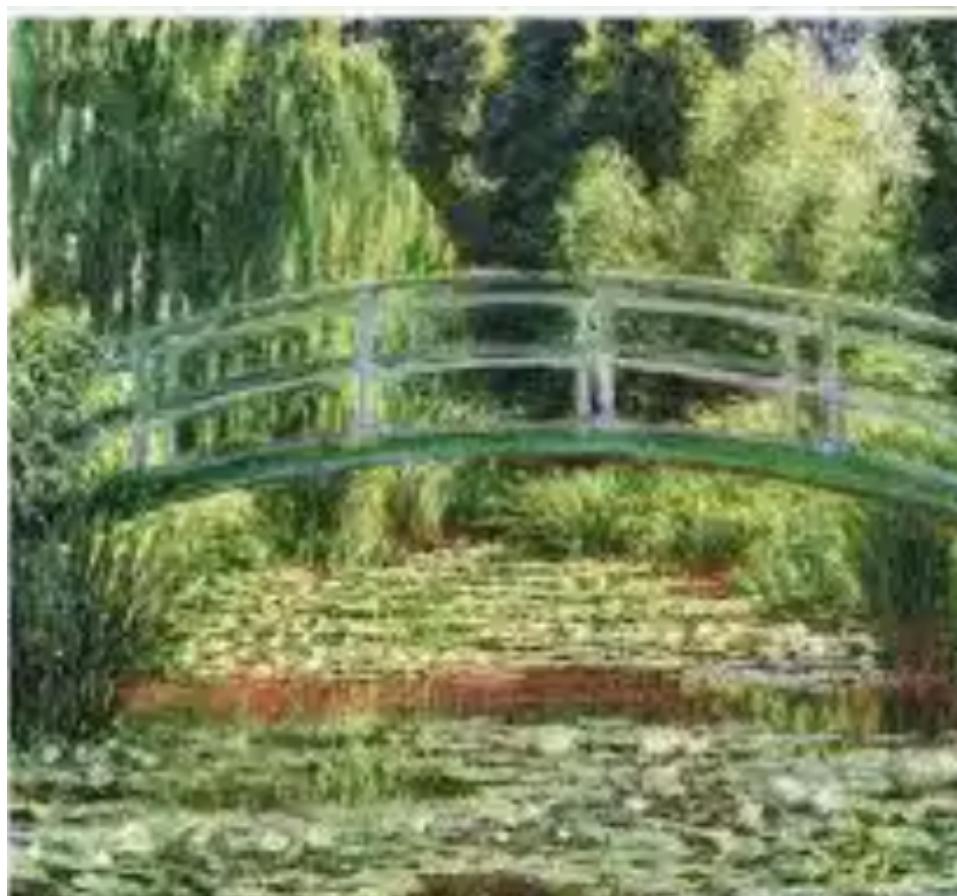


# IA Application Etude sur les GANs



## Tables des matières:

<b>1.0) Introduction</b>	<b>3</b>
<b>1.0) Analyse des données</b>	<b>4</b>
<b>3.0) GAN classique</b>	<b>8</b>
3.1) Schéma du générateur	10
3.2) Schéma du discriminator	10
3.3) Traitement :	10
3.4) Les premiers entraînements :	11
3.5) Première évaluation	12
3.6) Conclusion sur le GAN classique	13
<b>4.0) StyleGAN et StarGAN : Analyse et Résultats</b>	<b>15</b>
4.1) StyleGAN : Modèle et Fonctionnement	15
4.1.1) Entraînement :	15
Analyse des courbes :	17
4.1.2) Résultats et Tests	18
4.1.3) Interprétation des Résultats	19
4.2) StarGAN : Modèle et Fonctionnement	20
4.2.1) Entraînement :	20
4.2.2) Résultats et Tests	24
Résultats du Discriminateur	25
4.2.3) Interprétation des Résultats	26
<b>5.0) Conclusion</b>	<b>27</b>

## Table des figures :

Image 1: Schema Dataset.	4
Image 2 : Graphique de la distribution du contraste. (Peinture/Photo)	5
Image 3: Graphique des variances.	6
Image 4: Graphique de la distribution des moyennes d'intensité des pixels.	7
Image 6 :Schema GAN classique.	9
Image 7: Schema réseau GAN.	9
Image 8: Shema générateur.	10
Image 8 : Graphique de perte du discriminateur et générateur pendant l'entraînement.	11
Image 9 :Visualisation du résultat après 200 epoch.	12
Image 10 : Graphique de perte du discriminateur et générateur pendant l'entraînement avec ajustement du learning rate.	12
Image 11: Visualisation des échantillons d'évaluation du meilleur modèle.	13
Image 12 : Graphique de conclusion sur la création de fausses peintures.	14
Images Générées pendant l'entraînement	16
Image 13: Image générée après 5 epoch avec StyleGAN.	16
Image 14: Image générée après 100 epoch avec StyleGAN.	16
Image 15: Image générée après 195 epoch avec StyleGAN.	16
Image 16 : Graphique de perte du discriminateur et générateur pendant l'entraînement pour StyleGAN.	17
Image 17 : Visualisation comparaison bruit et images générées.	18
Image 18: Visualisation image Fake/Real pour StyleGAN.	19
Image 19: Image générée après 5 epoch avec StarGAN.	21
Image 20: Image générée après 30 epoch avec StarGAN.	22
Image 21 : Graphique de perte du discriminateur et générateur pendant l'entraînement pour StarGAN.	23
Image 22: Visualisation comparaison entre source, généré et cible.	24
Image 23: Visualisation image Fake/Real pour StarGAN.	25

## 1.0) Introduction

Durant ce rapport, nous allons passer en revue le travail réalisé durant le projet d'application en IA. Nous avons eu l'opportunité de collaborer avec Air Liquide, une entreprise spécialisée dans le transport de gaz, afin d'apporter notre aide à leur département R&D. Une alternative proposée était de participer à un des challenges Kaggle, offrant une multitude de projets variés. Travailler avec des professionnels nous semblait intéressant, mais un des projets Kaggle a particulièrement retenu notre attention.

Kaggle est une plateforme web qui accueille la plus grande communauté de Data Science au monde. On y retrouve des compétitions proposant de nombreux projets, notamment en deep learning. De notre côté, nous travaillons sur une veille technologique concernant les GAN dans la détection de deepfake pour le module d'IA de la recherche. Il nous semblait donc logique de continuer à explorer les GAN afin de nous spécialiser dans ce type de modèle et de monter en compétences.

Kaggle propose un projet intitulé "*I'm Something of a Painter Myself*". Ce projet consiste à utiliser un modèle de type GAN pour créer une peinture reproduisant le style du célèbre peintre Claude Monet à partir d'une photo. Nous avons ainsi décidé d'essayer différents modèles pour évaluer ceux qui fonctionnent le mieux pour ce type de tâche.

## 1.0) Analyse des données

Kaggle met à notre disposition un dataset complet comportant 7 038 images réelles et 300 peintures. Ce dataset est organisé en quatre dossiers, chacun contenant deux formats différents pour les deux ensembles : JPG et TFRecord. Ce dernier est un format propre à TensorFlow, particulièrement adapté aux méthodes de machine learning. Cependant, étant plus à l'aise avec la manipulation de données au format JPG, nous avons décidé de commencer notre analyse avec ce format.

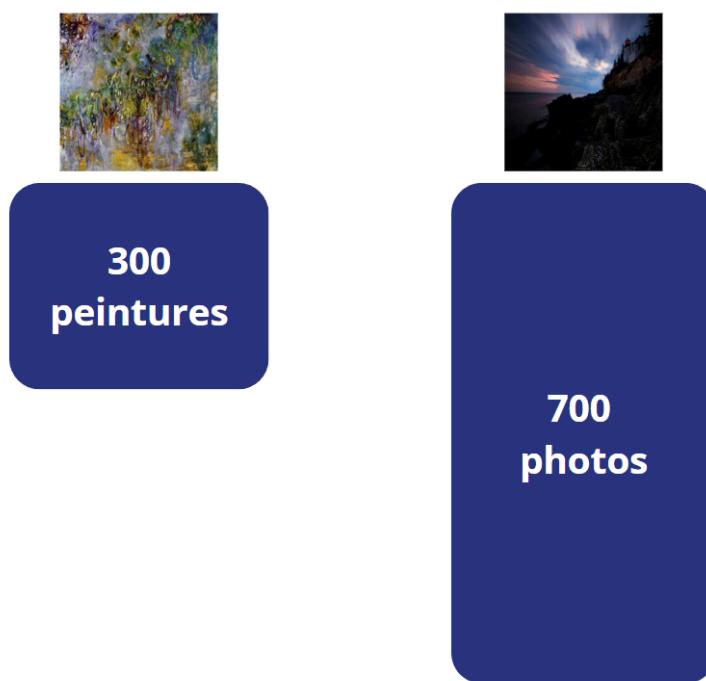


Image 1: Schema Dataset.

La première étape pour analyser les données disponibles consiste à trouver des statistiques décrivant nos deux ensembles. En examinant de plus près le style de Monet, on observe que ses peintures représentent des paysages, tout comme les photos. Cependant, la différence réside dans l'accentuation des traits, et donc dans le niveau de détails. Nous formulons ainsi l'hypothèse suivante : les peintures devraient comporter moins de détails que les images.

