République du Sénégal
Un Peuple – Un But – Une Foi
Ministère l'enseignement supérieur de la recherche et de l'innovation
Ecole Polytechnique de Thiès
B.P.A 10 Thiès
Tel: (221) 76 223 61 74 – Fax: (221) 33 951 14 67

Rapport de Projet Traitement d'images et Vision par Ordinateur -2023

Super Résolution appliquée à la stylisation et au diagnostic des tumeurs du cerveau

Membres du groupe :	Professeur :
• El Hadj Dame Lo Kaba • Mohamed Massamba Sene • Oumou Khaïry Sougou	Dr. Ndeye Fatou Ngom

Table des matières

Introduction

Ce rapport de projet vise à présenter les résultats de la super-résolution d'images et son application au neural style transfer et au diagnostic des tumeurs du cerveau. La super-résolution d'images est une technique de traitement d'images qui vise à augmenter la résolution et les détails des images à faible résolution pour produire des images de haute qualité. Cette technique trouve de plus en plus d'applications dans divers domaines, notamment dans le domaine artistique et médical.

Dans ce projet, nous avons exploré deux applications spécifiques de la super-résolution d'images. Tout d'abord, nous avons étudié le neural style transfer, une technique de traitement d'image qui permet de transférer le style artistique d'une image à une autre image. Nous avons examiné comment la super-résolution d'images peut être utilisée pour améliorer la qualité des images générées par cette technique.

Ensuite, nous avons examiné l'application de la super-résolution d'images dans le domaine médical, en particulier pour le diagnostic des tumeurs du cerveau. Nous avons étudié comment les techniques de super-résolution d'images peuvent être utilisées pour améliorer la qualité des images médicales et faciliter la détection des tumeurs du cerveau.

I. Super Résolution

1. Présentation Générale

Dans de nombreux films policiers, nous pouvons voir des personnes avec une image floue et obtenir une image claire en zoomant cependant cela est techniquement inexact car le traitement des données ne peut pas ajouter de contenu informatif.
Mais il existe un processus de mise à l'échelle et d'amélioration des images appelé super résolution. En effet, en utilisant des réseaux de neurones, nous pouvons halluciner des détails basés sur des données antérieures collectées à partir d'un grand nombre d'images. L'objectif étant que le réseau apprenne une cartographie entre les images basse et haute résolution.
Les premiers modèles de super-résolution basés sur le deep learning ont été introduits en 2016 avec le modèle SRCNN (Super Resolution Convolutional Neural Network), suivi par d'autres modèles tels que le SRGAN (Super Resolution Generative Adversarial Network) en 2017.
Notre objectif est donc de créer un modèle de super résolution d'images basées sur les SRGAN qui permettra d'améliorer la qualité des images.

- Aperçu sur les GAN

Créees par Ian Goodfellow en 2014, directeur du département R&D de Apple en deep learning jusqu'en 2022, un GAN est un type de réseau de neurones artificiels qui est utilisé pour générer des données synthétiques, telles que des images, des vidéos ou des sons, qui ressemblent à des données réelles. Le GAN est composé de deux parties : le générateur et le discriminateur. Le générateur génère des données synthétiques, tandis que le discriminateur évalue la qualité de ces données en les comparant à des données réelles. La formation d'un GAN se fait par un processus d'entraînement compétitif, où le générateur et le discriminateur s'affrontent dans un jeu du chat et de la souris (ou policier et faussaire comme l'indique la *Figure 1)*. Le générateur tente de générer des données synthétiques qui trompent le discriminateur en lui faisant croire qu'elles sont réelles, tandis que le discriminateur tente de distinguer les données synthétiques des données réelles.
Le processus d'entraînement commence par l'initialisation aléatoire des paramètres du générateur et du discriminateur. Le générateur prend en entrée un vecteur de bruit aléatoire et génère une image synthétique, tandis que le discriminateur prend en entrée une image réelle ou synthétique et retourne une valeur comprise entre 0 et 1, qui représente la probabilité que l'image soit réelle.
L'entraînement se poursuit en alternant les phases de formation du générateur et du discriminateur. Lors de la phase de formation du discriminateur, le discriminateur est entraîné à distinguer les images réelles des images synthétiques générées par le générateur. Lors de la phase de formation du générateur, le générateur est entraîné à générer des images synthétiques qui ressemblent le plus possible aux images réelles, en trompant le discriminateur.
Le processus d'entraînement se termine lorsque le générateur ne peut plus être amélioré et lorsque le discriminateur ne peut plus distinguer les données synthétiques des données réelles de manière fiable. À ce stade, le générateur peut être utilisé pour générer de nouvelles données synthétiques qui ressemblent à des

données réelles, ce qui peut être utile pour diverses applications telles que la synthèse de données manquantes ou la génération de contenus créatifs tels que des images ou des vidéos.

