \times 256$ est ensuite aplatie et projetée, à l'aide d'une couche entièrement connectée, vers le vecteur latent z. Ce dernier constitue une représentation compacte et informative des images d'entrée, conçue pour être régularisée adversarialement afin de suivre une distribution gaussienne imposée

- **Decoder / Generator**

Le décodeur, également appelé générateur dans l'architecture Adversarial Autoencoder (AAE), a pour objectif de reconstruire l'image d'entrée à partir de la représentation latente z produite par l'encodeur. Il implémente une fonction déterministe $G\theta: Z \rightarrow X$, où $z \subset R^{128}$ représente l'espace latent régularisé par discrimination adversariale et $X \subset R^{32\times32\times1}$ correspond à l'espace des images reconstruites. Le vecteur latent Z est d'abord projeté à travers une couche entièrement connectée afin de produire une représentation de dimension $4\times4\times8n$. Cette sortie est ensuite restructurée spatialement par une opération de *reshape* pour former un tenseur tridimensionnel. Une seconde couche dense suivie d'une fonction d'activation sigmoïde est ensuite appliquée afin de générer les valeurs de pixels de l'image reconstruite. Enfin, une fonction d'activation de type *tanh* est utilisée en sortie pour contraindre les intensités des pixels dans l'intervalle $[-1,1]$, assurant ainsi une compatibilité avec le prétraitement des données d'entrée. Ce décodeur joue un rôle central dans l'apprentissage de représentations latentes informatives et cohérentes, tout en permettant la génération d'images réalistes, lesquelles sont ensuite évaluées par le discriminateur d'images D_{img}.

- **Discriminateur latent Dz**

Le discriminateur latent $D_z$opère directement dans l'espace latent et non sur les images reconstruites. Sa fonction principale est de distinguer les vecteurs latents $z$générés par l'encodeur de ceux, notés z_{prior}, échantillonnés à partir d'une distribution gaussienne standard. Sur le plan architectural, $D_z$est implémenté sous la forme d'un réseau de neurones entièrement connecté de type MLP, utilisant des fonctions d'activation LeakyReLU afin d'assurer une bonne propagation du gradient. Des couches de Dropout sont intégrées pour limiter le surapprentissage et améliorer la généralisation du modèle. La couche de sortie emploie une activation sigmoïde, fournissant une probabilité indiquant si un vecteur latent provient de la distribution prior ou de l'encodeur. Le rôle du discriminateur latent est ainsi de contraindre l'encodeur à produire des représentations latentes indiscernables d'une loi gaussienne, garantissant un espace latent continu, régulier et propice à l'interpolation, condition essentielle pour une génération contrôlée et cohérente des données.

- **Discriminateur d'images Dimg**

Le discriminateur d'images constitue un élément central qui distingue ce modèle d'un **Adversarial Autoencoder (AAE) classique**, en introduisant une contrainte adversariale directement dans l'espace image. Sa fonction principale est de différencier les images réelles issues du jeu de données des images reconstruites générées par le décodeur. Sur le plan architectural, ce discriminateur est composé de trois couches convolutionnelles descendantes permettant une réduction progressive de la résolution spatiale de 32×32 à 16×16, puis 8×8 et enfin 4×4, tout en augmentant simultanément le nombre de canaux. Chaque couche intègre une **Batch Normalization** et une activation **LeakyReLU**, assurant une convergence stable et efficace de l'apprentissage. La sortie convolutionnelle est ensuite aplatie et transmise à une couche entièrement connectée finale utilisant une activation **sigmoïde**, produisant une probabilité indiquant si l'image est réelle ou générée. Le rôle de ce discriminateur est d'imposer une contrainte adversariale dans l'espace image afin d'améliorer la netteté, la cohérence et le réalisme visuel des images reconstruites. Il complète ainsi la perte de reconstruction de type L_1en limitant les reconstructions excessivement lisses, souvent observées lorsque seule une perte pixel à pixel est utilisée.

3.2.2. Fonctions de perte

L'apprentissage du modèle **AAE avec discriminateur d'images (AAE + D_{img})** repose sur une optimisation multi-objectifs combinant plusieurs fonctions de coût complémentaires. Il intègre d'une part une **perte de reconstruction**, propre au cadre des autoencodeurs, visant à assurer la fidélité entre les images d'entrée et leurs reconstructions. D'autre part, une **perte adversariale dans l'espace latent** est introduite afin de contraindre les représentations latentes produites par l'encodeur à suivre une distribution *a priori* donnée, généralement gaussienne. Enfin, une **perte adversariale dans l'espace image** est ajoutée pour encourager la production d'images visuellement réalistes, indiscernables des images réelles du jeu de données. Chaque composant du modèle — à savoir l'encodeur, le générateur et les différents discriminateurs — est optimisé à l'aide d'une fonction de coût spécifique, permettant d'assurer simultanément la qualité des reconstructions, la régularisation probabiliste de l'espace latent et le réalisme visuel des images générées.

3.2.2.1.Perte de reconstruction (L1 Loss)

La **perte de reconstruction** permet de mesurer l'écart pixel par pixel entre l'image d'entrée $x$et son image reconstruite $\hat{x}$produite par le modèle. Dans ce travail, la **norme** L_1est privilégiée par rapport à la norme L_2(ou erreur quadratique moyenne), car elle est moins sensible aux

$$\mathcal{L}_{rec} = \|x - \hat{x}\|_1 = \mathbb{E}_{x \sim p_{data}} \left[\sum_i |x_i - \hat{x}_i| \right]$$

valeurs aberrantes et favorise des reconstructions plus nettes et plus précises. En limitant la pénalisation excessive des grandes erreurs locales, la perte L_1 contribue à réduire le phénomène de flou fréquemment observé lors de l'utilisation de la MSE, notamment dans les tâches de génération d'images. Mathématiquement, la perte de reconstruction est définie par :

Cette fonction de coût joue un rôle fondamental dans le modèle, car elle garantit la fidélité structurelle entre l'image originale et sa reconstruction, tout en constituant la base du comportement autoencodeur sur laquelle viennent se greffer les contraintes adversariales imposées lors de l'apprentissage.

3.2.2.2 Perte adversariale sur l'espace latent (Encoder vs Dz)

Le principe de la régularisation adversariale dans l'espace latent repose sur l'interaction entre l'encodeur E et le discriminateur latent D_z. Ce dernier a pour objectif de distinguer les vecteurs latents $z = E(x)$ produits par l'encodeur à partir des données réelles, des vecteurs z_{prior} échantillonnés depuis une distribution *a priori* définie, généralement une loi gaussienne standard $\mathcal{N}(0, I)$. Le discriminateur D_z est ainsi entraîné à attribuer une probabilité élevée aux vecteurs provenant de la distribution prior et une probabilité faible à ceux générés par l'encodeur. À l'inverse, l'encodeur est optimisé de manière adversariale afin de tromper le discriminateur, en produisant des représentations latentes dont la distribution est indiscernable de la distribution gaussienne imposée. Cette dynamique adversariale permet d'aligner progressivement la distribution des codes latents encodés sur le prior souhaité, assurant ainsi un espace latent régulier, continu et propice à l'interpolation.

3.2.2.2.1 Perte du discriminateur latent Dz

Le discriminateur latent D_z joue un rôle central dans la régularisation probabiliste de l'espace latent. Son objectif est de distinguer correctement les vecteurs latents réels z_{prior}, échantillonnés à partir de la distribution *a priori* imposée, généralement une loi gaussienne standard $\mathcal{N}(0, I)$, des vecteurs latents générés par l'encodeur $z = E(x)$. En apprenant à effectuer cette discrimination, D_z permet d'imposer une contrainte probabiliste stable sur l'espace latent, garantissant que les représentations apprises suivent la distribution souhaitée.

La fonction de perte du discriminateur latent est définie à partir d'une **entropie croisée binaire**, et s'écrit comme suit :

$$\mathcal{L}_{D_z} = \mathbb{E}_{z_{prior}}[\log D_z(z_{prior})] + \mathbb{E}_{x \sim p_{data}}[\log(1 - D_z(E(x)))]$$

Cette fonction de coût encourage le discriminateur à attribuer une probabilité élevée aux vecteurs latents provenant de la distribution gaussienne et une probabilité faible à ceux issus de

l'encodeur. En optimisant cette perte, $D_z$renforce la séparation entre les *vrais* et les *faux* latents, ce qui stabilise le processus d'apprentissage adversarial et améliore la régularité et la continuité de l'espace latent.

3.2.2.2.2 Perte adversariale de l'encodeur (E vs Dz)

Dans le cadre de l'Autoencodeur Adversarial, l'encodeur Eest engagé dans un jeu adversarial avec le discriminateur latent D_z. Alors que $D_z$cherche à distinguer les vecteurs latents issus de la distribution *a priori* $z_{prior} \sim \mathcal{N}(0, I)$de ceux générés par l'encodeur $z = E(x)$, l'objectif de l'encodeur est inverse : il doit produire des représentations latentes indiscernables de celles provenant de la loi gaussienne imposée.

La perte adversariale de l'encodeur est définie de manière à maximiser la probabilité que les vecteurs latents encodés soient classés comme *réels* par le discriminateur latent. Elle s'exprime par la fonction de coût suivante :

$$\mathcal{L}_{E_z} = \mathbb{E}_{x \sim p_{data}}[\log D_z(E(x))]$$

En minimisant cette fonction de perte, l'encodeur apprend à « tromper » le discriminateur latent en ajustant progressivement la distribution des vecteurs $z = E(x)$pour qu'elle se rapproche de la distribution gaussienne cible $\mathcal{N}(0, I)$. Ce mécanisme de régularisation adversariale permet de garantir un espace latent continu, régulier et bien structuré, facilitant ainsi l'interpolation entre les points latents et la génération contrôlée de nouvelles données.

Ainsi, l'optimisation conjointe de l'encodeur et du discriminateur latent conduit à un alignement efficace entre la distribution latente apprise et la distribution *a priori*, condition essentielle pour la stabilité et la qualité du modèle génératif.

3.2.2.3. Perte adversariale dans l'espace image (Generator vs Dimg)

Le discriminateur d'images D_{img}opère directement dans l'espace image et introduit une contrainte adversariale de type GAN afin d'améliorer la qualité visuelle des reconstructions produites par le générateur. Son rôle principal consiste à distinguer les images réelles x, issues de la distribution des données $p_{data}(x)$, des images reconstruites $\hat{x} = G\big(E(x)\big)$, générées à partir des représentations latentes encodées.

3.2.2.3.1. Perte du discriminateur d'images Dimg

La perte du discriminateur d'images D_{img} pour objectif d'entraîner ce dernier à distinguer correctement les images réelles provenant du jeu de données et les images reconstruites générées par le modèle. Le discriminateur reçoit en entrée, d'une part, les images réelles $x \sim p_{data}(x)$et, d'autre part, les images reconstruites $\hat{x} = G\big(E(x)\big)$, issues du générateur. Son apprentissage repose sur un cadre adversarial inspiré des réseaux antagonistes génératifs (GAN), où D_{img}cherche à maximiser la probabilité d'assigner la bonne étiquette à chaque type d'image. Cette contrainte est formalisée par la fonction de perte suivante :

$$\mathcal{L}_{D_{img}} = \mathbb{E}_{x\sim p_{data}}[\log D_{img}(x)] + \mathbb{E}_{x\sim p_{data}}[\log(1 - D_{img}(\hat{x}))]$$

3.2.2.3.2. Perte adversariale du générateur (G vs Dimg)

La perte adversariale du générateur a pour objectif d'inciter ce dernier à produire des images reconstruites visuellement réalistes, capables de tromper le discriminateur d'images D_{img}. Contrairement à la perte de reconstruction L_1, qui se concentre principalement sur la fidélité pixel-à-pixel entre l'image originale et sa reconstruction, la contrainte adversariale agit à un niveau perceptuel plus global. Elle pénalise les reconstructions présentant des artefacts ou une apparence non naturelle, contribuant ainsi à améliorer la netteté et la cohérence visuelle des images générées. Cette fonction de coût est définie comme suit :

$$\mathcal{L}_{G_{img}} = \mathbb{E}_{x\sim p_{data}}[\log D_{img}(\hat{x})]$$

En minimisant cette perte, le générateur apprend à maximiser la probabilité que les images reconstruites soient classées comme réelles par le discriminateur. Ainsi, la perte adversariale du générateur complète efficacement la perte L_1, en forçant le modèle à produire des images à la fois fidèles à la structure originale et perceptuellement réalistes. Cette combinaison permet d'éviter les reconstructions excessivement lisses et favorise une meilleure qualité visuelle globale.