Pour vérifier cette hypothèse, nous utilisons différentes métriques, telles que la variance et le contraste. Le contraste représente la différence entre les zones claires et les zones sombres d'une image. Plus précisément, il s'agit de l'écart entre les tons les plus lumineux (blancs) et les plus sombres (noirs). Comme le montre le graphique ci-dessous, nos hypothèses ont été confirmées. En effet, le contraste est directement lié au niveau de détails présents sur une image. On observe naturellement une distribution des contrastes plus élevée pour les photos que pour les peintures. À noter que la ligne verticale représente la moyenne.

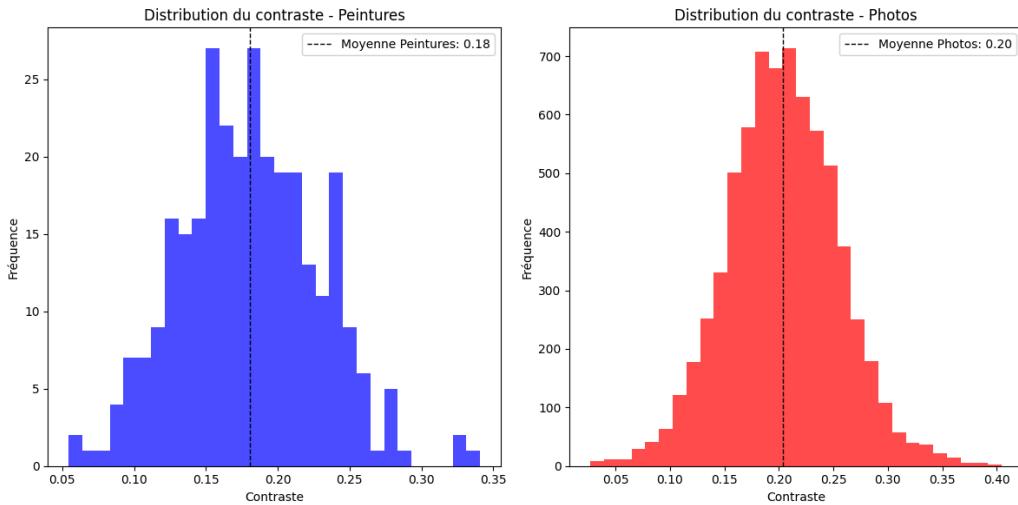


Image 2 : Graphique de la distribution du contraste. (Peinture/Photo)

Ensuite, continuons à étudier notre dataset à l'aide d'une autre métrique. La variance est une mesure statistique qui évalue la dispersion des niveaux de gris ou des couleurs autour de la moyenne des pixels de l'image. Elle reflète à quel point les valeurs de pixel sont dispersées ou proches les unes des autres. Le style des peintures étant moins précis que les photos, on devrait également distinguer une légère différence dans les distributions. C'est ce qu'on aperçoit dans le graphique suivant. On étudie la variance pour chaque couleur primaire, rouge, vert et bleu. On n'utilise pas la fréquence pour afficher la distribution mais plutôt la densité. En effet, ayant deux ensembles déséquilibrés, il est préférable d'utiliser la densité pour mieux visualiser le résultat. On y voit la distribution de la variance des couleurs pour les peintures plus à gauche que celles des photos. Ce résultat nous confirme bien l'hypothèse que nos photos sont plus détaillées.

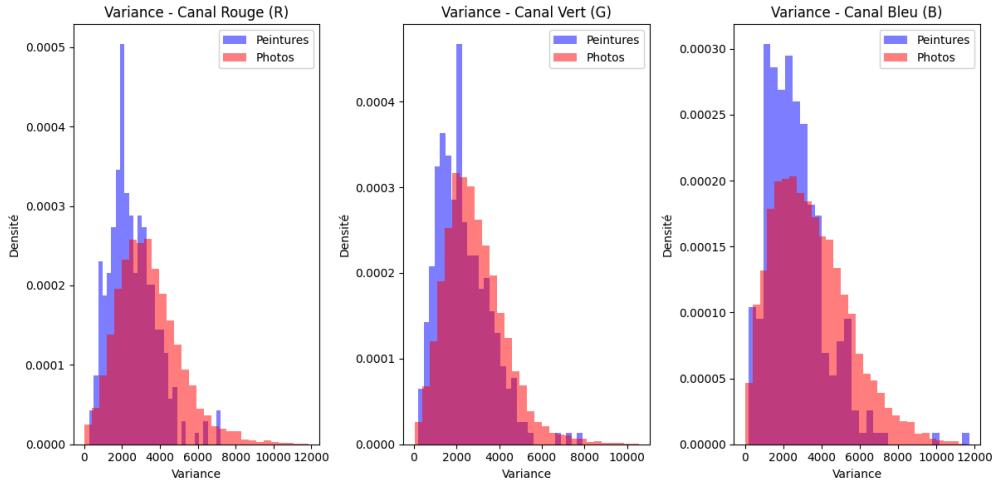


Image 3: Graphique des variances.

Enfin, la dernière métrique qui peut être mise en avant afin de nous donner plus d'informations sur notre jeu de donnée est l'intensité. En effet, l'intensité des pixels est un bon indicateur. Lorsqu'on prend une photo plusieurs facteurs sont à prendre en compte. L'exposition du sujet, l'ouverture du capteur, les iso, la température, une multitude de paramètres qui peuvent interférer avec le résultat final. En sachant que l'entièreté des photos sont des photos de paysage on va se concentrer sur un des problème, la sur exposition. La gestion de la lumière en photographie est quelque chose de très minutieux et compliqué. La peinture, quant à elle, ne connaît pas ce genre de problèmes. Voilà pourquoi nous étudions l'intensité. Sur le schéma suivant on aperçoit que la distribution de l'intensité des pixels est plus élevée pour nos photos. L'hypothèse est également confirmée.

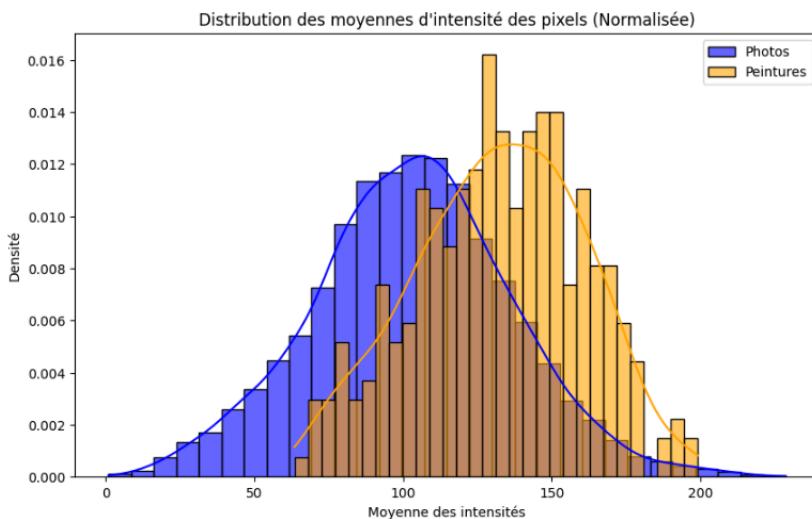


Image 4: Graphique de la distribution des moyennes d'intensité des pixels.

Afin d'aller plus loin dans l'analyse de l'intensité, on peut étudier les "boîtes à moustaches". Le résultat suivant nous en apprend plus sur la différence entre les deux ensembles. Les photos possèdent plus d'outliers au niveau de l'intensité, probablement dû aux problèmes évoqués plus tôt. On voit également qu'il y a des outliers en basse intensité laissant paraître le problème de sous exposition. Enfin, le boxplot nous permet de voir que l'intensité moyenne est plus faible pour les photos que pour les peintures.

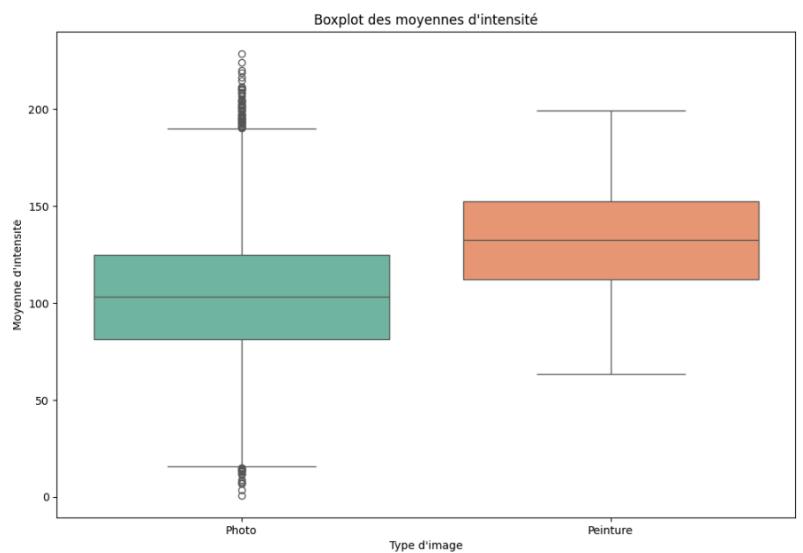


Image 5 :Boite à moustache des moyennes d'intensité.