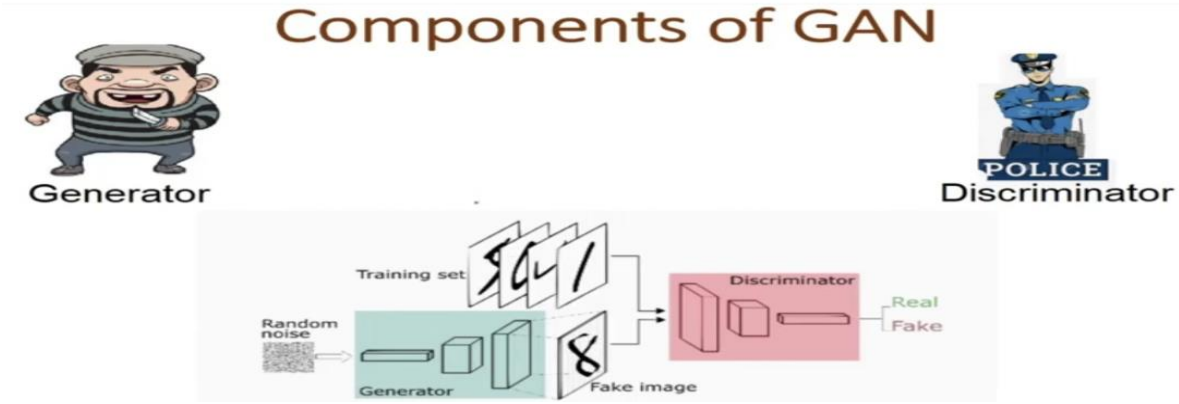


Figure 1 : Illustration du fonctionnement d'un GAN

Comme nous le voyons dans la *Figure 1*, l'objectif est que le faussaire (générateur) arrive à tromper le policier (discriminateur) en fabriquant des faux billets et au fur et à mesure qu'il se fait attraper il retourne peaufiner ses faux billets jusqu'à être un faussaire irréprochable ainsi le policier le détectera avec plus de mal.

2. Dataset Utilisé

Le dataset utilisé est Flickr2K qui est utilisé pour la restauration d'images. Il est constitué de 2560 images de résolution 384x384.
C'est un ensemble de données utilisé pour les tâches de super-résolution d'images, en particulier pour le développement et l'évaluation de modèles d'apprentissage en profondeur. Les images proviennent de Flickr et couvrent un large éventail de scènes et de sujets, notamment la nature, l'architecture et les personnes.

3. Etapes de l'implémentation

Diminution de la résolution et pixélisation

Le jeu de données étant composé d'images haute résolution de taille 384x384, nous avons d'abord commencé par diminuer la résolution spatiale et tonale des images. Ceci nous permettra d'avoir les images basses résolutions et leur équivalent haute résolution qui seront utilisés pour entrainer notre modèle.

- **Premier sous échantillonnage avec Lancsoz**

Le sous-échantillonnage Lanczos est une technique utilisée pour la réduction d'échelle d'image, qui consiste à appliquer un filtre Lanczos à l'image avant de la rééchantillonner à une taille plus petite. Ce filtre utilise une fonction sinc fenêtrée pour interpoler en douceur les données d'image et réduire les artefacts de crénelage qui peuvent se produire lors du rééchantillonnage.
En utilisant une méthode de sous-échantillonnage de haute qualité comme Lanczos, l'image de résolution inférieure résultante aura moins d'artefacts et plus de détails que si une méthode de sous-échantillonnage plus simple comme le plus proche voisin ou l'interpolation bilinéaire était utilisée. Ceci aide à conserver plus de détails au niveau de l'image avant d'appliquer l'interpolation bicubique.

Figure 2 : Image originale et Image obtenue après sous échantillonnage Lancsoz

- **Deuxième sous échantillonnage avec interpolation bicubique**

On applique un second sous échantillonnage afin d'introduire un effet de pixélisation due au fait que l'image devient trop petite pour représenter correctement les détails fins de l'image.
L'interpolation bicubique permet d'essayer de calculer les valeurs pixels manquantes en utilisant une fonction mathématique qui prend en compte les valeurs des pixels voisins. Cette technique permet de remplacer les pixels manquants par des valeurs plus précises qui sont calculées en fonction de leur voisinage. Cela permet d'améliorer la qualité de l'image sous-échantillonnée et facilite la récupération des détails fins lors de la super-résolution.