3.2.2.4. Fonction de coût globale (Encoder + Generator)

L'encodeur E et le générateur Gsont optimisés conjointement à travers une fonction de coût globale qui combine de manière pondérée les différentes composantes de perte intervenant dans l'apprentissage du modèle. Cette formulation multi-objectifs permet d'assurer simultanément

$$\mathcal{L}_{E,G} = \mathcal{L}_{rec} + \lambda_z \mathcal{L}_E^{adv} + \lambda_{img} \mathcal{L}_G^{adv}$$

la fidélité des reconstructions, la régularisation probabiliste de l'espace latent et le réalisme visuel des images générées. La fonction de coût globale est définie comme suit :

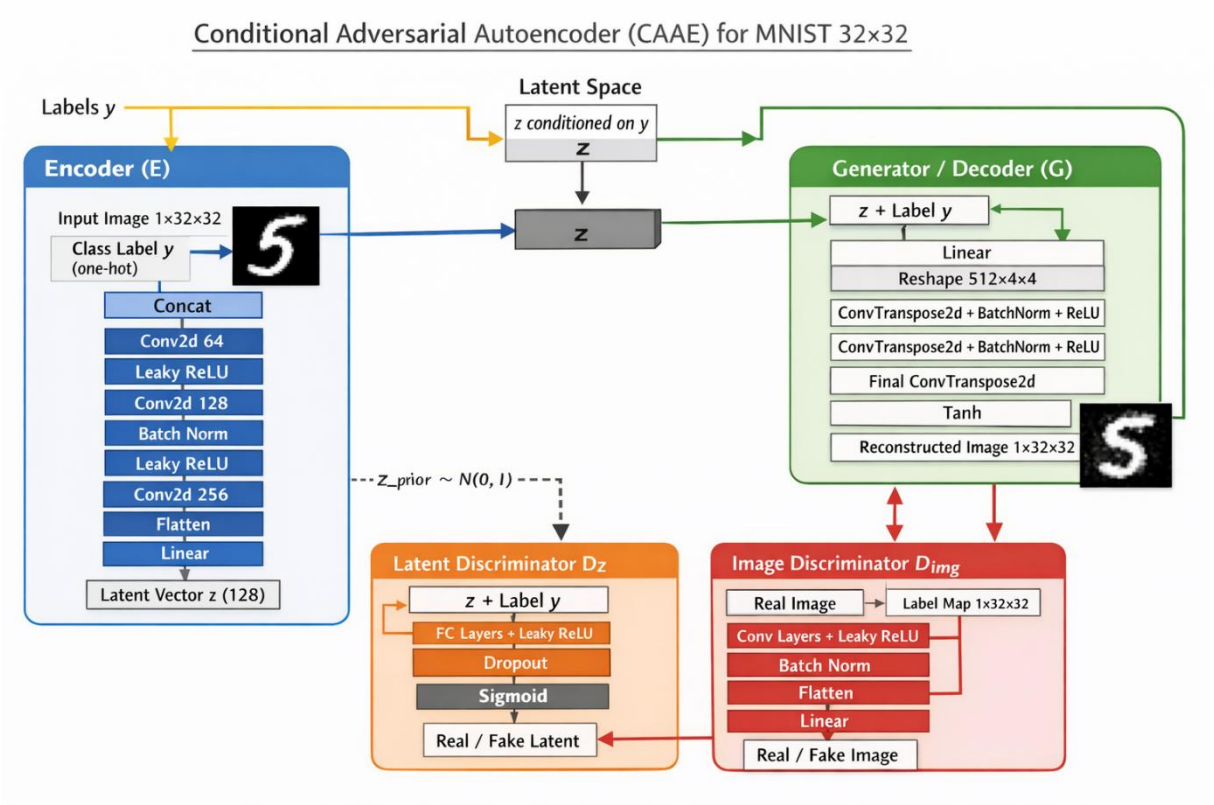
- $\mathcal{L}_{rec}$ Représente la perte de reconstruction (norme L_1),
- $\mathcal{L}_E^{adv}$ Correspond à la perte adversariale de l'encodeur face au discriminateur latent D_z,
- $\mathcal{L}_G^{adv}$ Désigne la perte adversariale du générateur face au discriminateur d'images D_{img},
- λ_z et λ_{img} sont des coefficients de pondération contrôlant l'influence relative des contraintes adversariales.

3.3. Architecture du modèle CAAE

La structure d'un Conditional Adversarial Autoencoder (CAAE) peut être ajustée en fonction de la complexité des données et des objectifs de modélisation. Toutefois, la majorité des CAAE reposent sur trois éléments fondamentaux, comme illustré dans le schéma ci-dessus **:**

L'encodeur conditionnel, qui convertit les données d'entrée, en même temps que l'information conditionnelle liée, en un espace latent comprimé ,

Le décodeur conditionnel, qui a pour tâche de restituer les données initiales à partir du vecteur latent et de la condition spécifiée ; ainsi que le discriminateur conditionnel, qui vise à régulariser la distribution de l'espace latent en distinguant les codes produits par l'encodeur des échantillons provenant d'une distribution préalable, tout en considérant la condition liée.



3.3.1. Explication de l'architecture de l'Adversarial Autoencoder (CAAE)

Le schéma démontre la configuration d'un Autoencodeur Adversarial Conditionnel (CAAE), représentant une prolongation de l'Autoencodeur Adversarial traditionnel grâce à l'intégration d'un apprentissage conditionnel. Cette structure combine un autoencodeur avec un mécanisme d'adversariale pour réguler l'espace latent, tout en incorporant une information conditionnelle qui permet de maîtriser la reconstruction et la production des données.

- Encodeur conditionnel $\boldsymbol{E(x, y)}$

L'encodeur a pour rôle de projeter une image d'entrée $x \in \mathbb{R}^{1\times32\times32}$, associée à son label y(encodé en one-hot), vers un vecteur latent continu $z \in \mathbb{R}^{128}$. Il est constitué de plusieurs couches convolutionnelles successives intégrant des activations **LeakyReLU** et des **Batch Normalization**, permettant une réduction progressive de la résolution spatiale (32 → 16 → 8 → 4) tout en augmentant le nombre de canaux.

Après l'aplatissement des cartes de caractéristiques, une couche fully connected produit le vecteur latent z, qui est ensuite concaténé avec le label y. Cette concaténation permet d'injecter

explicitement l'information de classe dans l'espace latent, favorisant ainsi une structuration conditionnelle des représentations apprises.

- Décodeur conditionnel $\boldsymbol{G(z, y)}$

Le générateur reçoit en entrée le vecteur latent $z$concaténé au label y, et a pour mission de reconstruire l'image d'origine. La génération s'effectue en deux étapes : une couche fully connected transforme le vecteur latent conditionné en un tenseur de départ, puis plusieurs couches de convolutions transposées (ConvTranspose2D) augmentent progressivement la résolution spatiale ($4 \rightarrow 8 \rightarrow 16 \rightarrow 32$).

Chaque étape est suivie d'une activation LeakyRelu et d'une Batch Normalization, tandis que la couche finale utilise une activation Tanh afin de produire des pixels normalisés dans l'intervalle $[-1,1]$. Ce décodeur conditionnel permet ainsi une génération d'images cohérentes avec la classe imposée.

- Discriminateur latent conditionnel $\boldsymbol{D_z(z, y)}$

Le discriminateur latent agit dans l'espace latent et reçoit en entrée le couple (z, y). Son objectif est de distinguer les vecteurs latents produits par l'encodeur de ceux échantillonnés depuis une distribution prior conditionnelle, généralement une loi gaussienne :

$$z_{prior} \sim \mathcal{N}(0, I)$$

Le réseau est implémenté sous forme d'un perceptron multicouche (MLP) avec des activations LeakyReLU et une sortie sigmoïde. Le rôle de $D_z$est de contraindre l'encodeur à produire des vecteurs latents dont la distribution conditionnelle $P(z \mid y)$est indiscernable du prior imposé.

- Discriminateur d'images conditionnel $D_{img}(x, y)$

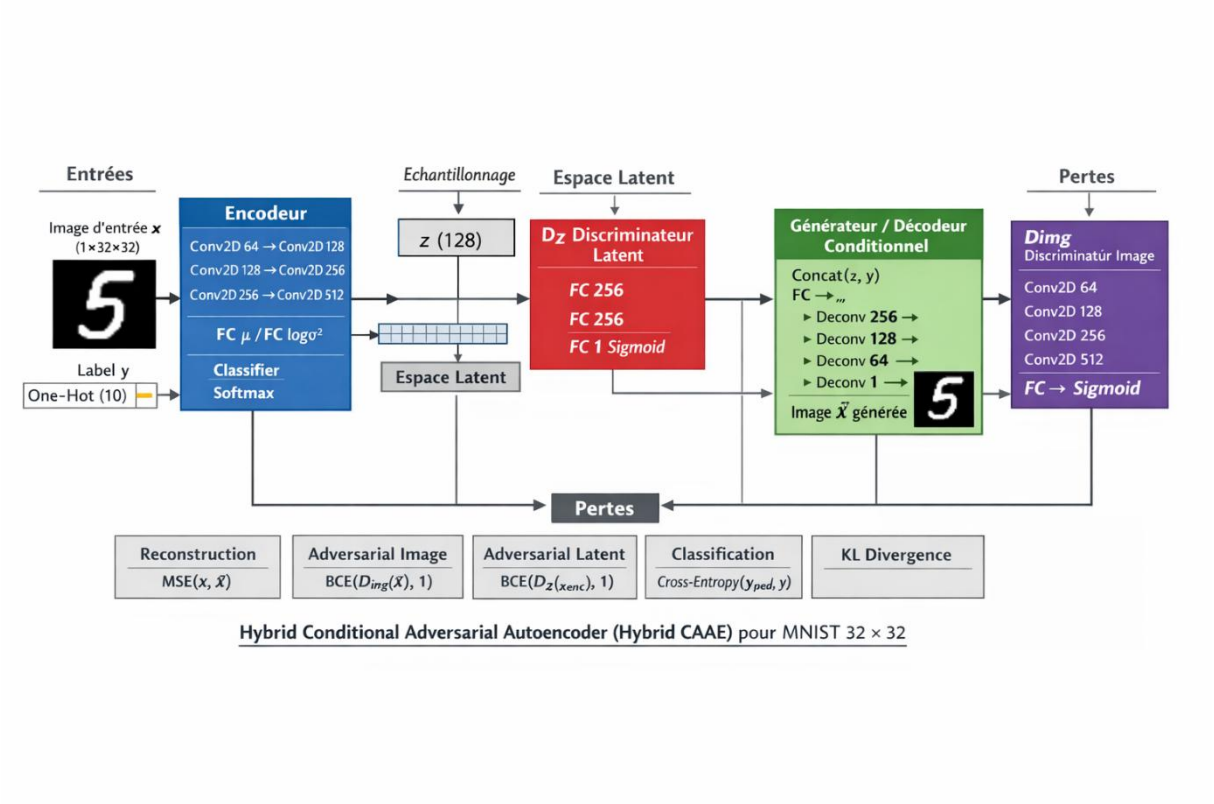
Le discriminateur d'images opère directement dans l'espace image et distingue les images réelles du jeu de données MNIST des images reconstruites par le générateur, en tenant compte du label y. Il est composé de couches convolutionnelles descendantes suivies d'une couche fully connected avec une sortie sigmoïde.

Cette composante introduit un apprentissage de type GAN dans l'espace image, renforçant le réalisme visuel des reconstructions et complétant la perte de reconstruction pixel-à-pixel.

3.3.2. Fonctions de perte

Voir les fonctions de pertes AAE, sont les mêmes fonctions utilisent dans AAE

3.4. Architecture du modèle CAAE hybride



Hybrid Conditional Adversarial Autoencoder (Hybrid CAAE) pour MNIST 32 × 32

La structure d'un **Conditional Adversarial Autoencoder hybride (CAAE hybride)** peut être adaptée en fonction de la complexité des données et des objectifs de modélisation. Néanmoins, ce type d'architecture repose généralement sur **plusieurs composants fondamentaux**, comme illustré dans le schéma ci-dessus.

Elle comprend tout d'abord **un encodeur conditionnel**, chargé de transformer les données d'entrée, conjointement avec l'information de classe associée, en un espace latent compact et discriminant. Ensuite, **un décodeur (ou générateur) conditionnel** permet de reconstruire ou de générer les données à partir du vecteur latent et de la condition spécifiée.

3.4.1. Explication de L'architecture de CAAE hybride

- **Encodeur convolutionnel (Encoder)**

L'encodeur reçoit en entrée une image en niveaux de gris de taille 32×32. Il est composé de plusieurs couches convolutionnelles successives permettant d'extraire progressivement des caractéristiques visuelles de haut niveau. À l'issue de cette extraction, les représentations sont projetées vers un espace latent probabiliste.