Pour conclure sur l'analyse des données, nous avons mis en avant les points clés définissant une peinture et une photo. A partir de là on pourra évaluer les performances de générateur à transformer nos photos en peintures. Par exemple, si la moyenne des contrastes diminue, on pourra dire qu'on se rapproche d'une peinture de Monet.

### 3.0) GAN classique

Les réseaux génératifs (GANs, pour Generative Adversarial Networks) sont des modèles d'apprentissage profond composés de deux réseaux neuronaux en compétition : un générateur et un discriminateur. Ces deux réseaux interagissent dans un cadre de jeu à somme nulle, où le générateur tente de produire des images réalistes afin de tromper le discriminateur, tandis que ce dernier cherche à distinguer les images réelles des images générées.

Le générateur est chargé de créer des images à partir d'un vecteur de bruit aléatoire. Son objectif principal est de produire des images qui se rapprochent le plus possible des images réelles afin de tromper le discriminateur. Il apprend à affiner progressivement la qualité des images générées grâce aux retours fournis par le discriminateur. De son côté, le discriminateur reçoit en entrée à la fois des images réelles issues d'un jeu de données et des images synthétiques produites par le générateur. Son rôle est d'apprendre à différencier ces deux types d'images et de fournir une évaluation qui sert de rétroaction au générateur.

L'entraînement des GANs repose sur un processus itératif dans lequel les deux réseaux s'améliorent mutuellement. Le générateur produit des images à partir de bruit aléatoire et tente de duper le discriminateur. Ce dernier évalue les images générées en les comparant aux images réelles et fournit un retour sous forme d'erreur. Par la rétropropagation, les paramètres des deux réseaux sont ajustés pour améliorer leurs performances respectives. Le générateur cherche ainsi à minimiser l'erreur en créant des images plus réalistes, tandis que le discriminateur affine sa capacité à distinguer les images authentiques des images synthétiques.

Au fil des itérations, le générateur produit des images de plus en plus convaincantes, tandis que le discriminateur devient plus précis dans ses jugements, menant à une convergence où les images générées sont indiscernables des images réelles.

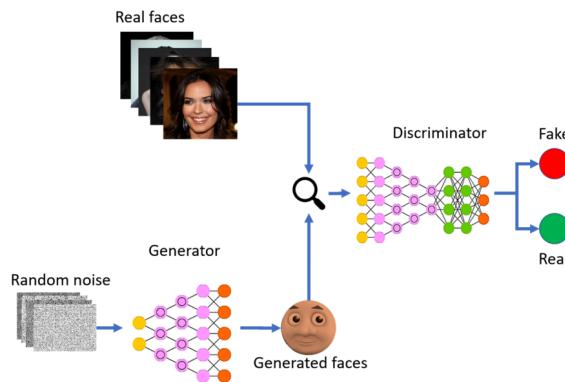


Image 6 :Schéma GAN classique.

Afin de commencer notre projet artistique de création de peinture façon Monet. Nous allons utiliser en premier lieu un modèle GAN dit “basique”. Le but est d'avoir un premier aperçu sur le fonctionnement d'un GAN et voir, par la suite, si il est possible de l'améliorer pour notre tâche. C'est pourquoi nous avons créé deux modèles (Generator et discriminator) composés de 4 et 3 couches de convolution. Ces dernières permettent d'extraire les caractéristiques de nos images. Elles vont permettre de décider si l'image est une peinture ou une photo pour le discriminator ou bien décider de reproduire des patrons ressemblant aux peintures pour le generator. L'avantage des couches de convolution est qu'elle utilise les mêmes paramètres. Cela est particulièrement utile dans le traitement d'image comme notre GAN. Regardons plus en détails :

### 3.1) Schéma du générateur

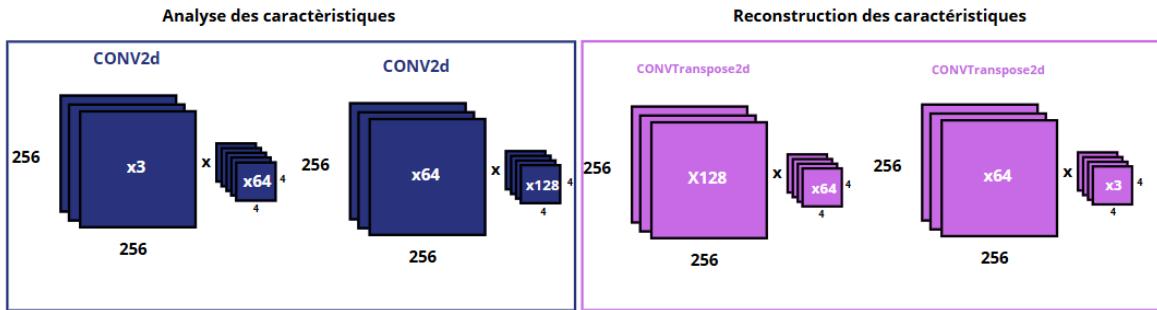


Image 7: Schéma réseau GAN.

On aperçoit qu'un générateur est divisé en deux parties. Un côté pour comprendre les caractéristiques données en entrée (nos photos) et un deuxième pour reconstruire ce qu'on veut (nos fausses peintures). A noter que entre chaque couche de convolution il y a un batch normalisation pour simplifier la convergence et limiter les fluctuations. Les fonctions d'activation utilisées sont RELU, classique dans le traitement d'image et la dernière fonction est tangente hyperbolique.

### 3.2) Schéma du discriminator

Même processus pour le discriminateur qui lui a uniquement la partie de décodage. En effet il va seulement comprendre les images en entrée afin de dire si oui ou non elles ont été générées par un générateur. C'est pourquoi la fonction d'activation est sigmoïde.

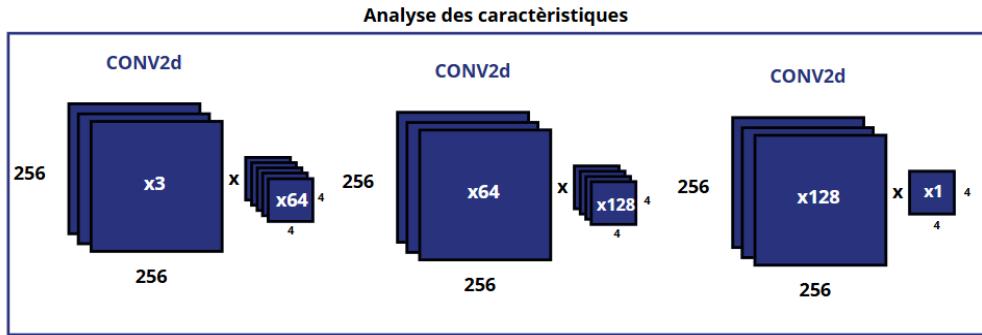


Image 8: Schéma générateur.

### 3.3) Traitement :

Peut de traitement a été fait sur nos données. En effet nos images sont déjà de même taille 256x256. Nous avons simplement normalisé nos données et transformé en Tensor pytorch, un traitement indispensable pour donner en entrée de nos modèles.

### 3.4) Les premiers entraînements :

Nous avons divisé notre entraînement en deux parties. La première est l'entraînement du discriminateur, on lui donne les peintures et les fausses peintures afin qu'il s'entraîne à reconnaître. Ensuite on entraîne le générateur en calculant la loss face aux performances du discriminateur. La loss utilisée est la binary cross entropy qui est classique dans la classification binaire, ici on veut savoir si c'est une vraie peinture ou non. Les learning rate initiaux étaient de 0.0002 pour le générateur et 0.0002 pour le discriminateur minator. Néanmoins, nous avons eu quelques problèmes car le discriminateur apprenait plus vite que notre générateur. Sur la loss ci-dessous on aperçoit un discriminateur qui est de plus en plus performant et un générateur qui a du mal à s'améliorer avec une loss qui augmente. A noter que nous avons des oscillations du au système d'apprentissage “ping pong” entre les deux modèles. C'est un comportement normal pour un GAN.

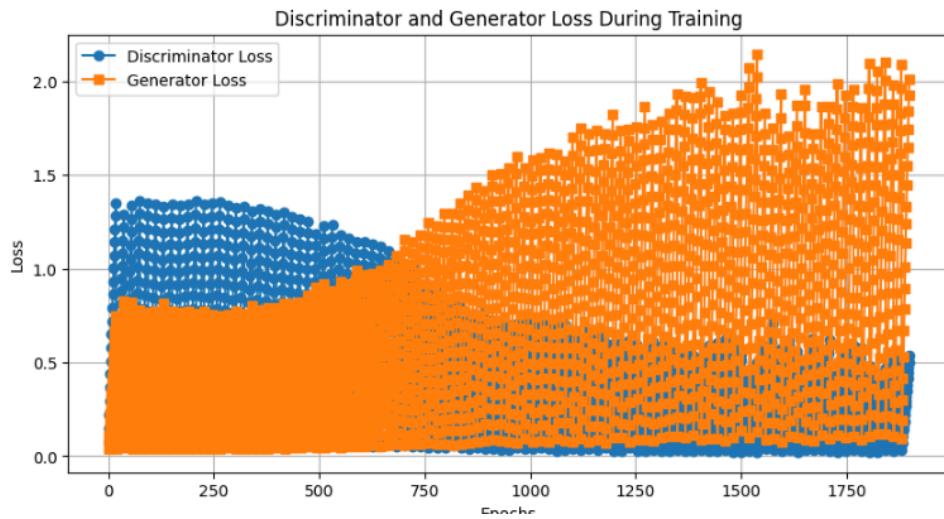


Image 8 : Graphique de perte du discriminateur et générateur pendant l'entraînement.