Deux vecteurs sont appris à la sortie de l'encodeur : la moyenne μ et la variance σ^2, définissant une distribution gaussienne latente. Le vecteur latent z est ensuite obtenu par échantillonnage stochastique à l'aide du reparameterization trick :

$$z = \mu + \epsilon \cdot \sigma, \epsilon \sim \mathcal{N}(0, I)$$

En parallèle, l'encodeur intègre une tête de classification appliquée directement au vecteur latent z, permettant de prédire la classe associée à l'image d'entrée. Cette prédiction est optimisée via une fonction de perte de type *Cross-Entropy*, favorisant une meilleure séparation inter-classes dans l'espace latent.

- **Régularisation adversariale de l'espace latent (Dz)**

Le vecteur latent $z$produit par l'encodeur est ensuite transmis à un discriminateur latent D_z. Ce dernier a pour objectif de distinguer les vecteurs latents générés par l'encodeur de ceux échantillonnés à partir d'une distribution gaussienne cible $\mathcal{N}(0, I)$.

L'encodeur est entraîné de manière adversariale afin de tromper le discriminateur D_z, de sorte que la distribution des vecteurs latents appris soit indiscernable de la distribution gaussienne cible. Cette interaction est régie par une perte adversariale de type *Binary Cross-Entropy* :

$$\mathcal{L}_{Dz} = \mathrm{BCE}(D_z(z_{\mathrm{prior}}), 1) + \mathrm{BCE}(D_z(z_{\mathrm{enc}}), 0)$$

Cette régularisation garantit une structure continue et régulière de l'espace latent, facilitant l'interpolation et la génération contrôlée.

- **Générateur conditionnel (Generator)**

Le générateur reçoit en entrée le vecteur latent $z$ainsi que l'étiquette de classe y, encodée sous forme *one-hot*. Ces deux vecteurs sont concaténés afin d'introduire explicitement l'information conditionnelle :

$$h = [z; y]$$

Le vecteur résultant est ensuite transformé par des couches entièrement connectées suivies de couches de déconvolution (*ConvTranspose2d*), permettant de produire une image reconstruite ou générée de taille 32×32.

Le conditionnement par classe permet de contrôler le contenu sémantique des images générées, constituant ainsi une différence fondamentale entre un AAE classique et un CAAE.

Discriminateur d'image (Dimg)

Le discriminateur d'image $Dimg$ reçoit soit des images réelles issues du jeu de données, soit des images synthétiques produites par le générateur. Son rôle est de distinguer les images réelles des images générées à l'aide d'une fonction de perte adversariale.

La perte associée au discriminateur d'image est définie comme suit :

$$\mathcal{L}_{Dimg} = \mathrm{BCE}(D(x_{real}), 1) + \mathrm{BCE}\big(D\big(x_{fake}\big), 0\big)$$

Le générateur est entraîné pour tromper ce discriminateur, ce qui améliore progressivement la qualité visuelle et le réalisme des images produites.

3.4.2. Fonctions de perte

L'apprentissage du modèle repose sur une combinaison pondérée de plusieurs fonctions de perte complémentaires.

3.4.2.1.Perte de reconstruction

Mesurée par l'erreur quadratique moyenne (*Mean Squared Error*), garantit la fidélité entre l'image d'entrée et l'image reconstruite :

$$Lrecon = MSE(x, \hat{x})$$

3.4.2.2.Perte de classification

Est basée sur la *Cross-Entropy*, renforce la séparation des classes dans l'espace latent :

$$\mathcal{L}_{\text{cls}} = \text{CE}(\hat{y}, y)$$

3.4.2.3.Perte variationnelle KL

Est inspirée des Variational Autoencoders, favorise la continuité et la stabilité de l'espace latent :

$$\mathcal{L}_{KL} = -\frac{1}{2}\,\mathbb{E}[1 + \log(\sigma^2) - \mu^2 - \sigma^2]$$

Enfin, les **pertes adversariales** associées à $D_z$et $Dimg$ imposent respectivement une distribution gaussienne sur le latent space et un réalisme visuel sur les images générées.

3.4.2.4.Fonction de perte total

$$\mathcal{L}_{total} = \underbrace{\mathcal{L}_{E\_adv}}_{\text{encoder trompe D}} + \underbrace{\mathcal{L}_{G\_adv}}_{\text{generator trompe D}} + \lambda_{recon}\mathcal{L}_{recon} + \lambda_{cls}\mathcal{L}_{cls} + \lambda_{KL}\mathcal{L}_{KL} + \underbrace{\mathcal{L}_{Ez}}_{\text{encoder trompe Dz}}$$

Conclusion

Ce chapitre a présenté les architectures AAE, CAAE et CAAE hybride, illustrant une évolution progressive vers des modèles génératifs mieux structurés et plus contrôlables. L'AAE permet d'imposer une distribution probabiliste à l'espace latent, tandis que le CAAE améliore la séparation inter-classes grâce à l'intégration de l'information conditionnelle. Le CAAE hybride combine ces avantages avec une contrainte adversariale dans l'espace image, renforçant le réalisme des reconstructions. Ces modèles constituent la base des expérimentations et analyses présentées par la suite.

Chapitre 4 : Mise en œuvre et expérimentations

4.1. Introduction

Ce chapitre expose la mise en pratique des modèles Adversarial Autoencoder (AAE) et Conditional Adversarial Autoencoder (CAAE) élaborés dans le contexte du projet BRUSCO, ainsi que l'étude des résultats expérimentaux obtenus. Suite à l'analyse théorique de ces architectures, nous allons décrire l'environnement de développement, le jeu de données utilisé, les décisions architecturales prises ainsi que la stratégie d'apprentissage mise en œuvre. Les tests ont été effectués sur le jeu de données MNIST, qui sert de référence pour vérifier la capacité des modèles à organiser efficacement l'espace latent et à produire des reconstructions et des créations cohérentes. L'évaluation se base essentiellement sur la qualité de la reconstruction, la représentation de l'espace latent et les performances en matière de génération. Cela représente une étape préliminaire avant d'envisager une extension vers des données de gestes réels.

4.2. Environnement de développement

L'implémentation des modèles **Adversarial Autoencoder (AAE)**, **Conditional Adversarial Autoencoder (CAAE)** et **CAAE hybride**, conçus dans le contexte du projet **BRUSCO**, a été réalisée à l'aide du langage de programmation **Python**. Le framework **PyTorch** a été retenu en raison de sa souplesse, de sa modularité et de son efficacité pour la mise en œuvre de modèles d'apprentissage profond, en particulier ceux reposant sur des mécanismes adversariaux.

Les expérimentations ont été menées principalement sur la plateforme **Google Colab**, qui offre un environnement clé en main et permet, lorsque cela est disponible, l'exploitation d'unités de traitement graphique (**GPU**). L'utilisation du GPU a permis d'accélérer significativement les phases d'entraînement et de faciliter les itérations expérimentales. En complément, certaines phases de développement et d'expérimentation ont également été réalisées sur la plateforme **Kaggle**, en exploitant son environnement de calcul avec **accélération GPU**, ce qui a contribué à améliorer les performances d'entraînement et la reproductibilité des résultats.

Les données utilisées dans ce projet ont été collectées à partir de la plateforme **Kaggle**, puis intégrées et manipulées directement dans les environnements Colab et Kaggle.

L’implémentation des réseaux de neurones et la gestion des ensembles de données ont été effectuées à l’aide des bibliothèques **torch** et **torchvision**. La bibliothèque **NumPy** a été utilisée pour les calculs numériques, tandis que **Matplotlib** a permis la visualisation des images reconstruites, des échantillons générés et des résultats expérimentaux.

Par ailleurs, la bibliothèque **scikit-learn** a été employée pour l’analyse et la visualisation de l’espace latent, notamment à travers des techniques de réduction de dimension telles que **PCA** et **t-SNE**. Enfin, la bibliothèque **tqdm** a été utilisée afin de suivre l’avancement de l’entraînement de manière claire et intuitive.

4.3. Données utilisées pour l'entraînement.

4.3.1. Description du jeu de données MNIST.

Dans cette étape d'expérimentation, nous avons sélectionné le jeu de données MNIST comme référence. MNIST est couramment employé au sein de la communauté scientifique pour juger les modèles d'apprentissage profond, particulièrement dans le domaine de la vision informatique. Ce dernier comprend 60 000 images destinées à l'entraînement et 10 000 images dédiées aux tests, qui illustrent des chiffres manuscrits de 0 à 9. Chaque image, qui est en noir et blanc et mesure 28×28 pixels, est liée à une étiquette représentant l'une des dix catégories.
Même si le projet BRUSCO était au départ centré sur l'étude des mouvements de la main, comme ceux provenant de bases de données ASL, l'emploi de MNIST est justifié par sa simplicité, sa fiabilité et son aptitude à fonctionner comme un banc d'essai robuste. Cette décision permet de confirmer les architectures AAE et cAAE, de simplifier la mise en parallèle avec des recherches antérieures et d'examiner de façon contrôlée le comportement de l'espace latent avant une généralisation vers des données plus sophistiquées.

4.3.2 Préparation des données

Avant d'être utilisées pour l'entraînement des modèles, les images MNIST ont passé par diverses phases de prétraitement. Les images ont initialement été mises à l'échelle de 64×64 pixels pour correspondre à l'architecture convolutionnelle employée. Elles ont par la suite été transformées spécifiquement en un unique canal correspondant aux niveaux de gris. Les valeurs de pixels ont été standardisées dans l’intervalle [-1, 1], ce qui aide à renforcer la stabilité de l’apprentissage, spécialement lors d’un entraînement adversaire. Pour finir, le dataset a été scindé en deux parties :
80 % pour l'apprentissage et 20 % pour l'évaluation, assurant de cette manière une analyse impartiale des performances des modèles.

4.4. Implémentation des modèles AAE, CAAE et CAAE hybride

4.4.1. Implémentation de modèles AAE

Cette partie explique la mise en œuvre du modèle Adversarial Autoencoder (AAE) qui est employée dans le projet BRUSCO. Ce modèle vise à obtenir une représentation latente concise et organisée des images, tout en appliquant une contrainte probabiliste sur la distribution de l'espace latent grâce à un apprentissage antagoniste.
Le modèle de l'AAE en place est basé sur trois réseaux neuronaux séparés :
Un encodeur, un générateur (ou décodeur) et un discriminateur latent, qui sont formés ensemble en suivant une approche adversariale.

- **Architecture de l'encodeur**

L'encodeur est un réseau de neurones convolutifs approfondi conçu pour représenter l'image d'entrée dans un espace latent ayant une dimension de 128. Il est constitué de plusieurs couches de convolution qui se succèdent, chaque couche étant suivie d'une normalisation par lot et d'une fonction d'activation LeakyReLU.
Ces strates autorisent une extraction hiérarchique des propriétés visuelles.
Les couches convolutives produisent ensuite une sortie qui est aplatie et transmise à une couche entièrement connectée, générant ainsi le vecteur latent z.
Cette représentation latente sert de description condensée de l'image d'entrée, tout en préservant les données cruciales requises pour la reconstruction.

- **Architecture du générateur (décodeur)**

Le générateur, aussi connu sous le nom de décodeur, a pour mission de reconstituer l'image d'origine à partir du vecteur latent z.
Il est élaboré de façon symétrique par rapport à l'encodeur.
Initialement, le vecteur latent est transféré dans un espace de haute dimension grâce à une couche totalement reliée. Cette représentation est par la suite graduellement convertie en image grâce à une série de couches de convolution inversée.
La couche finale applique une fonction d'activation **Tanh**, assurant que les valeurs produites soient conformes à la normalisation mise en œuvre sur les données entrantes.

❖ Architecture du Discriminateur latent

Le discriminateur latent opère directement dans l'espace latent. Son rôle est de distinguer les vecteurs latents produits par l'encodeur de ceux échantillonnés à partir d'une distribution gaussienne standard $\mathcal{N}(0, I)$.

Il est implémenté sous la forme d'un réseau entièrement connecté composé de plusieurs couches linéaires et de fonctions d'activation **LeakyReLU**, suivies d'une couche de sortie sigmoïde. Ce mécanisme impose une contrainte adversariale sur l'espace latent et force l'encodeur à produire des représentations conformes à la distribution cible.

4.4.2 Fonctions de perte et stratégie d'apprentissage

Le modèle AAE est entraîné sur la base de deux fonctions de perte majeures :

- **Perte de reconstruction**, calculée à l'aide de l'erreur quadratique moyenne (MSE) entre l'image originale et l'image reconstruite. Elle garantit la fidélité de la reconstruction.
- **Perte adversariale latente**, basée sur l'entropie croisée binaire (BCE), qui met en compétition l'encodeur et le discriminateur latent.

L'encodeur et le générateur optimisent une fonction de coût globale qui est un mélange pondéré de leurs deux pertes, mettant l'accent sur la qualité de la reconstruction. L'optimisation est effectuée en utilisant l'algorithme Adam, avec des taux d'apprentissage séparés pour le discriminateur latent et le duo encodeur-générateur.

4.4.3 Procédure d'entraînement

Le modèle AAE a été formé à l'aide d'une stratégie alternative qui fusionne l'apprentissage de reconstruction et l'apprentissage adversarial. Lors de chaque itération, nous mettons d'abord à jour le discriminateur latent pour qu'il puisse différencier les vecteurs latents tirés de la distribution gaussienne a priori de ceux générés par l'encodeur. Par la suite, l'encodeur ainsi que le générateur sont formés simultanément afin de réduire l'erreur de reconstruction tout en essayant de duper le discriminateur latent.

Cette interaction conflictuelle facilite une régularisation optimale de l'espace latent tout en assurant une reconstruction précise des images. Le modèle est entraîné sur une période de 20 époques, en surveillant constamment les fonctions de perte à chaque itération. De plus, des représentations visuelles régulières des images reconstruites sont réalisées pour juger de manière qualitative la convergence du modèle et l'amélioration de la qualité des reconstructions pendant la phase d'entraînement.

4.4.4 Analyse de l'espace latent

Après la phase de formation, on extrait les vecteurs latents des images de test pour examiner la structure de l'espace latent assimilé par le modèle. On applique ensuite des techniques de réduction de dimension comme l'Analyse en Composantes Principales (PCA) et le t-SNE, afin de projeter ces représentations latentes dans un espace à dimension réduite pour en faciliter la visualisation. Les données recueillies démontrent une structuration logique de l'espace latent, illustrée par l'émergence de clusters distincts associés aux diverses catégories de chiffres écrits à la main. Cette organisation valide l'efficacité de la régularisation antagoniste introduite par le discriminateur latent et illustre la compétence du modèle AAE à acquérir des représentations latentes pertinentes et distinctives.

4.4.5 Implémentation de modèles CAAE

Le modèle utilisé est un Conditional Adversarial Autoencoder (CAAE). Il combine un autoencodeur conditionnel pour la reconstruction des images et un apprentissage adversarial visant à structurer l'espace latent selon une distribution a priori. L'information de classe est intégrée dans l'encodeur, le générateur et les discriminateurs, ce qui permet une séparation des classes dans l'espace latent et une génération d'images conditionnelle plus contrôlée.

- **Implémentation de l'Encodeur**

L'encodeur est implémenté sous la forme d'un réseau convolutionnel suivi de couches entièrement connectées. Son rôle est d'extraire les caractéristiques visuelles des images MNIST, puis de compresser ces informations dans un vecteur latent $z$de dimension fixée, servant de représentation continue des données. Cette représentation est conçue pour être compatible avec l'apprentissage adversarial imposé dans l'espace latent. Formellement, le fonctionnement de l'encodeur peut être exprimé par la relation
$z = E(x)$. Contrairement aux autoencodeurs variationnels (VAE), l'encodeur ne produit ni moyenne ni variance, ce qui confirme que le modèle appartient à la famille des Adversarial Autoencoders (AAE).

- **Implémentation de décodeur conditionnel**

Le **décodeur conditionnel**, reçoit en entrée le **vecteur latent** $z$ainsi que le **label de classe sous forme one-hot**. Ces deux vecteurs sont d'abord **concaténés**, puis projetés dans l'espace image à l'aide de **couches entièrement connectées**, suivies de **couches de déconvolution** (ou d'un mécanisme de reshaping suivi de convolutions). Le rôle principal du générateur est de **reconstruire l'image d'origine**, de **produire des images réalistes conditionnées par la**

classe, et, lorsque le discriminateur d'image est présent, de **tromper ce dernier**. Formellement, le processus de génération peut être décrit par la relation $\hat{x} = G(z, y)$.

❖ **Implémentation du Discriminateur latent**

Le **discriminateur latent** $D_z$est implémenté sous la forme d'un **réseau entièrement connecté**. Il reçoit en entrée soit un **vecteur latent produit par l'encodeur**, $z = E(x)$, soit un vecteur **échantillonné à partir d'une distribution gaussienne standard** $\mathcal{N}(0, I)$. Son rôle est de prédire la probabilité que le vecteur latent provienne de la **distribution a priori**. L'objectif de cet apprentissage adversarial est d'imposer l'approximation $q(z) \approx p(z)$, où $q(z)$représente la distribution des codes latents appris par l'encodeur. Ainsi, l'encodeur est entraîné à **tromper le discriminateur latent**, ce qui permet de **structurer l'espace latent** et de favoriser une meilleure séparation des représentations.

❖ **Implémentation du Discriminateur d'image**

Un **discriminateur d'image** est également intégré au modèle afin de distinguer les **images réelles du jeu de données** des **images reconstruites ou générées par le générateur**. Ce composant joue le rôle de **critique visuel**, contribuant à améliorer la **netteté des reconstructions** ainsi que la **qualité perceptuelle des images générées**. Bien qu'il n'utilise pas explicitement le label en entrée, le discriminateur d'image est **conditionné indirectement par la classe** à travers les images produites par le générateur, lesquelles sont elles-mêmes générées de manière conditionnelle.

4.4.6 Fonctions de perte utilisées

❖ **Perte de reconstruction**

La perte de reconstruction est utilisée pour mesurer l'écart entre l'image d'entrée $x$et l'image reconstruite $\hat{x}$produite par le générateur. Dans le code, cette perte est implémentée à l'aide de la **norme L1** (ou MSE selon la configuration), définie par $L\_rec = \| x - \hat{x} \|$. Elle permet de garantir la **fidélité des reconstructions** et constitue la base de l'apprentissage de l'autoencodeur conditionnel.

❖ **Perte adversariale latente**

La perte adversariale appliquée à l'espace latent repose sur une **Binary Cross Entropy (BCE)**. Elle vise à aligner la distribution des vecteurs latents produits par l'encodeur avec une **distribution gaussienne standard** $\mathcal{N}(0, I)$.

Le discriminateur latent est entraîné à distinguer les vecteurs issus de la distribution a priori de ceux générés par l'encodeur, tandis que ce dernier apprend à **tromper le discriminateur**, ce qui permet d'imposer la contrainte $q(z) \approx p(z)$et de structurer l'espace latent.

- **Perte adversariale sur l'image**

Une perte adversariale supplémentaire est introduite au niveau de l'espace image à travers un **discriminateur d'image**. Cette perte encourage le générateur à produire des images **visuellement réalistes**, difficilement distinguables des images réelles du jeu de données. Elle agit comme un critère perceptuel complémentaire à la perte de reconstruction et contribue à améliorer la qualité visuelle des images reconstruites et générées.

- **Perte totale**

La fonction de perte globale utilisée pour l'entraînement du modèle est une **combinaison pondérée** des différentes pertes. Elle est définie comme la somme de la perte de reconstruction, de la perte adversariale latente et de la perte adversariale sur l'image, pondérées par des coefficients $\lambda_z$et λ_img. Ces coefficients permettent d'**équilibrer l'importance relative** entre la fidélité de reconstruction et les contraintes adversariales, assurant ainsi un apprentissage stable et efficace

4.4.7 Procédure d'entraînement

L'apprentissage du modèle suit une **stratégie alternée en trois étapes**, implémentée explicitement dans la boucle d'entraînement. Dans un premier temps, l'encodeur et le générateur sont mis à jour conjointement afin de minimiser la perte de reconstruction et de tromper les discriminateurs. Ensuite, le discriminateur latent est entraîné à distinguer les vecteurs latents échantillonnés depuis $\mathcal{N}(0, I)$de ceux produits par l'encodeur. Enfin, le discriminateur d'image est mis à jour pour différencier les images réelles des images reconstruites. Cette alternance garantit une **convergence stable** du modèle.

4.4.8 Analyse et visualisation de l'espace latent

Après l'entraînement, les vecteurs latents extraits par l'encodeur sont analysés à l'aide de méthodes de réduction de dimension telles que la **PCA** et la **t-SNE**. Ces techniques permettent de visualiser la structure de l'espace latent et de vérifier la **séparation des clusters par classe**,

l'effet du conditionnement par labels ainsi que la régularité globale de l'espace latent imposée par l'apprentissage adversarial.

4.4.9 Génération conditionnelle

Le générateur est ensuite utilisé pour produire des images de manière conditionnelle en échantillonnant des vecteurs latents depuis la distribution gaussienne $\mathcal{N}(0, I)$et en fixant explicitement la classe souhaitée. Cette procédure démontre la capacité du modèle à **générer des chiffres spécifiques de façon contrôlée**, tout en conservant une bonne qualité visuelle.

4.4.10 Implémentation Conditional Adversarial Autoencoder hybride (CAAE + régularisation variationnelle)

Le modèle implémenté dans ce code est un **Conditional Adversarial Autoencoder hybride (CAAE avec régularisation variationnelle)**. Il repose sur la combinaison d'un autoencodeur conditionnel capable de reconstruire les images et de générer des échantillons contrôlés par la classe, d'un apprentissage adversarial appliqué à l'espace latent afin d'imposer une distribution gaussienne et d'assurer sa régularité, ainsi que d'un discriminateur conjoint image–latent qui renforce la qualité visuelle des images produites et la cohérence entre l'image générée, le vecteur latent et l'étiquette de classe. Cette architecture permet ainsi d'obtenir simultanément une reconstruction fidèle des images MNIST, une structuration efficace et continue de l'espace latent, et une génération conditionnelle contrôlée par classe.

- **Implémentation de l'Encodeur (Encoder)**

L'encodeur est implémenté comme un **réseau convolutionnel profond** composé de plusieurs couches Conv2D suivies de BatchNorm et LeakyReLU. Son rôle est d'extraire progressivement les caractéristiques visuelles des images MNIST redimensionnées en 64×64, puis de les projeter dans un espace latent compact.

Contrairement à un AAE classique, ton encodeur produit **une moyenne (μ) et une variance (logσ^2)**, puis applique une **reparamétrisation stochastique**, exactement comme dans un VAE. Le vecteur latent final est donc échantillonné selon $z = \mu + \epsilon \cdot \sigma$.
En parallèle, l'encodeur inclut un **classifieur linéaire** qui prédit la classe du chiffre à partir de

z, ce qui force le latent à contenir de l'information discriminante. Cette conception rend l'espace latent à la fois **continu, régulier et structuré par classe**.

- **Implémentation du Générateur (Décodeur conditionnel)**

Le générateur agit comme le **décodeur conditionnel** du modèle. Il reçoit en entrée le vecteur latent $z$concaténé avec le label encodé en one-hot. Cette concaténation garantit que l'information de classe influence directement la génération.

Le générateur commence par des couches fully connected qui projettent le vecteur latent conditionné vers un tenseur spatial, puis utilise une série de **couches de convolution transposée** pour reconstruire progressivement l'image. La fonction Tanh finale permet de produire des images normalisées entre -1 et 1.
Son objectif est double : **reconstruire fidèlement les images d'entrée** et **générer des images réalistes conditionnées par la classe**, tout en trompant le discriminateur.

- **Implémentation du Discriminateur image–latent (Discriminator)**

Ce discriminateur est une composante clé de ton architecture. Il ne juge pas uniquement l'image, mais la **cohérence conjointe entre image, latent et classe**.
Il extrait d'abord des caractéristiques visuelles de l'image via des convolutions, puis traite séparément le couple $(z\text{ⓜ}, y)$à l'aide de couches fully connected.

Les deux représentations sont ensuite concaténées et passées dans un classifieur final qui prédit si le triplet (image, latent, classe) est réel ou généré. Ce mécanisme force le générateur à produire des images **compatibles avec le latent et la classe**, améliorant fortement la qualité visuelle et la cohérence sémantique.

- **Implémentation du Discriminateur latent (Dz)**

Le discriminateur latent est un réseau fully connected simple chargé de distinguer les vecteurs latents issus de l'encodeur de ceux échantillonnés depuis une distribution gaussienne standard $\mathcal{N}(0, I)$.

Son rôle est d'imposer la contrainte $q(z) \approx p(z)$.
L'encodeur est entraîné à tromper ce discriminateur, ce qui force la distribution latente apprise à devenir **continue, régulière et exploitable pour la génération**. Cette étape est cruciale pour obtenir un espace latent bien structuré.