Nous avons tout de même décidé d'évaluer son résultat sur un échantillon afin de nous donner un aperçu.

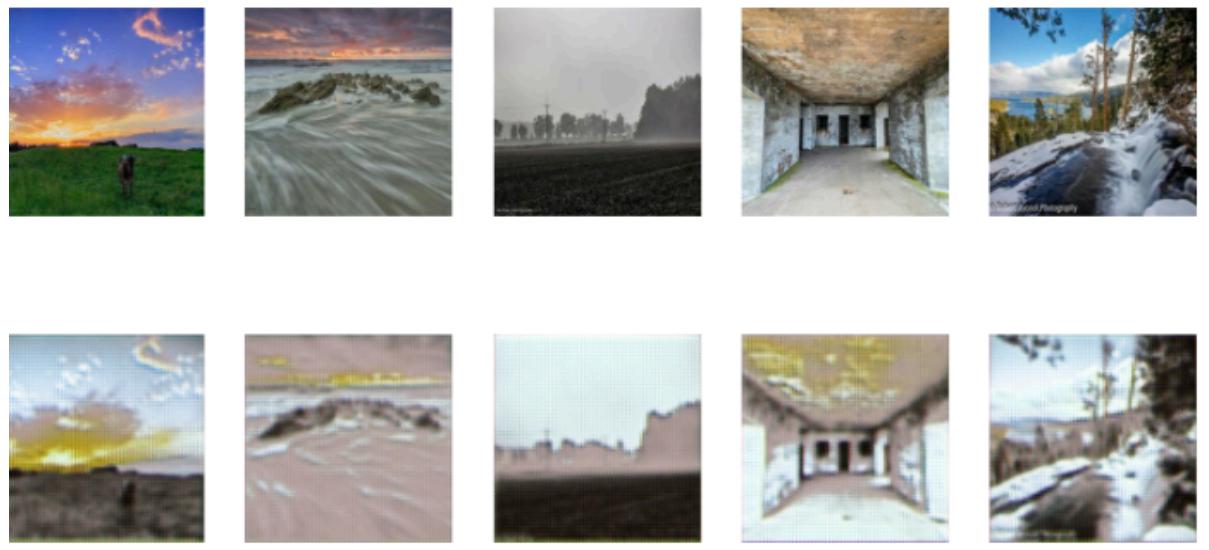


Image 9 : Visualisation du résultat après 200 epoch.

### 3.5) Première évaluation

Afin d'améliorer nos résultats nous décidons de modifier plusieurs hyper paramètres comme le nombre d'epoch, le learning rate, la loss ect... Finalement, nous avons conclu que baisser le learning rate du discriminateur permet de le faire apprendre moins vite et donc de laisser le temps au générateur d'apprendre. Nous avons donc changé le learning rate de 0.0002 à 0.00009 pour le discriminateur. Cette valeur a été choisie à la suite de nombreux essais. Sur le graphique suivant on aperçoit une loss qui converge moins vite pour le discriminateur diminuant la monté du générateur.

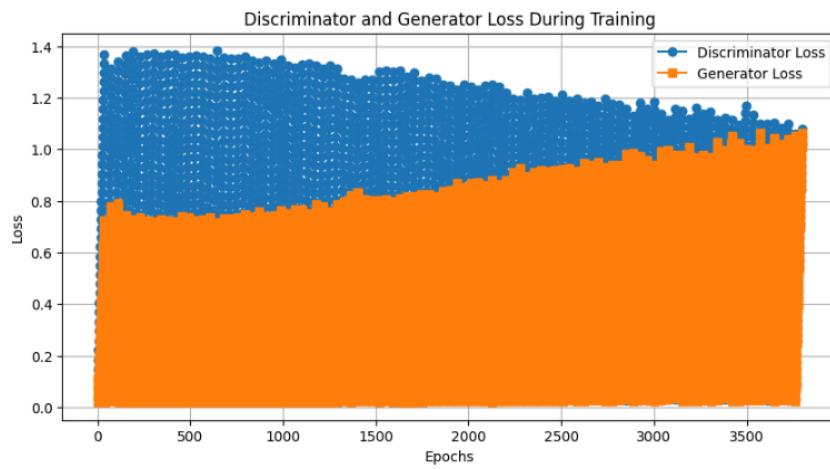


Image 10 : Graphique de perte du discriminateur et générateur pendant l'entraînement avec ajustement du learning rate.

Enfin, nous avons affiché un échantillon pour évaluer visuellement notre modèle. Il semble bien plus efficace, on reconnaît le style très particulier de Monet.

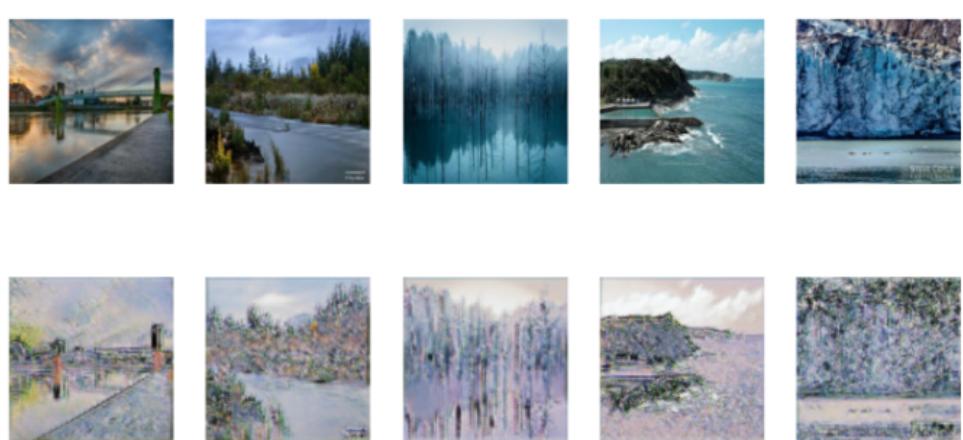


Image 11: Visualisation des échantillons d'évaluation du meilleur modèle.

Néanmoins, afin de confirmer que nous avons réussi à créer un modèle transformant des photos en un style se rapprochant de celui de Monet. Nous pouvons reprendre nos graphiques précédents.

### 3.6) Conclusion sur le GAN classique

Afin de conclure proprement sur l'efficacité de notre modèle il faut afficher l'intensité, la variance et le contraste en comparant peinture et fausses peintures. Selon nos hypothèses de départ, les distributions devraient bouger et même se rapprocher. C'est ce qu'on observe sur les graphiques suivants. Les distributions de nos photos pour la variance et le contraste nous semblaient plus élevées. Maintenant si nous comparons avec nos fausses peintures, on aperçoit des distribution nettement plus proches avec des moyenne presque identique pour le contraste. Nous pouvons donc conclure que notre modèle a réussi à transformer nos photos pour ressembler à des peintures du style de Monet.