4.4.11 Fonctions de perte utilisées

La perte de reconstruction est calculée via une **MSE**, garantissant que les images reconstruites restent proches des images originales.
La perte adversariale image pousse le générateur à produire des images réalistes capables de tromper le discriminateur conjoint.
La perte adversariale latente aligne la distribution des latents encodés avec la loi gaussienne.
La perte de classification force le latent à rester discriminant par classe.
Enfin, une **perte KL** régularise la variance du latent et stabilise l'apprentissage.

Toutes ces pertes sont combinées avec des coefficients pondérés afin d'équilibrer reconstruction, génération, régularisation et discrimination.

4.4.12 Stratégie d'entraînement

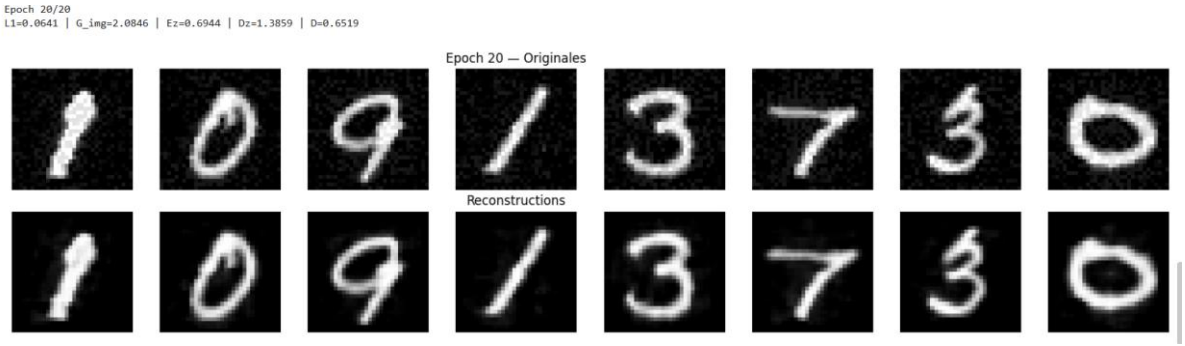
L'entraînement suit une alternance stricte en trois étapes.
D'abord, le discriminateur image–latent est entraîné à distinguer les vrais triplets des faux. Ensuite, le discriminateur latent apprend à séparer les latents réels des latents encodés. Enfin, l'encodeur et le générateur sont entraînés conjointement pour reconstruire les images, tromper les deux discriminateurs et préserver l'information de classe.

Cette stratégie garantit une **convergence stable**, évite l'effondrement du latent et assure une génération contrôlée de haute qualité.

4.5 Résultats expérimentaux

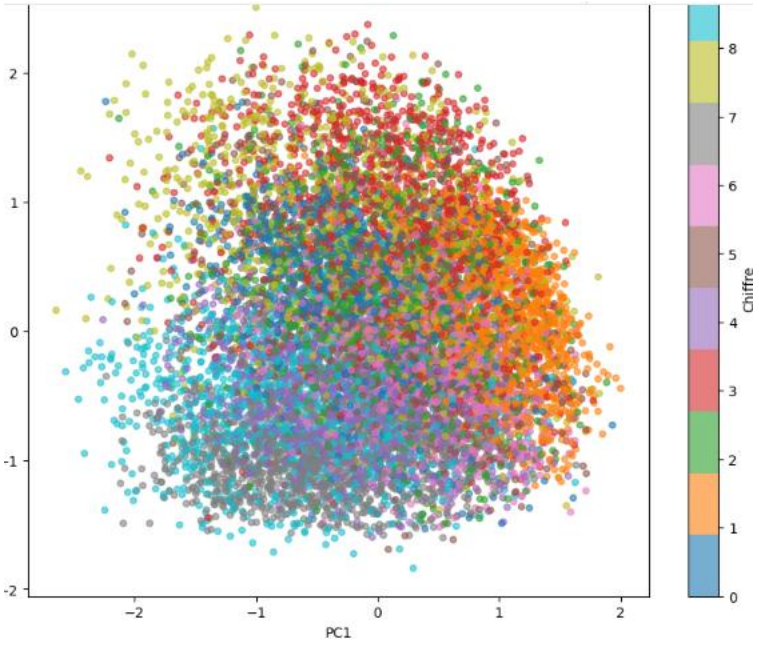
4.5.1. Résultats de AAE

- **Reconstruction des Images**



Après 20 epochs d'entraînement, l'Adversarial Autoencoder démontre une bonne capacité de reconstruction des images MNIST. Les chiffres reconstruits conservent leurs caractéristiques essentielles et restent clairement reconnaissables, malgré un léger lissage des contours. Les valeurs des différentes fonctions de perte témoignent d'une convergence stable du modèle et d'une régularisation efficace de l'espace latent, confirmant ainsi l'efficacité de l'approche AAE pour l'apprentissage non supervisé de représentations latentes structurées.

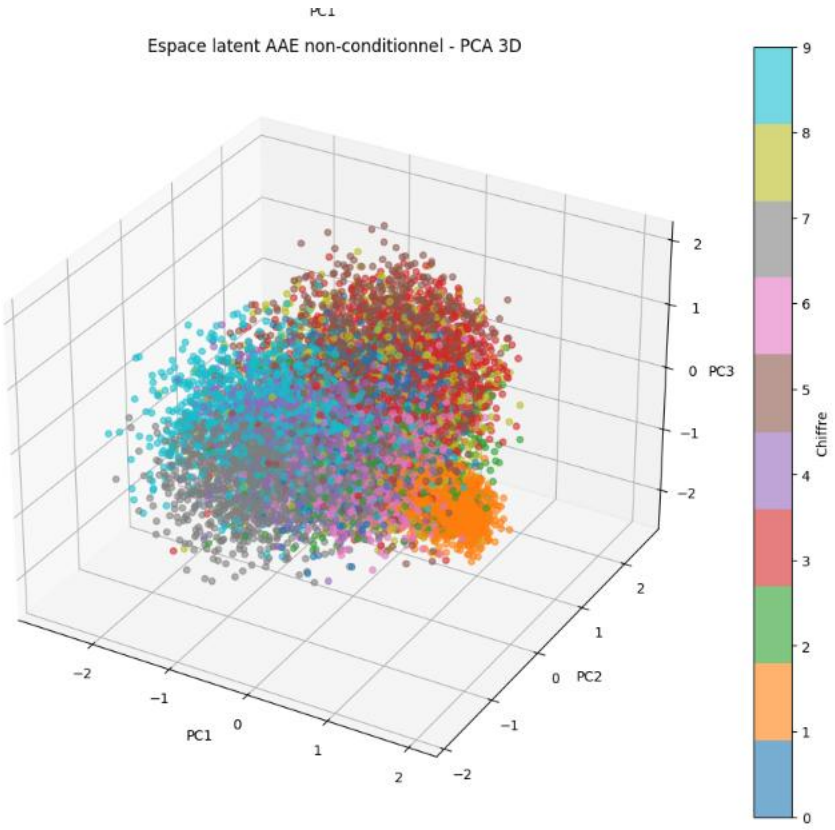
- **Espace latent PCA 2D**



Cette figure illustre la projection de l'espace latent appris par l'Adversarial Autoencoder à l'aide d'une réduction de dimension par PCA (Projection sur les deux premières composantes

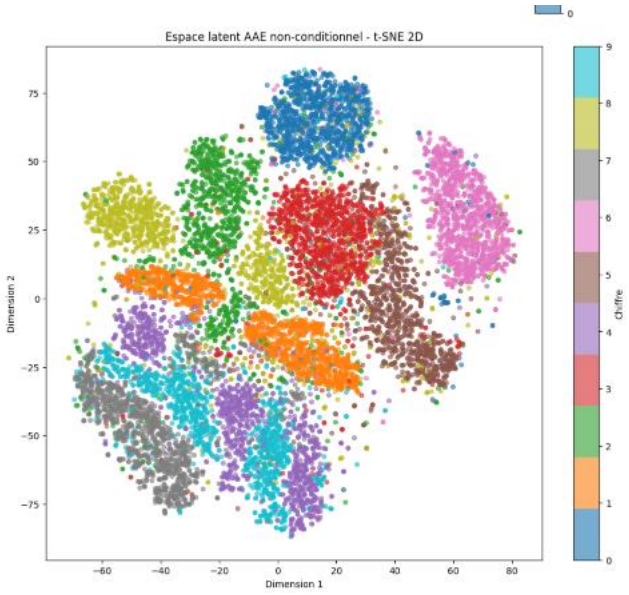
principales, PC1 et PC2). Chaque point représente une image du jeu de données MNIST encodée dans l'espace latent, tandis que les couleurs correspondent aux différentes classes de chiffres (0 à 9). On observe une distribution globalement continue et compacte des points, ce qui indique que l'AAE a appris un espace latent régularisé et cohérent. Bien que les classes ne soient pas parfaitement séparées — ce qui est attendu dans un cadre non supervisé — certaines régions montrent une concentration accrue de chiffres similaires, suggérant que le modèle capture des caractéristiques discriminantes pertinentes. Cette organisation progressive de l'espace latent confirme l'efficacité de la contrainte adversariale imposée sur la variable latente, favorisant une distribution proche d'une loi gaussienne tout en préservant la structure intrinsèque des données.

- **Espace latent PCA 3D**



La figure présentée illustre la projection en trois dimensions de l'espace latent appris par le modèle Adversarial Autoencoder (AAE) non conditionnel, obtenue à l'aide d'une analyse en composantes principales (PCA). Chaque point représente une image du jeu de données MNIST encodée dans l'espace latent, tandis que les couleurs correspondent aux différentes classes de chiffres. On observe globalement que les points forment un nuage relativement compact et continu, ce qui indique que le mécanisme adversarial a permis d'imposer efficacement une distribution *a priori* gaussienne à l'espace latent. Cette organisation homogène confirme que le modèle apprend des représentations latentes régulières et bien réparties. Toutefois, malgré cette structuration globale satisfaisante, les différentes classes ne sont pas clairement séparées : les points correspondant aux chiffres se chevauchent fortement, traduisant l'absence d'information conditionnelle dans le processus d'apprentissage. Cette superposition montre que l'AAE non conditionnel ne distingue pas explicitement les classes dans l'espace latent, ce qui limite le contrôle sémantique sur la génération des données. Ainsi, bien que l'AAE permette une structuration globale cohérente de l'espace latent, il reste insuffisant pour produire des représentations discriminantes par classe, ce qui justifie l'introduction de modèles conditionnels tels que le CAAE pour améliorer la séparation et le contrôle des représentations latentes.

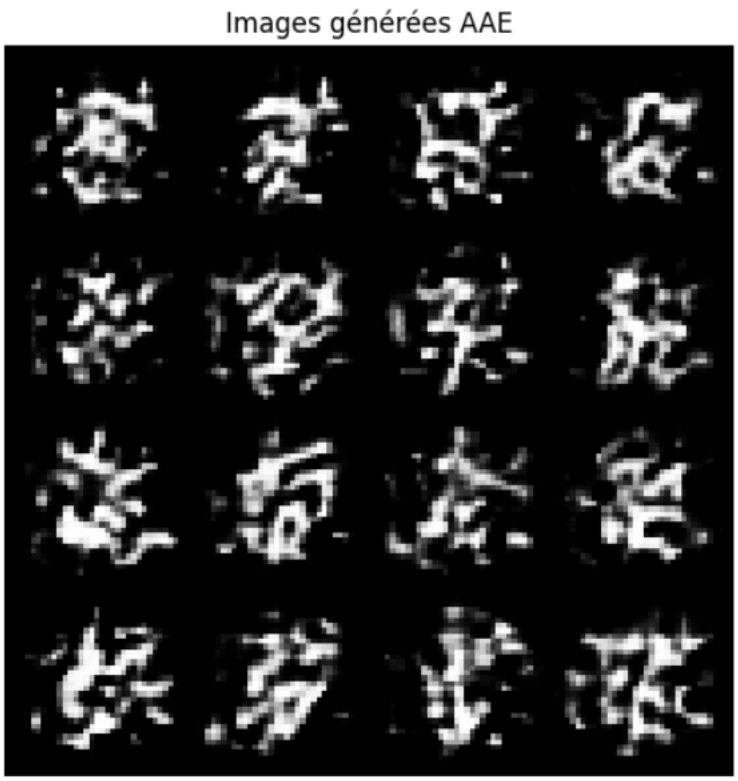
- **Espace latent T-SNE 2D**



La figure présente une visualisation bidimensionnelle de l'espace latent appris par le modèle Adversarial Autoencoder (AAE) non conditionnel, obtenue à l'aide de la méthode de réduction de dimension t-SNE. Chaque point correspond à une image du jeu de données MNIST projetée dans l'espace latent, tandis que les couleurs représentent les différentes classes de chiffres. Contrairement à la projection PCA, la visualisation t-SNE met davantage en évidence la structure locale des données, ce qui permet d'observer des regroupements relativement distincts associés à certaines classes. Ces regroupements indiquent que le modèle apprend des représentations latentes contenant implicitement des informations discriminantes, malgré l'absence explicite de conditionnement par classe lors de l'apprentissage. Néanmoins, on constate également un chevauchement partiel entre plusieurs clusters, traduisant le fait que la séparation entre classes n'est pas parfaitement maîtrisée. Cette observation confirme que, bien que l'AAE non conditionnel parvienne à organiser l'espace latent de manière cohérente et à capturer certaines similarités entre chiffres, il ne permet pas un contrôle strict et explicite de la structure sémantique de l'espace latent.

- **Génération des Images**

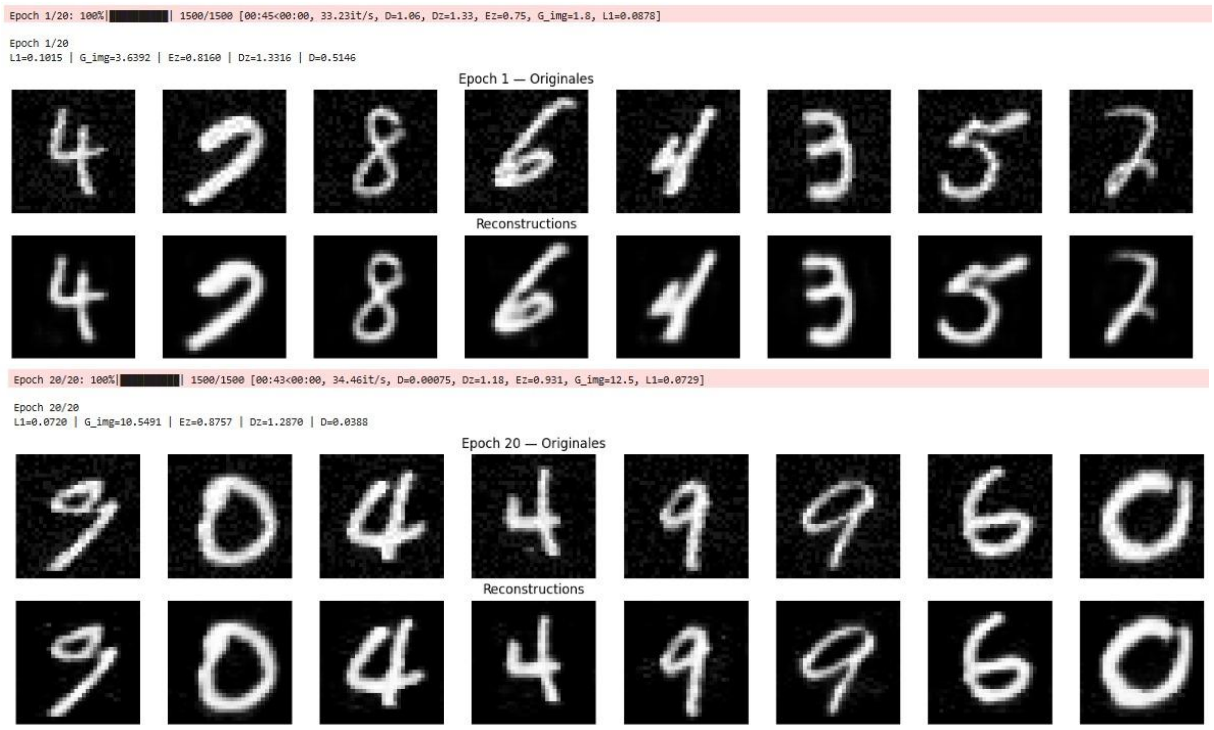
...



La figure présente un ensemble d'images générées à partir de vecteurs latents échantillonnés selon la distribution *a priori* imposée à l'espace latent par le modèle Adversarial Autoencoder (AAE) non conditionnel. Les images produites révèlent que le modèle a partiellement appris la structure globale des données du jeu MNIST, comme en témoignent certaines formes rappelant des chiffres manuscrits. Toutefois, ces images restent floues et difficilement interprétables, avec une absence de contours nets et de motifs clairement reconnaissables. Ce comportement s'explique par l'absence de conditionnement explicite sur les classes, ce qui empêche le modèle de contrôler la génération en fonction d'une information sémantique précise. De plus, bien que la régularisation adversariale permette d'assurer la continuité et la cohérence globale de l'espace latent, elle ne garantit pas une séparation suffisante des modes correspondant aux différentes classes. Par conséquent, les vecteurs latents échantillonnés produisent souvent des images ambiguës résultant d'un mélange de caractéristiques de plusieurs chiffres. Ces résultats mettent en évidence les limites de l'AAE non conditionnel en matière de génération contrôlée et soulignent la nécessité d'introduire des mécanismes de conditionnement, comme dans le CAAE ou le CAAE hybride, afin d'améliorer la qualité visuelle et la cohérence sémantique des images générées.

4.4.2. Résultats de CAAE

- **Reconstruction des Images**



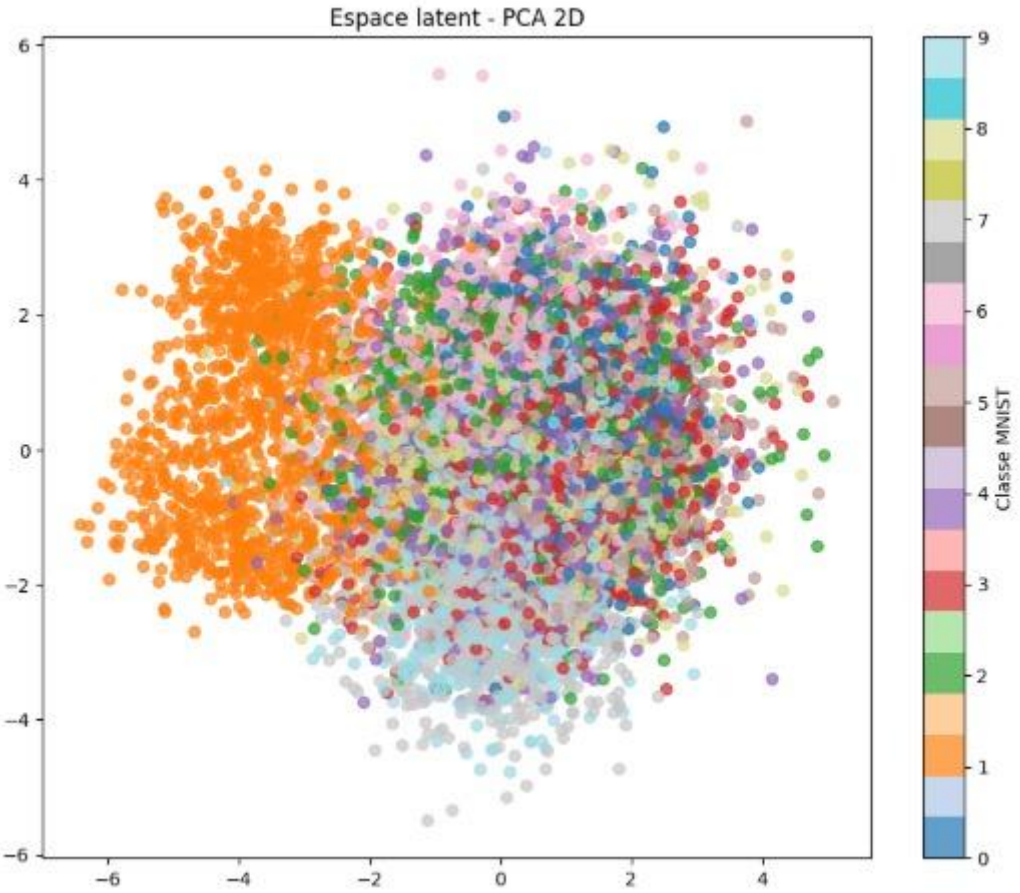
Les résultats de reconstruction montrent que le modèle est capable de restituer fidèlement les images MNIST à partir de leurs représentations latentes. À l'époque 1, les images reconstruites conservent déjà la structure globale des chiffres et leur identité de classe, bien qu'un léger flou soit encore présent. Cela indique que l'encodeur parvient rapidement à extraire les caractéristiques essentielles des images et à les compresser dans un vecteur latent informatif, tandis que le générateur est capable de produire des images reconnaissables à partir de cette représentation.

À l'époque 20, une amélioration nette de la qualité des reconstructions est observée. Les contours deviennent plus précis, le bruit diminue et les formes manuscrites sont plus proches des images originales. Cette évolution reflète la diminution progressive de la perte de reconstruction et traduit une meilleure correspondance entre les images d'entrée et celles générées par le modèle. Elle montre également que la contrainte adversariale appliquée à l'espace latent n'empêche pas le modèle de préserver l'information nécessaire à une reconstruction fidèle.

Globalement, ces résultats confirment que le Conditional Adversarial Autoencoder apprend une représentation latente à la fois compacte et structurée, permettant une reconstruction stable et

cohérente des chiffres. Le léger lissage encore visible dans certaines images est un compromis classique lié à la fonction de perte utilisée et à la régularisation adversariale, mais il reste acceptable au regard de la qualité visuelle obtenue et de la stabilité de l'apprentissage.

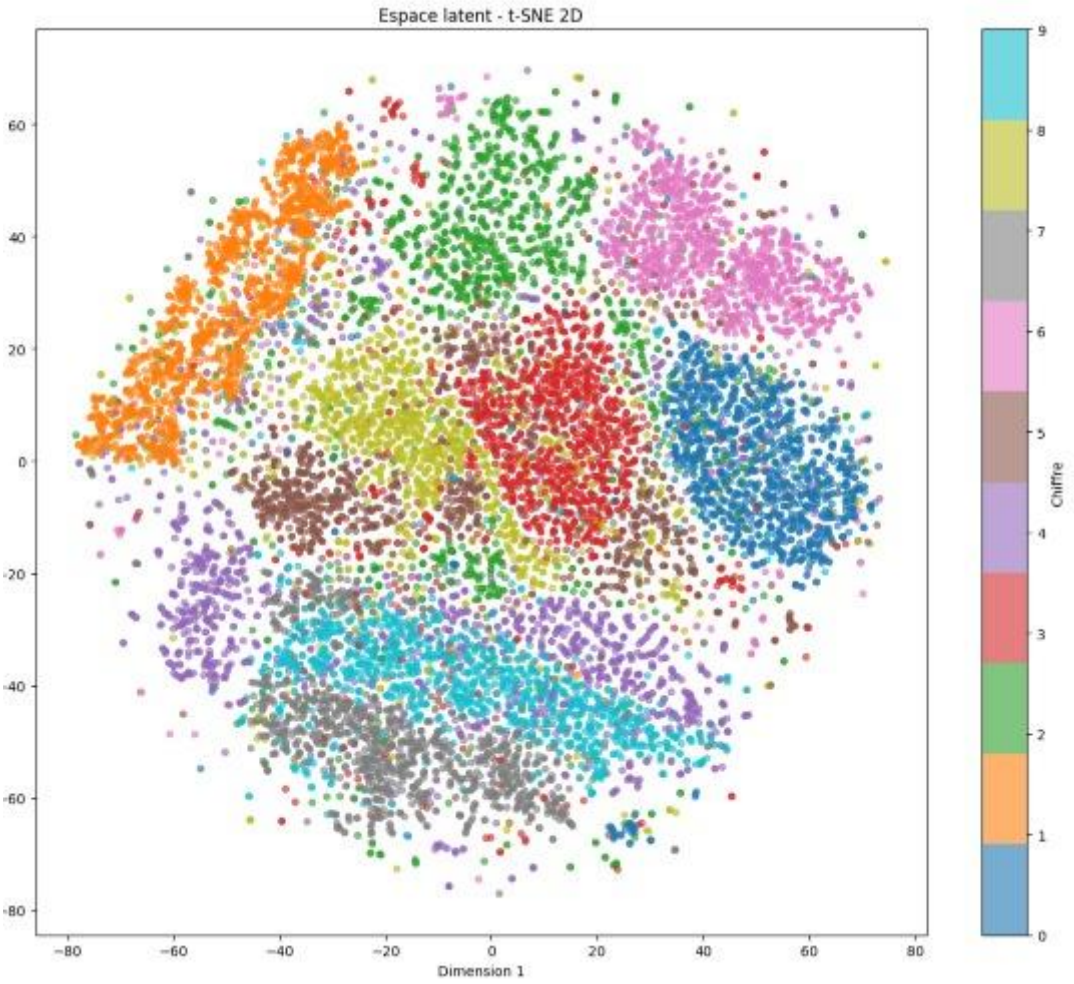
- **Espace latent-PCA 2D**



Cette visualisation de l'espace latent en PCA 2D représente la projection des vecteurs latents appris par le modèle dans un plan bidimensionnel, les couleurs indiquant les différentes classes MNIST. On observe que les points ne sont pas distribués de manière aléatoire, mais organisés selon une structure globale continue, ce qui indique que l'encodeur a appris une représentation latente cohérente et régularisée. Les classes ne forment pas des clusters parfaitement séparés, ce qui est attendu avec une PCA linéaire et une contrainte gaussienne sur l'espace latent, mais certaines régions présentent une dominance de classes spécifiques, traduisant une organisation sémantique partielle. Cette superposition entre classes montre que l'espace latent privilégie la continuité et la régularité plutôt qu'une séparation stricte, ce qui est cohérent avec l'objectif d'un Adversarial Autoencoder visant à aligner la distribution latente sur une loi normale tout