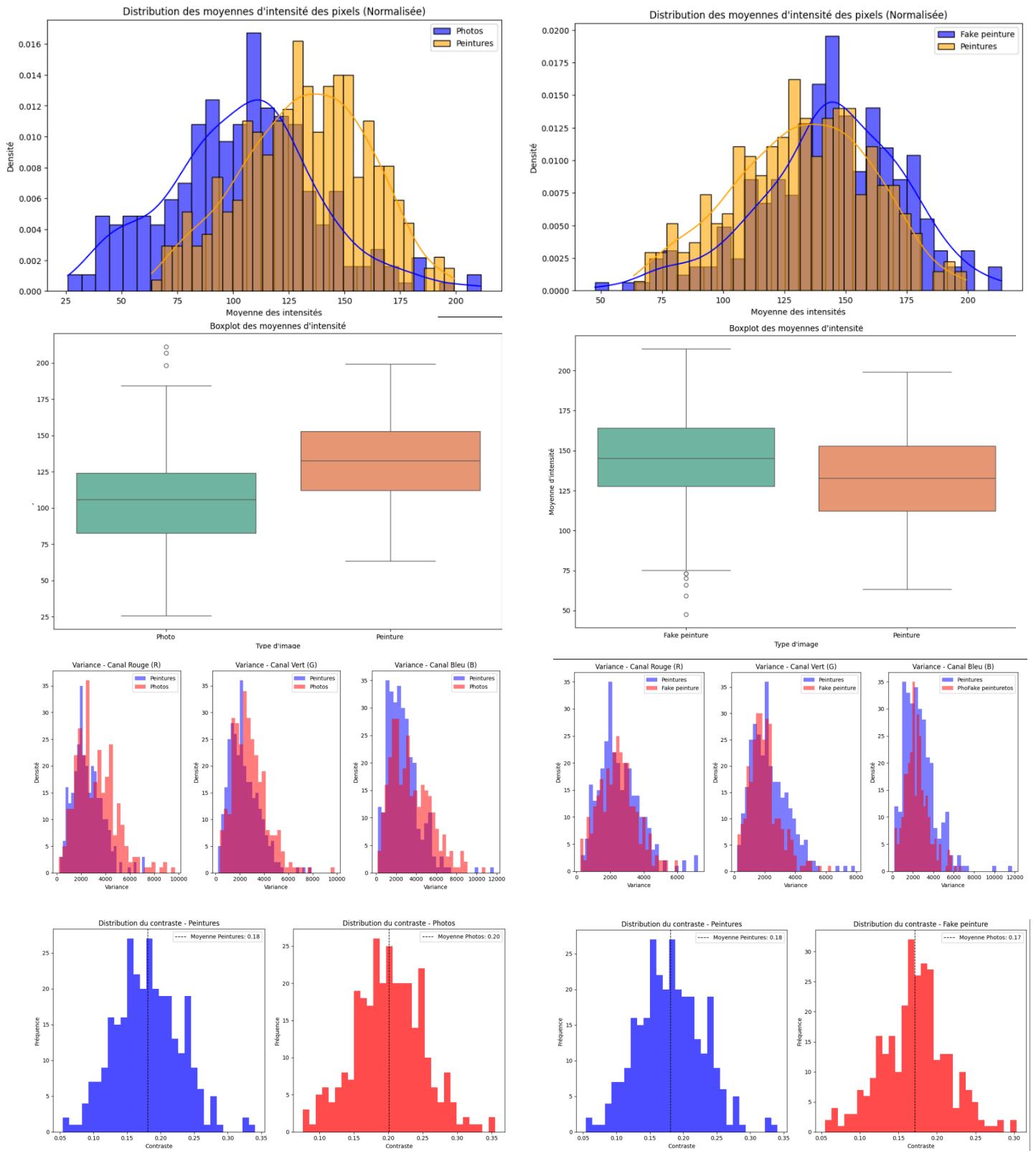


Image 12 : Graphique de conclusion sur la création de fausses peintures.

## 4.0) StyleGAN et StarGAN : Analyse et Résultats

### 4.1) StyleGAN : Modèle et Fonctionnement

StyleGAN est une architecture de réseau antagoniste génératif (GAN) qui introduit un contrôle précis sur les caractéristiques des images générées en décomposant le bruit d'entrée en éléments de style distincts. Cette approche permet de moduler les attributs de l'image à différents niveaux de détail.

#### **Architecture :**

Un vecteur latent aléatoire est transformé en un espace latent intermédiaire à l'aide d'un réseau de mappage composé de plusieurs couches entièrement connectées. Cette transformation vise à obtenir une représentation plus disentanglée des caractéristiques de l'image. Le réseau de synthèse génère ensuite l'image finale en appliquant les styles issus de l'espace latent intermédiaire à chaque couche de convolution. Chaque couche reçoit également un bruit aléatoire pour introduire des variations fines, ce qui permet un contrôle granulaire sur les détails de l'image.

L'implémentation de StyleGAN dans notre projet suit cette architecture en utilisant des couches de convolution transposées pour le générateur et des couches de convolution standard pour le discriminateur. Le générateur commence par une entrée constante qui est progressivement modifiée par les styles à chaque niveau de convolution, permettant ainsi de contrôler les caractéristiques de l'image à différentes échelles.

#### **4.1.1) Entraînement :**

Nous avons entraîné StyleGAN sur un ensemble de données comprenant 300 images réelles et 300 peintures de Monet. L'objectif était de transformer des photos en peintures impressionnistes fidèles au style de Monet. Le modèle a été entraîné pendant 200 époques.

## Images Générées pendant l'entraînement

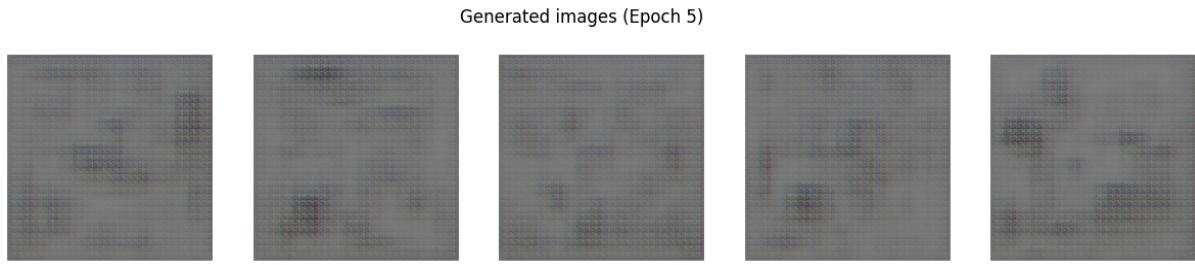


Image 13: Image générée après 5 epoch avec StyleGAN.

Après 5 époques, les images générées affichent une base grise avec des tâches sombres, reflétant une phase préliminaire où le modèle commence à saisir les premières caractéristiques du style de Monet.

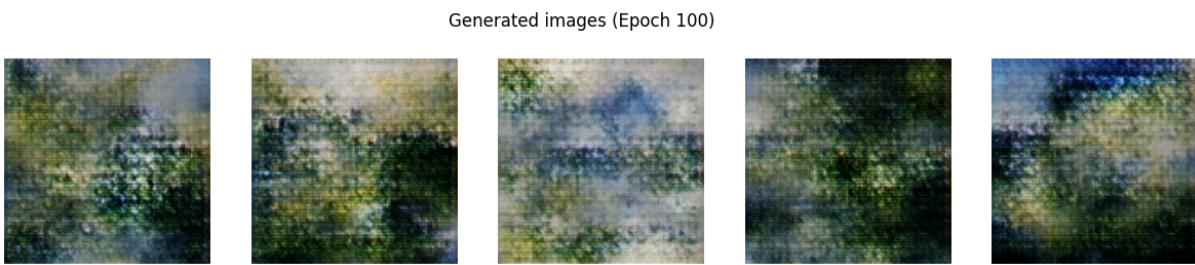


Image 14: Image générée après 100 epoch avec StyleGAN.

Après 100 époques, les progrès sont évidents. Les structures et textures se dessinent, avec des regroupements de couleurs évoquant le style impressionniste.

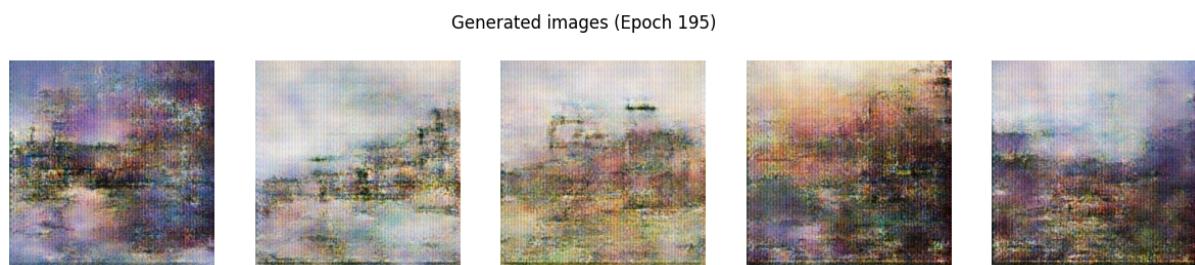


Image 15: Image générée après 195 epoch avec StyleGAN.

À l'époque 195, vers la fin de l'entraînement, les images révèlent des motifs plus précis et des couleurs typiques de Monet. Les transitions de couleur deviennent plus variées, ajoutant les premices de détails et une profondeur aux images.

## Graphique des Pertes

Le graphique ci-dessous présente les courbes de perte pour le générateur et le discriminateur, sur les ensembles d'entraînement et de validation.

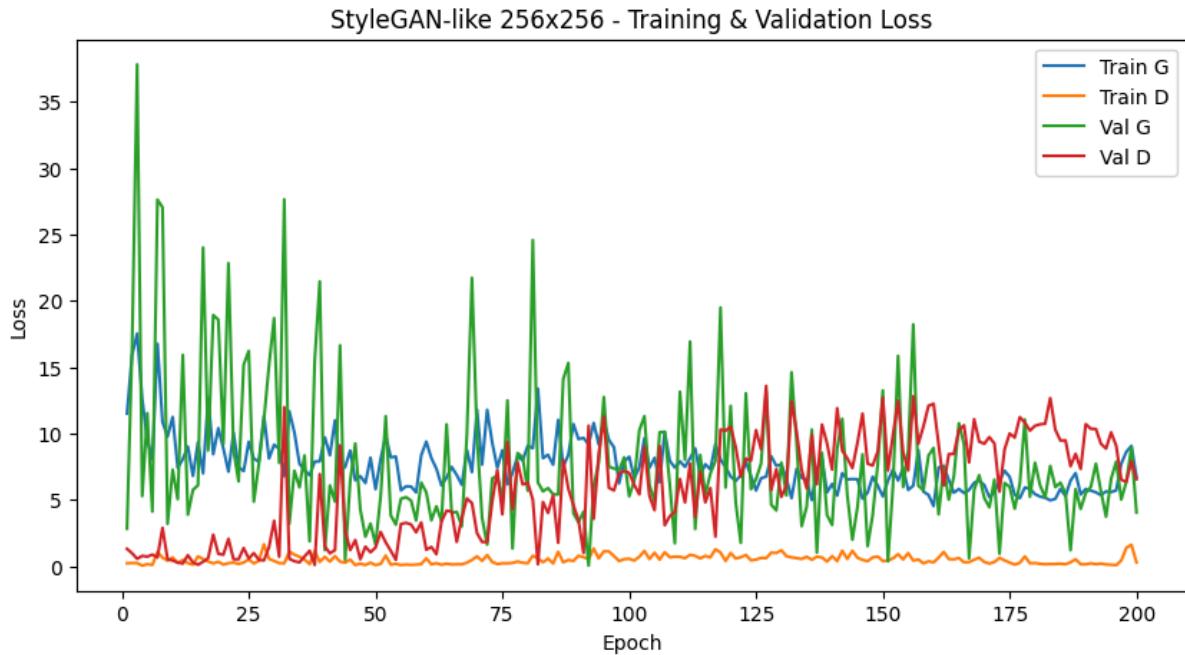


Image 16 : Graphique de perte du discriminateur et générateur pendant l'entraînement pour StyleGAN.