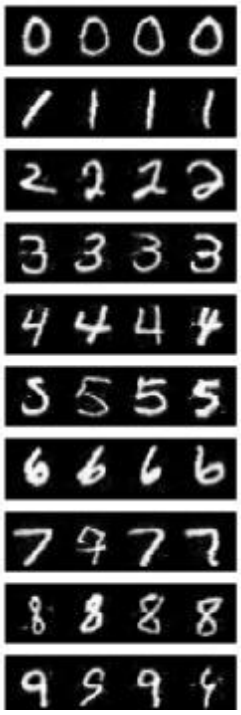
en conservant l'information discriminante nécessaire à la reconstruction et à la génération conditionnelle.

- **Espace latent T-SNE 2D**



Cette visualisation de l'espace latent en t-SNE 2D met en évidence une séparation claire des données selon les classes MNIST, chaque couleur formant des clusters bien distincts. Contrairement à la PCA, le t-SNE capture les relations non linéaires de l'espace latent, ce qui permet de révéler plus finement la structure apprise par le modèle. La formation de groupes compacts et majoritairement séparés indique que les vecteurs latents encodent efficacement l'information discriminante liée aux classes, malgré la contrainte imposée par l'apprentissage adversarial. Les zones de léger chevauchement entre certains clusters traduisent des similarités visuelles entre chiffres proches (par exemple 3 et 5, ou 4 et 9), ce qui est cohérent d'un point de vue sémantique. Globalement, ces résultats confirment que le Conditional Adversarial Autoencoder a appris un espace latent à la fois structuré, continu et discriminant, favorable à la reconstruction fidèle et à la génération conditionnelle contrôlée

- **Génération des Images**



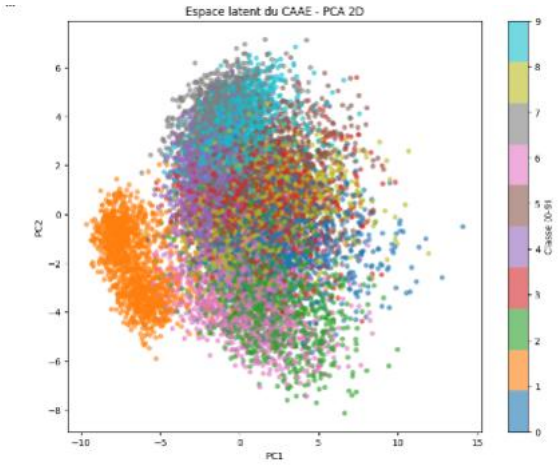
Ces résultats illustrent la capacité du modèle à réaliser une génération conditionnelle contrôlée à partir de l'espace latent appris. Pour chaque ligne, un label de classe spécifique est fixé et le générateur reçoit des vecteurs latents z échantillonnés depuis une distribution gaussienne, ce qui lui permet de produire plusieurs variantes d'un même chiffre. On observe que les images générées conservent correctement l'identité de la classe imposée, tout en présentant une diversité intra-classe au niveau de l'épaisseur des traits, de l'inclinaison et du style d'écriture. Cela montre que l'espace latent est à la fois bien structuré et suffisamment continu pour autoriser des variations réalistes autour d'une même classe. Globalement, ces résultats confirment que le Conditional Adversarial Autoencoder a appris une représentation latente cohérente, permettant une génération d'images crédibles, contrôlée par la classe et fidèle à la distribution des chiffres MNIST.

4.4.3. Résultats Conditional Adversarial Autoencoder hybride (CAAE + régularisation variationnelle)

- **Reconstruction**

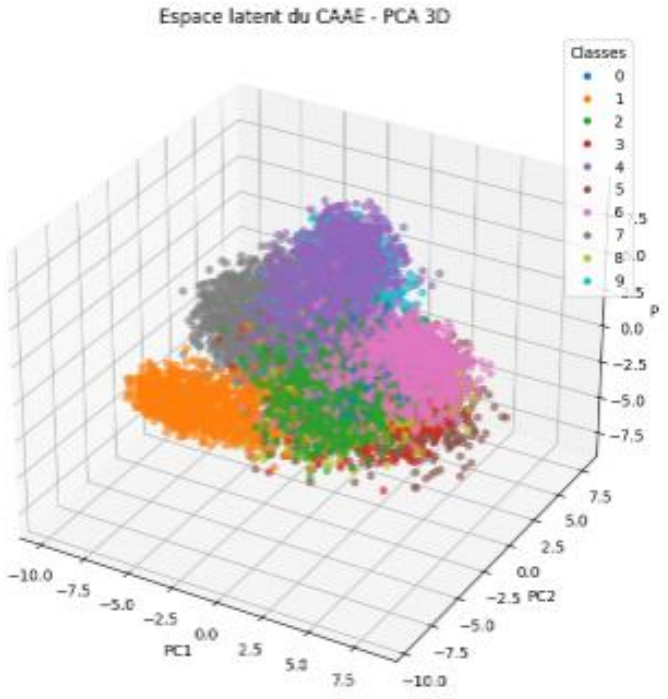
Ces résultats montrent l'évolution et la qualité de la **reconstruction** produite par le modèle au cours de l'apprentissage, jusqu'à l'époque finale. Pour chaque bloc, la ligne du haut correspond aux **images originales** du jeu de données MNIST, tandis que la ligne du bas présente les **images reconstruites** par le générateur à partir des vecteurs latents encodés et des labels de classe. On observe que, dès les dernières époques, les reconstructions sont très proches des images d'entrée : la forme globale des chiffres est correctement préservée, les contours sont nets et l'identité de chaque chiffre est clairement reconnaissable. Les légères différences visibles (épaisseur des traits, petites variations locales) traduisent le compromis normal entre fidélité de reconstruction et régularisation du latent imposée par les pertes adversariales et variationnelles. Globalement, ces résultats confirment que l'encodeur apprend une représentation latente informative et stable, et que le générateur est capable de décoder cette représentation de manière cohérente, ce qui valide la bonne convergence du modèle et l'efficacité de l'architecture CAAE hybride utilisée.

- **Espace latent PCA 2D**



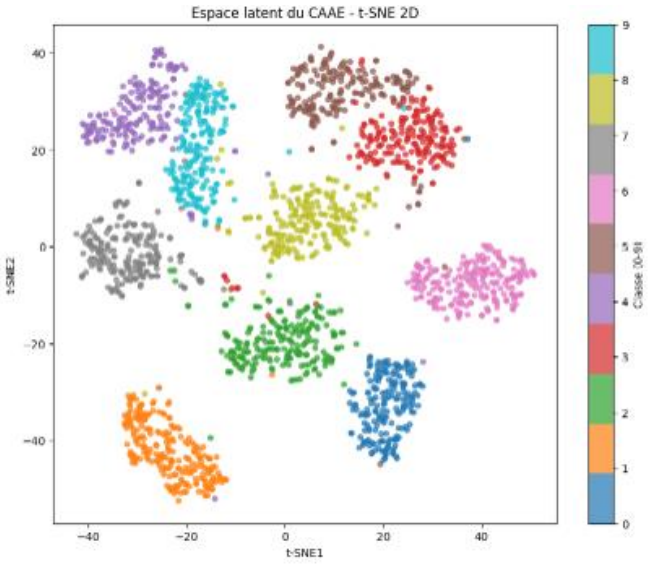
Cette figure représente une **visualisation de l'espace latent du modèle CAAE hybride projeté en deux dimensions à l'aide de la PCA**. Chaque point correspond à une image du jeu de données MNIST encodée par l'encodeur, et la couleur indique la **classe du chiffre** associée. On observe que les points ne sont pas distribués de manière aléatoire, mais organisés selon une structure globale continue, ce qui montre que l'espace latent a été correctement régularisé vers une distribution gaussienne grâce à l'apprentissage adversarial. Les différentes classes forment des **régions partiellement distinctes**, avec un certain chevauchement entre chiffres visuellement proches (par exemple 3, 5 et 8), ce qui est normal dans une projection linéaire comme la PCA. La présence de clusters cohérents indique que le conditionnement par la classe et la perte de classification ont permis d'encoder des informations discriminantes dans le latent, tout en conservant une continuité globale favorable à la génération. Globalement, cette visualisation confirme que le modèle apprend un espace latent structuré, régulier et informatif, adapté à la reconstruction et à la génération conditionnelle.

- **Espace latente PCA 3D**



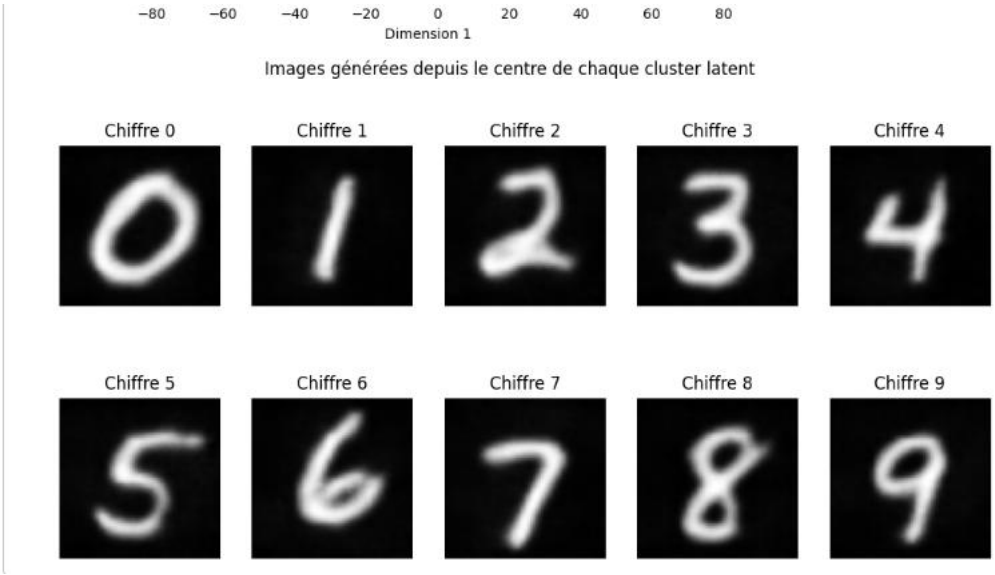
Cette figure illustre la **projection de l'espace latent du CAAE hybride en trois dimensions à l'aide de la PCA (PC1, PC2, PC3)**. Chaque point correspond à une image MNIST encodée par l'encodeur, et la couleur représente la **classe du chiffre**. On observe une organisation spatiale claire du latent : les points occupent un volume compact et continu, ce qui indique que la régularisation adversariale et variationnelle a bien forcé la distribution latente à se rapprocher d'une gaussienne. Les différentes classes forment des **amas partiellement séparés**, avec des régions spécifiques pour certains chiffres, tandis que d'autres présentent un chevauchement dû à leurs similarités visuelles, phénomène attendu dans un espace latent continu. La séparation est plus lisible qu'en PCA 2D, ce qui montre que l'information discriminante est répartie sur plusieurs dimensions latentes. Globalement, ce résultat confirme que le CAAE hybride apprend un espace latent structuré, régulier et sémantiquement organisé, favorable à la reconstruction fidèle et à la génération conditionnelle contrôlée.

- **Espace latent T-SNE 2D**



Cette figure montre la **visualisation de l'espace latent du CAAE hybride à l'aide du t-SNE en deux dimensions**. Chaque point correspond à une image MNIST encodée par le modèle, et la couleur indique la **classe du chiffre**. Contrairement à la PCA, le t-SNE met l'accent sur les relations locales, ce qui explique la **séparation très nette des clusters par classe** observée sur l'image. Chaque chiffre forme un groupe compact et bien distinct, ce qui indique que le CAAE hybride a appris des représentations latentes fortement discriminantes, où les échantillons d'une même classe sont proches les uns des autres et éloignés des autres classes. Cette structuration est le résultat combiné du conditionnement par la classe, de la perte de classification et de la régularisation adversariale/variationnelle appliquée au latent. Globalement, ce résultat confirme que l'espace latent appris est non seulement régulier et continu, mais aussi sémantiquement organisé, ce qui est essentiel pour une génération conditionnelle contrôlée et pour des tâches d'analyse ou de séparation de classes.

- **Génération des Images**



Cette image illustre la **génération d'images à partir du centre de chaque cluster de l'espace latent appris par le CAAE hybride**. Pour chaque classe (chiffres 0 à 9), un vecteur latent représentatif — correspondant au centre du cluster associé dans l'espace latent — est sélectionné, puis injecté dans le générateur avec le label de classe correspondant. Les images produites sont des **prototypes de classe**, qui capturent les caractéristiques moyennes de chaque chiffre (forme globale, orientation et structure des traits). La forte lisibilité et la cohérence visuelle des chiffres générés montrent que chaque région de l'espace latent est sémantiquement bien alignée avec sa classe, ce qui confirme une bonne séparation inter-classes et une faible ambiguïté. Ces résultats valident que le CAAE hybride a appris un espace latent structuré et interprétable, dans lequel le générateur peut produire des images représentatives et stables à partir de points centraux, démontrant ainsi l'efficacité du conditionnement et de la régularisation adversariale/variationnelle.

- **Images générées par classe (0–9)**

Ces résultats illustrent clairement la **génération conditionnelle réalisée par le CAAE hybride**. Chaque ligne correspond à une **classe fixée (un chiffre précis)**, tandis que les différentes images sur une même ligne sont générées à partir de **vecteurs latents différents** échantillonnés autour de la distribution gaussienne, tout en conservant le **même label de classe**. On observe que l'identité du chiffre est parfaitement respectée pour chaque ligne (0, 1, 2, 3, 4, 5, etc.), ce qui montre que le conditionnement par la classe est correctement appris et bien intégré dans le générateur. En même temps, il existe une **variabilité intra-classe** visible (épaisseur des traits, inclinaison, forme plus

Chapitre 05 : Analyse et comparaison des résultats

5.1. Analyse des performances de reconstruction

La qualité de reconstruction constitue un critère fondamental pour l'évaluation des autoencodeurs, car elle reflète la capacité du modèle à préserver l'information visuelle essentielle lors du passage par l'espace latent.

5.1.1. Adversarial Autoencoder (AAE)

Pour le modèle **AAE**, les images reconstruites conservent la structure globale des chiffres. Néanmoins, elles présentent un flou perceptible ainsi qu'une perte de détails fins, notamment au niveau des contours. De plus, certaines classes visuellement proches peuvent afficher des similarités non souhaitées, traduisant une représentation latente encore peu discriminante.

5.1.2. Conditional Adversarial Autoencoder (CAAE)

Le **CAAE** améliore de manière significative la qualité de reconstruction grâce à l'introduction du conditionnement par classe. Les chiffres reconstruits sont plus nets et davantage cohérents avec leur classe d'origine, car le décodeur bénéficie explicitement de l'information conditionnelle. Toutefois, de légères déformations persistent pour certaines écritures complexes ou atypiques.

5.1.3. Conditional Adversarial Autoencoder hybride (CAAE + régularisation variationnelle)

Le **CAAE hybride** fournit les meilleures reconstructions parmi les trois modèles. Les chiffres obtenus sont plus précis, bien définis et visuellement très proches des images originales. Cette amélioration s'explique par la combinaison du conditionnement par classe et d'une régularisation adversariale renforcée, permettant l'apprentissage d'un espace latent plus structuré, plus stable et plus informatif.

5.2. Analyse de l'espace latent (PCA et t-SNE)

L'analyse de l'espace latent permet d'évaluer la qualité des représentations internes apprises par les différents modèles ainsi que leur capacité à séparer les classes.

5.2.1. Adversarial Autoencoder (AAE)

Les projections PCA de l'espace latent du AAE montrent une distribution globalement continue, mais caractérisée par un fort chevauchement entre les classes. Les clusters correspondant aux différents chiffres ne sont pas clairement séparés, ce qui rend l'interprétation de l'espace latent difficile.

5.2.2. Conditional Adversarial Autoencoder (CAAE)

Pour le CAAE, la PCA révèle une organisation plus structurée de l'espace latent. Les points appartenant à une même classe tendent à se regrouper, bien que des recouvrements subsistent entre certaines classes visuellement similaires, indiquant une séparation encore partielle.

5.2.3. Conditional Adversarial Autoencoder hybride (CAAE + régularisation variationnelle)

En revanche, le CAAE hybride présente une structuration nettement plus claire en PCA 2D et 3D. Les clusters sont plus compacts et mieux séparés, tout en conservant une continuité globale de l'espace latent. Cette organisation traduit une meilleure capacité du modèle à apprendre des représentations discriminantes et cohérentes.

5.3. Analyse de L'espace latent t-SNE

5.3.2. Adversarial Autoencoder (AAE)

La visualisation t-SNE confirme les observations issues de la PCA. Pour l'AAE, les clusters sont fortement entremêlés et peu distincts. Le cAAE améliore la séparation inter-classes, mais certaines frontières demeurent floues.

5.3.3. Conditional Adversarial Autoencoder (CAAE)

La visualisation t-SNE confirme les observations issues de la PCA. Le CAAE permet une meilleure séparation inter-classes grâce à l'intégration de l'information conditionnelle, conduisant à des clusters plus compacts et mieux structurés. Toutefois, certaines frontières entre classes demeurent partiellement floues, ce qui indique que la séparation n'est pas totalement parfaite. Ces résultats soulignent l'efficacité du conditionnement dans l'amélioration de la structuration de l'espace latent.

5.3.4. Conditional Adversarial Autoencoder hybride (CAAE + régularisation variationnelle)

Le **CAAE hybride** se distingue par une séparation très nette des clusters correspondant aux différentes classes. Chaque chiffre forme un groupe bien individualisé, avec une faible

dispersion intra-classe, témoignant d'une excellente structuration de l'espace latent conditionnel.

5.4. Analyse des images générées

L'analyse des images générées permet d'évaluer la capacité des modèles à exploiter l'espace latent pour produire de nouvelles données réalistes et cohérentes.

5.4.1. Adversarial Autoencoder (AAE)

Le AAE génère des images visuellement plausibles, mais sans contrôle explicite sur la classe. Bien que la variabilité soit présente, la cohérence sémantique reste limitée.

5.4.2. Conditional Adversarial Autoencoder (CAAE)

Le CAAE permet une génération conditionnelle par classe, produisant des chiffres reconnaissables et conformes à la condition imposée. Toutefois, la diversité intra-classe peut parfois être restreinte, avec une tendance à produire des formes similaires.

5.4.3. Conditional Adversarial Autoencoder hybride (CAAE + régularisation variationnelle)

Le CAAE hybride démontre une capacité de génération conditionnelle de haute qualité. Les images générées à partir du centre de chaque cluster latent sont représentatives de leur classe, tandis que les variations autour de ces centres produisent une diversité de styles d'écriture tout en préservant l'identité du chiffre. Cela confirme que l'espace latent appris est à la fois structuré, continu et exploitable.

Conclusion

À travers l'analyse des performances de reconstruction, de la structuration de l'espace latent et de la qualité des images générées, il apparaît clairement que le CAAE hybride surpasse les modèles AAE et CAAE. Il offre une meilleure qualité visuelle, une organisation latente plus discriminante et une génération conditionnelle plus stable et plus diversifiée. Ces résultats confirment l'intérêt du modèle proposé pour des applications nécessitant à la fois interprétabilité, contrôle et génération réaliste des données.

Chapitre 06 : Conclusion générale et perspectives

6.1. Conclusion

Ce travail avait pour objectif principal l'étude, l'implémentation et la comparaison de modèles d'autoencodeurs adversariaux appliqués à la génération et à l'analyse d'images, en particulier les modèles **AAE**, **CAAE** et **CAAE hybride**. À travers une approche progressive, nous avons analysé l'impact du conditionnement par classe et de la régularisation adversariale sur la qualité des représentations latentes et sur les performances de génération.

Les expérimentations menées ont permis de montrer que l'AAE constitue une base solide pour l'apprentissage de représentations latentes continues, mais reste limité en termes de contrôle sémantique et de séparation des classes. L'introduction du conditionnement dans le cAAE améliore nettement la cohérence des reconstructions et permet une génération guidée par la classe, confirmant l'importance de l'information conditionnelle dans ce type de modèles.

Le **CAAE hybride**, proposé et évalué dans ce travail, a démontré des performances supérieures sur l'ensemble des critères étudiés. Il permet non seulement une reconstruction fidèle des images, mais aussi une structuration claire et discriminante de l'espace latent, ainsi qu'une génération conditionnelle réaliste et diversifiée. Les visualisations par PCA et t-SNE ont confirmé la qualité de l'organisation latente, tandis que les résultats de génération ont mis en évidence la capacité du modèle à produire des images cohérentes tout en conservant une variabilité intra-classe.

Par conséquent, cette étude confirme l'efficacité du CAAE hybride en tant que modèle robuste et fiable, qui peut allier reconstruction, interprétabilité et génération contrôlée. Il s'agit donc d'une option adaptée à une multitude d'applications dans le domaine de la vision par ordinateur et de l'apprentissage des représentations.

6.2. Limites du travail

Malgré les résultats encourageants obtenus, ce travail présente plusieurs limites qu'il convient de mentionner. Tout d'abord, les expérimentations finales ont été réalisées principalement sur le jeu de données **MNIST**, qui reste relativement simple, peu varié et composé d'images en niveaux de gris. Ce choix a été motivé par des contraintes techniques, notamment des **limitations matérielles liées aux ressources GPU**, qui ont restreint la possibilité d'entraîner efficacement les modèles sur des jeux de données plus complexes.

En effet, une première phase du travail s'est appuyée sur un **jeu de données ASL (American Sign Language)**, plus riche et plus réaliste. Cependant, ce dataset présentait des problèmes de qualité et d'organisation des données (déséquilibre entre classes, variations importantes de résolution et de conditions d'acquisition), ce qui a entraîné des difficultés lors de l phase d'entraînement et une instabilité accrue des modèles. Face à ces contraintes, un changement de dataset a été nécessaire afin de garantir une expérimentation stable et reproductible.

Par ailleurs, le **choix des hyperparamètres** (dimension de l'espace latent, pondération des pertes adversariales, paramètres d'optimisation et architecture des réseaux) a été effectué de manière empirique, en raison des limitations de temps de calcul. Une optimisation plus systématique pourrait améliorer davantage les performances du modèle.

Enfin, comme pour tout modèle basé sur un **apprentissage adversarial**, l'entraînement du CAAE hybride reste sensible à l'instabilité et nécessite un réglage fin de l'équilibre entre les différents réseaux afin d'assurer une convergence optimale.

6.3. Perspectives

Les perspectives de ce travail portent principalement sur l'extension du modèle CAAE hybride à des jeux de données plus complexes afin d'évaluer sa robustesse face à une plus grande diversité visuelle. Une optimisation plus systématique des hyperparamètres pourrait également améliorer les performances et la stabilité de l'apprentissage. Par ailleurs, l'intégration de techniques adversariales plus robustes permettrait de réduire les problèmes d'instabilité lors de l'entraînement. Enfin, le modèle ouvre la voie à des applications avancées telles que la génération conditionnelle contrôlée, l'augmentation de données et l'analyse approfondie de l'espace latent pour une meilleure interprétabilité.

Références

1. LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning*. **Nature, 521**(7553), 436–444. https://www.nature.com/articles/nature14539
2. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. https://www.deeplearningbook.org/
3. Hinton, G. E., & Salakhutdinov, R. R. (2006). *Reducing the dimensionality of data with neural networks*. **Science, 313**(5786), 504–507. https://www.science.org/doi/10.1126/science.1127647
4. Kingma, D. P., & Welling, M. (2014). *Auto-encoding variational Bayes*. arXiv preprint arXiv:1312.6114. https://arxiv.org/abs/1312.6114
5. Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., & Frey, B. (2016). *Adversarial autoencoders*. arXiv preprint arXiv:1511.05644. https://arxiv.org/abs/1511.05644