## Analyse des courbes :

L'analyse des courbes révèle que pour la perte du générateur (Train G et Val G), les courbes fluctuent, ce qui est typique pour les modèles GANs comme vu dans la première étude. Après environ 150 époques, la perte du générateur devient relativement stable, suggérant une convergence du modèle. Concernant la perte du discriminateur (Train D et Val D), des oscillations marquées sont observées au début de l'entraînement, ce qui est attendu étant donné la compétition entre le générateur et le discriminateur. À partir de la moitié de l'entraînement, les oscillations diminuent et les pertes deviennent plus régulières, indiquant un équilibre atteint entre les deux modèles. Les différences entre les ensembles d'entraînement et de validation montrent que les pertes pour les deux modèles restent relativement proches tout au long de l'entraînement. Cela suggère que le modèle généralise bien et qu'il n'y a pas de sur apprentissage significatif.

#### 4.1.2) Résultats et Tests

##### Comparaison Bruit et Images Générées

Pour évaluer le générateur, nous avons introduit du bruit aléatoire de dimensions  $10 \times 20$  comme entrée. Ce bruit est ensuite transformé en images synthétiques par le générateur. Voici les résultats de ce test :

- **Bruit (Entrée)** : La première ligne de l'image illustre le bruit initial utilisé comme entrée pour le générateur. Ce bruit est un vecteur aléatoire multidimensionnel.
- **Images Générées** : La seconde ligne présente les images générées à partir du bruit. Les résultats montrent que le modèle est capable de produire des images qui imitent le style de Monet, avec des textures et des couleurs typiques.

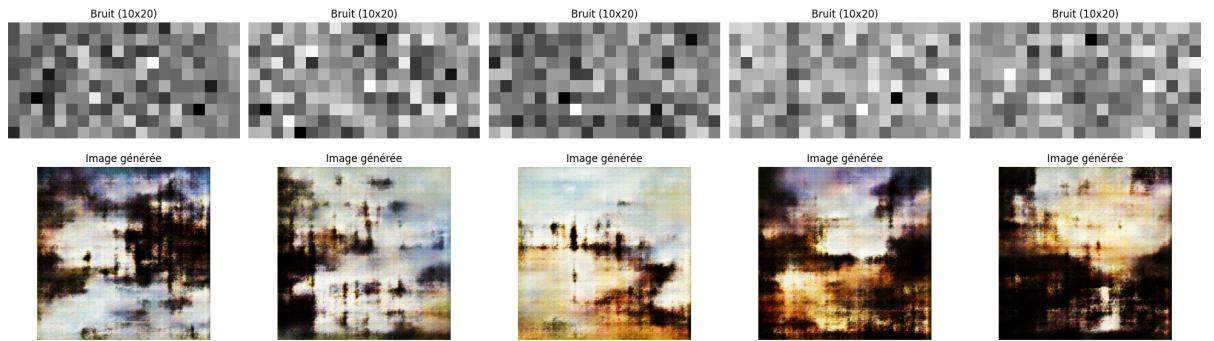


Image 17 : Visualisation comparaison bruit et images générées.

##### Test du Discriminateur

Nous avons testé le discriminateur en lui fournissant un lot d'images réelles et un lot d'images générées. Les logits de sortie permettent d'évaluer sa capacité à distinguer les vraies images des fausses. Voici les résultats :

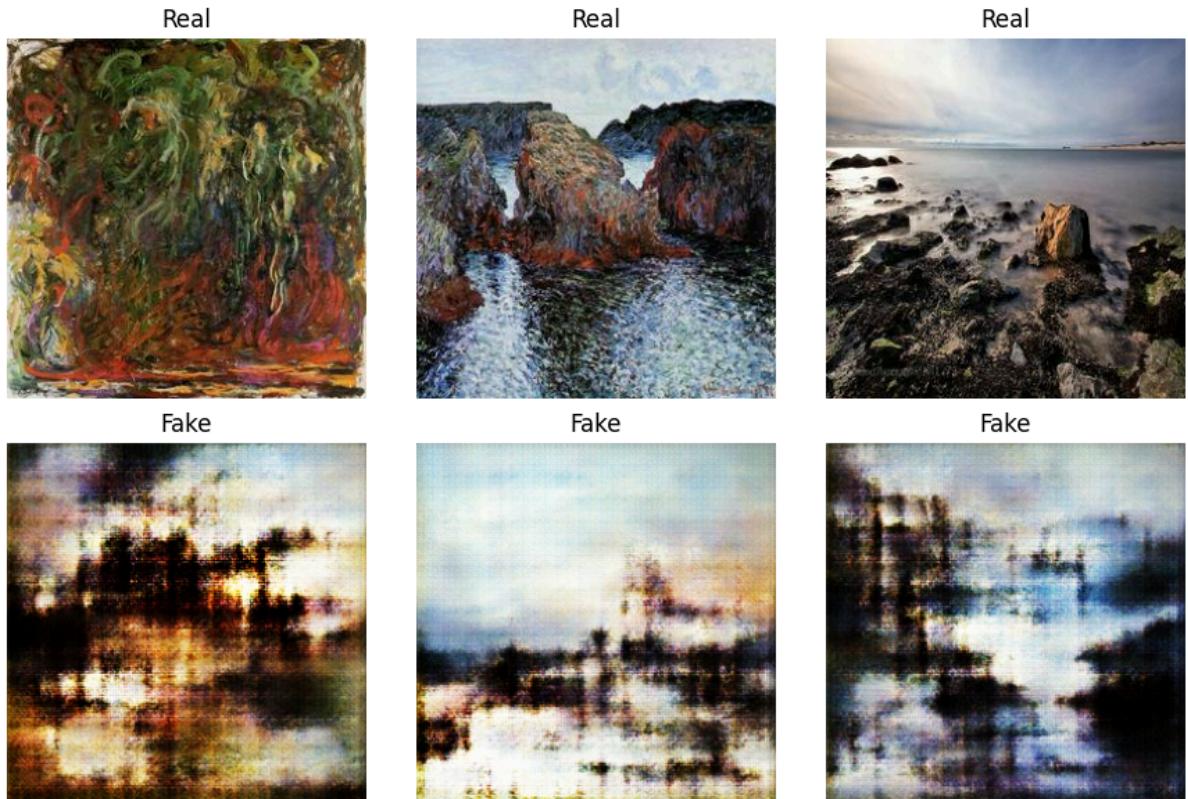


Image 18: Visualisation image Fake/Real pour StyleGAN.

- **Logits pour les Vraies Images :**  
[8.019674, 4.0475254, 7.5723143]  
Ces valeurs élevées indiquent que le discriminateur reconnaît correctement les vraies images comme authentiques.
- **Logits pour les Fausses Images :**  
[-4.917577, -1.7919527, -4.127987]  
Les valeurs négatives montrent que le modèle identifie efficacement les images générées comme fausses.

#### 4.1.3) Interprétation des Résultats

Le générateur produit des images qui tentent d'imiter le style de Monet, avec des textures et des palettes de couleurs évoquant le style impressionniste. Cependant, les résultats restent perfectibles, car les motifs générés manquent parfois de finesse et présentent des artefacts. De son côté, le discriminateur parvient à séparer efficacement les vraies images des fausses, les logits montrant qu'il identifie facilement les images générées comme artificielles. Cela suggère que le générateur n'a pas encore atteint un niveau de sophistication suffisant pour tromper le discriminateur, probablement en raison des contraintes d'entraînement.

## 4.2) StarGAN : Modèle et Fonctionnement

StarGAN est une architecture de GAN conçue pour effectuer des traductions d'images entre multiples domaines en utilisant un seul modèle. Cette capacité permet de transformer une image d'un domaine source (par exemple, une photo) vers un domaine cible (par exemple, une peinture) de manière flexible et efficace.

### **Architecture :**

Le générateur de StarGAN prend en entrée une image ainsi qu'un vecteur de conditionnement représentant le domaine cible et utilise des blocs résiduels pour apprendre les transformations nécessaires tout en conservant les caractéristiques essentielles de l'image source. Quant au discriminateur, il évalue non seulement si une image est réelle ou générée, mais prédit également le domaine auquel elle appartient. Cette double fonction permet au modèle de mieux comprendre les spécificités de chaque domaine, améliorant ainsi la qualité des traductions.

Dans notre projet, l'implémentation de StarGAN a été adaptée pour effectuer des traductions entre les photos et les peintures de Monet. Le générateur et le discriminateur ont été conçus pour gérer les spécificités de ces deux domaines, en intégrant des mécanismes de normalisation et des fonctions d'activation appropriées pour assurer une convergence stable et des résultats de haute qualité.

### 4.2.1) Entraînement :

Nous avons entraîné StarGAN sur un ensemble de données composé de 300 photos et 300 peintures de Monet sur 30 époques.

## Images Générées

### Après 5 époques

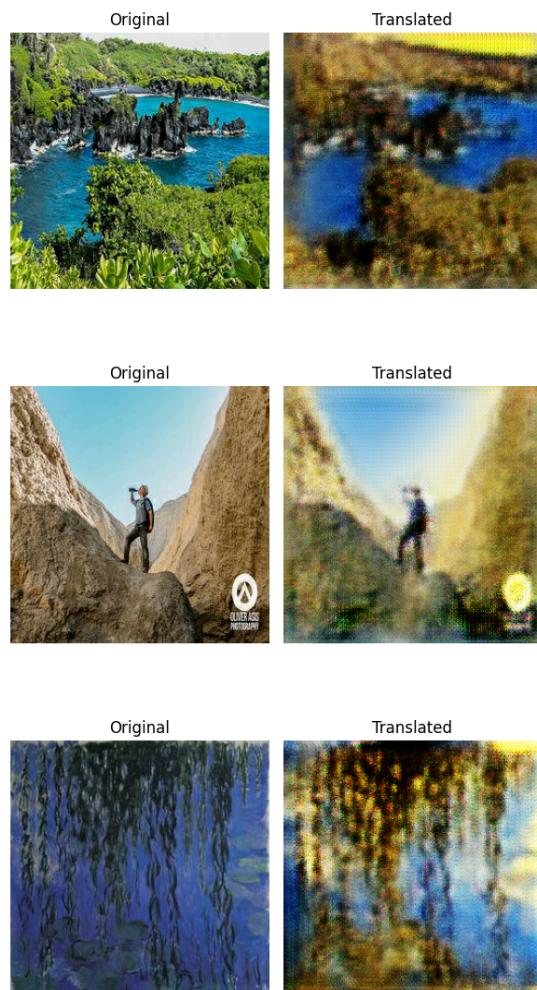


Image 19: Image générée après 5 epoch avec StarGAN.

Les images générées après 5 époques montrent que le modèle commence à apprendre les caractéristiques de base du style impressionniste. Cependant, les résultats sont encore rudimentaires avec des zones floues et un manque de détails. Le style de Monet est perceptible, mais les textures restent peu raffinées.

### Après 30 époques

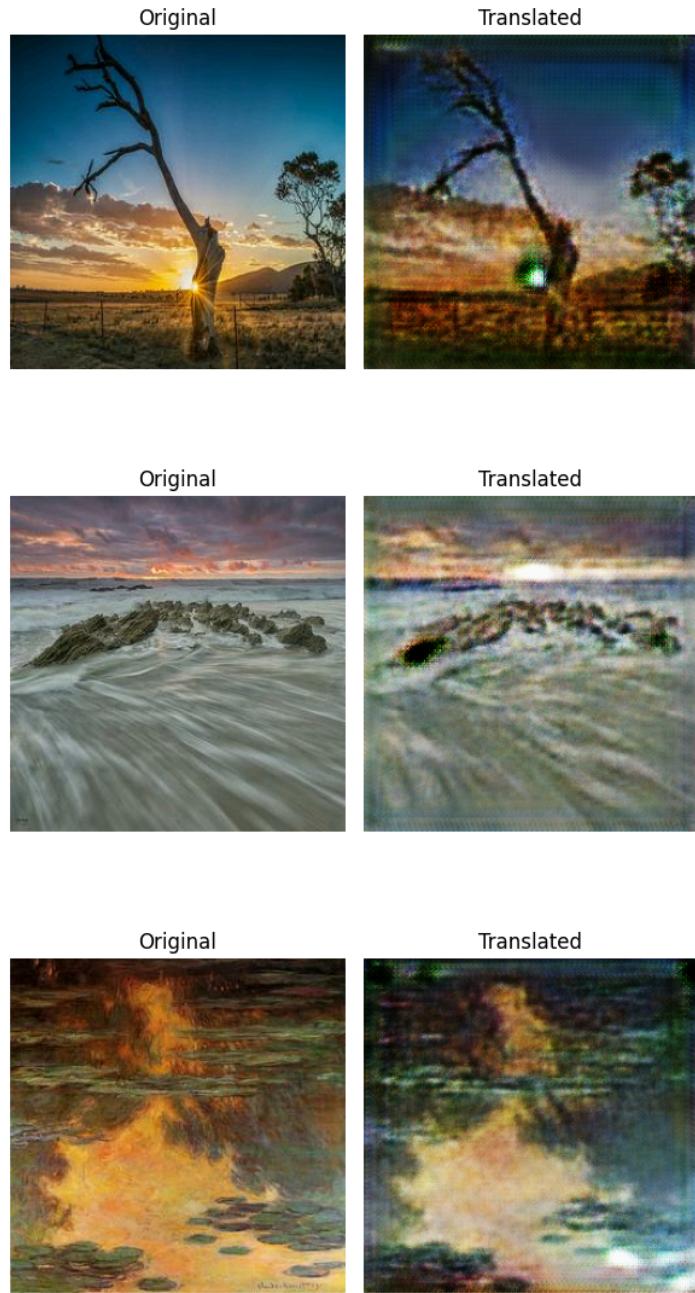


Image 20: Image générée après 30 epoch avec StarGAN.

À la fin de l'entraînement, après 30 époques, les images générées présentent une amélioration notable. Les couleurs typiques de Monet et les transitions entre les zones claires et sombres deviennent plus nettes. Cependant, des tâches de pixels blancs apparaissent, révélant des artefacts liés aux limites du modèle par son faible nombre d'échantillons d'entraînement.

## Graphique des Pertes

Le graphique ci-dessous illustre les pertes du générateur et du discriminateur pour les ensembles d'entraînement et de validation.

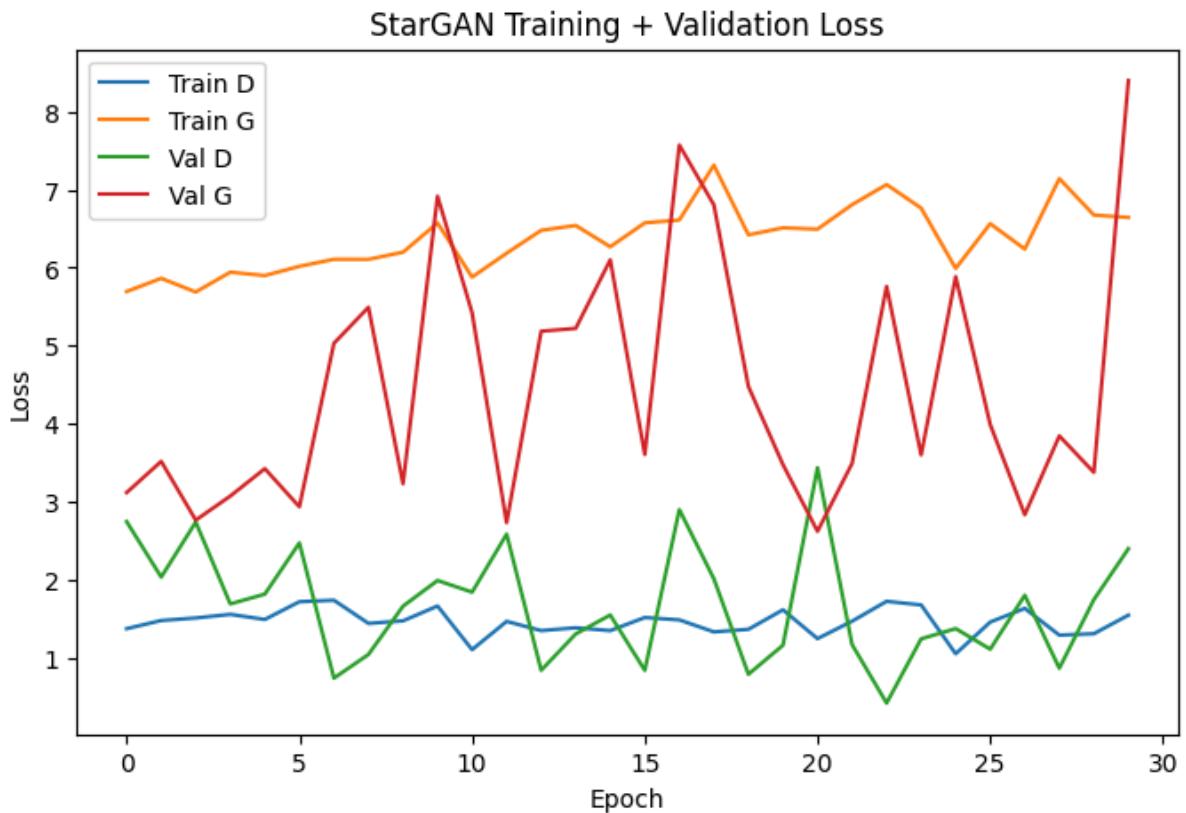


Image 21 : Graphique de perte du discriminateur et générateur pendant l'entraînement pour StarGAN.

### Analyse des courbes :

Les pertes du générateur (Train G et Val G) présentent une tendance à fluctuer tout au long de l'entraînement, une instabilité typique des GANs en raison de la compétition entre le générateur et le discriminateur. Cependant, les fluctuations de la perte de validation (Val G) sont plus marquées, ce qui indique que le modèle a du mal à généraliser sur les données de validation, probablement en raison de la taille limitée du dataset. Pour la perte du discriminateur (Train D et Val D), les courbes révèlent une alternance constante entre augmentation et diminution, reflétant les ajustements dynamiques entre les deux modèles. Une différence notable entre les pertes d'entraînement (Train D) et de validation (Val D) est visible, suggérant que le

discriminateur sur-apprend légèrement sur l'ensemble d'entraînement. Vers la fin des 30 époques, une légère stabilisation des pertes est perceptible, bien que des oscillations persistent. Cette tendance montre que le modèle atteint progressivement un équilibre, mais reste influencé par la nature compétitive des GANs et les limites du dataset.

#### 4.2.2) Résultats et Tests

##### Comparaison entre Source, Généré et Vrai (Cible)

Pour évaluer les performances de StarGAN, nous avons testé le modèle sur des images provenant du domaine source (photos). Les images ont été converties vers le domaine cible (peintures de Monet) à l'aide du générateur. Nous avons également inclus de vraies peintures pour comparaison. Voici un aperçu des résultats :

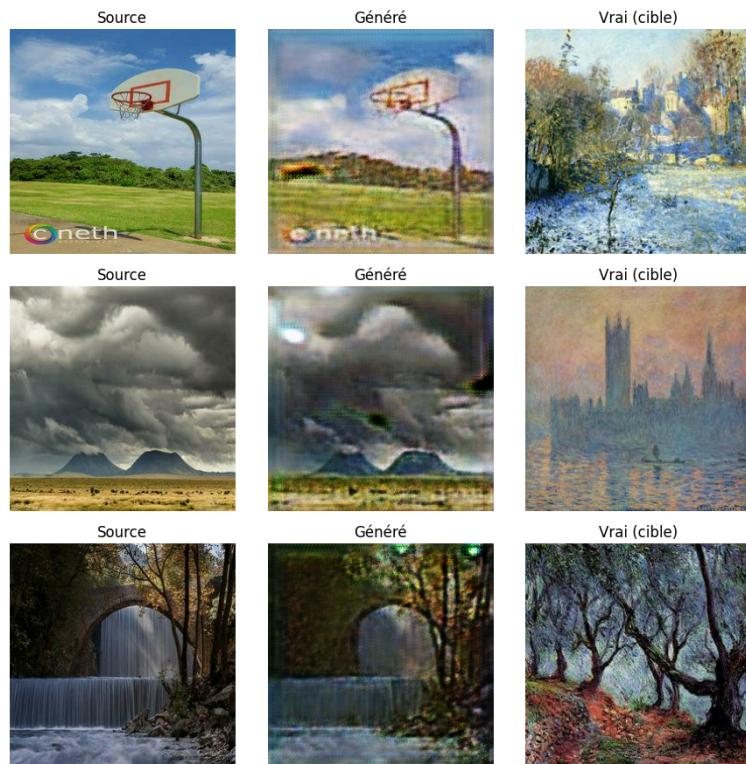


Image 22: Visualisation comparaison entre source, généré et cible.

- **Source** : Les images initiales montrent des photos réalistes issues du domaine source.
- **Généré** : Les images traduites révèlent un transfert de style impressionniste, avec des couleurs et des textures qui évoquent le style de Monet. Cependant, certains détails originaux sont parfois altérés ou brouillés.
- **Vrai (Cible)** : Les vraies peintures de Monet servent de référence, mettant en évidence les nuances et la complexité que le modèle cherche à imiter.

## Résultats du Discriminateur

Le discriminateur a été testé sur trois images issues de notre dataset, comprenant à la fois des images réelles et des images générées. Voici une analyse des résultats obtenus :



Image 23: Visualisation image Fake/Real pour StarGAN.

### Logits (vrai/faux) :

Les logits indiquent si une image est considérée comme réelle (valeurs positives élevées) ou générée (valeurs négatives). Voici les résultats :

- **Image 1 (Vraie)** : **8.053648** → Cette valeur élevée indique que le discriminateur a correctement identifié l'image comme réelle.
- **Image 2 (Générée)** : **-3.7512021** → La valeur négative confirme que le modèle a bien détecté cette image comme générée.
- **Image 3 (Vraie)** : **7.7942648** → Une autre valeur élevée montrant que l'image a été reconnue comme authentique.

**Interprétation** : Le discriminateur distingue efficacement les vraies images des images générées, avec des logits bien séparés.

### Prédition de Domaine (Photo vs. Monet) :

Les prédictions de domaine permettent de classifier une image comme appartenant au domaine "Photo" ou "Monet". Les résultats sont présentés sous forme de logits pour chaque domaine, où la valeur la plus élevée indique l'appartenance.

- **Image 1 (Vraie peinture)** : [5.883867, -4.0034165] → Le modèle identifie correctement cette image comme appartenant au domaine "Monet", grâce au logit positif dominant pour ce domaine.
- **Image 2 (Générée)** : [-3.6657493, -1.5773065] → Les valeurs négatives indiquent une hésitation du modèle à classifier cette image, mais montre une valeur plus forte au domaine de Monet.
- **Image 3 (Vraie peinture)** : [-3.7442582, 3.667882] → Le modèle classe correctement cette image comme Monet, avec un logit positif dominant pour ce domaine.

#### 4.2.3) Interprétation des Résultats

Le générateur démontre sa capacité à traduire des photos en peintures en incorporant des éléments stylistiques caractéristiques du domaine cible, notamment le style de Monet. Cependant, des artefacts visuels tels que des taches ou des zones floues apparaissent sur certaines images traduites, ce qui indique que le modèle n'a pas encore atteint une traduction totalement aboutie. Ces limitations pourraient être dues à un entraînement sur un dataset restreint ou à des ressources limitées. Le discriminateur, quant à lui, réussit à distinguer efficacement les vraies images des images générées, comme en témoignent les logits clairement différenciés. De plus, la prédition des domaines est généralement précise, bien qu'un léger manque de confiance soit observé dans certains cas, particulièrement pour les images générées. Cela reflète une marge d'amélioration dans la capacité du modèle à généraliser sur des exemples complexes.

## 5.0) Conclusion

À travers ce projet, nous avons pu explorer différentes approches pour transformer des photographies de paysages en peintures dans le style de Monet, en nous appuyant sur des modèles GAN. L'étude préliminaire des données nous a permis de dégager des indicateurs clés (variance, contraste et intensité) afin de qualifier objectivement l'écart entre une photo et une peinture impressionniste. Ces mêmes métriques ont ensuite servi de référence pour évaluer la pertinence des images générées.

Dans un premier temps, le GAN « classique » nous a offert une solide base de compréhension du fonctionnement des réseaux antagonistes : le générateur et le discriminateur s'améliorent mutuellement via un apprentissage dit « en ping-pong ». Malgré quelques difficultés de stabilisation (le discriminateur apprenant parfois plus vite que le générateur), les ajustements d'hyperparamètres, comme la réduction du taux d'apprentissage pour le discriminateur, ont permis d'obtenir des résultats visuellement cohérents. Les mesures de contraste, de variance et d'intensité des images générées se sont rapprochées de celles des véritables peintures, validant le bon fonctionnement du modèle pour ce type de tâche.

Ensuite, l'exploration de modèles plus sophistiqués tels que StyleGAN et StarGAN a mis en avant le potentiel d'une meilleure adaptation du style et d'une traduction multi-domaines (photo  $\leftrightarrow$  Monet). StyleGAN, avec son approche de contrôle par styles, a offert une granularité plus fine pour générer des détails, tandis que StarGAN a démontré la capacité à gérer simultanément plusieurs domaines de conversion au sein d'un seul réseau. Cependant, la complexité de ces modèles et les contraintes liées à la taille du jeu de données (faible nombre d'images de Monet) ont occasionné des artefacts et des difficultés de convergence plus marquées.

En définitive, nos expériences ont montré que, même avec un ensemble de données limité, il est possible de produire des images convaincantes évoquant le style de Monet. Les indicateurs visuels et statistiques confirment que les GAN, lorsqu'ils sont correctement entraînés et ajustés, parviennent à générer des peintures factices dont les caractéristiques se rapprochent sensiblement de l'œuvre originale. Pour aller plus loin, il serait intéressant d'enrichir le jeu de données, d'approfondir le réglage des hyperparamètres et d'explorer des variantes récentes de GAN qui gèrent mieux la stabilité de l'apprentissage et la diversité des images générées.

## 5.1) Bibliographie

Lien vers nos données :

[https://drive.google.com/drive/folders/1Sqs5M3OPrOllbebYI04\\_G-NsM51LvDtj?usp=drive\\_link](https://drive.google.com/drive/folders/1Sqs5M3OPrOllbebYI04_G-NsM51LvDtj?usp=drive_link)

Lien vers notre github : [https://github.com/MerlierUgo/IA\\_Application](https://github.com/MerlierUgo/IA_Application)

Lien vers le challenge kaggle : <https://www.kaggle.com/competitions/gan-getting-started>