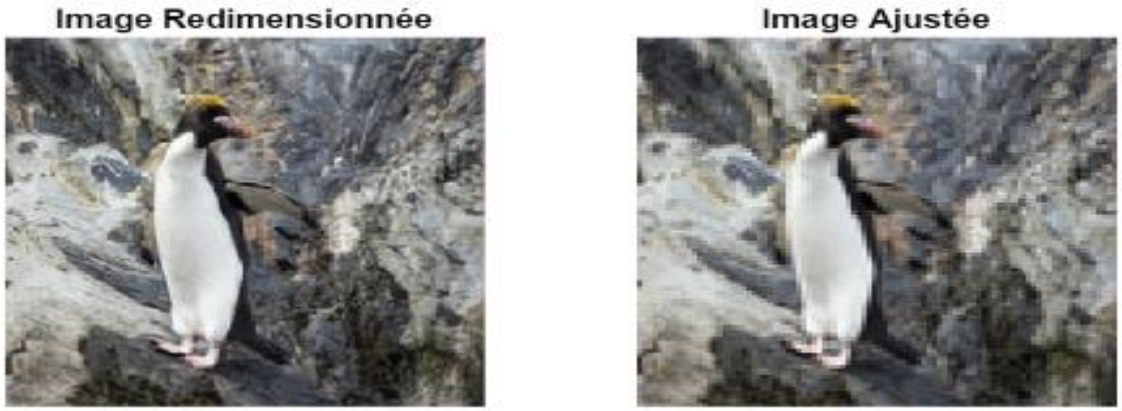


Figure 3 : Image obtenue après Lancsoz et Image obtenue après sous échantillonnage par interpolation bicubique

Après ceci nous obtenons donc une image basse résolution à partir de l'image haute résolution que nous avions au départ. Ceci nous permet de créer le jeu de données qui sera utilisé pour permettre au modèle d'apprendre une cartographie entre les images haute résolution et les images basse résolution.

Modèle utilisé

Nous utilisons donc un modèle basé sur les SRGAN qui sont eux même un type de GAN et comprend donc deux parties :

- **Générateur**

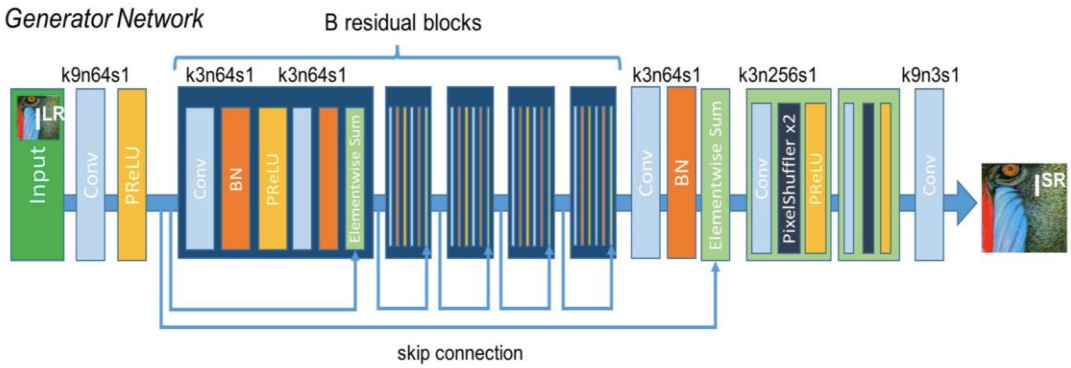


Figure 4 : Architecture du générateur

L'image basse résolution est l'input du Générateur qui après une couche de convolution de kernel size 3x3, se faisant sur l'ensemble de l'image comme nous l'indique le stride qui est égale à 1 et 64 features maps en seront extraites. Lorsque l'on veut un réseau de neurones puissant on augmente le nombre de couches c'est à dire sa profondeur mais ceci s'avérait être efficace jusqu'à un moment où on se rende compte que cela entraine le « vanishing gradient » qui n'est rien d'autre que quand on applique l'algorithme de back propagation pour ajuster les paramètres de nos modèles , l'intensité de notre gradient est multiplié par le poids synaptique entre les couches , voilà qui fait que cette intensité est démultipliée encore et encore ce qui rend la mise à jour des poids inefficace et entraîne une convergence lente. Les ResNets (blocs bleus du model) résolvent ce problème en utilisant des connexions résiduelles (skip connections) qui permettent aux gradients de sauter directement d'une couche à une autre plutôt que de devoir traverser de nombreuses couches intermédiaires. Pour le modèle d'implémentation B=16 d'après les papiers de recherche du docteur Ian Goodfellow. Nous utilisons cependant 5 couches dans notre projet afin d'alléger la complexité du modèle. Ensuite après une couche de convolution et une batch normalization qui sert à accélérer l'entrainement en normalisant la distribution d'entrée qu'elle reçoit nous sommons la sortie avec l'input de départ de cette convolution.

Nous entrons ensuite dans les upscale layers dont la particularité est le Pixel Shuffler qui consiste à réarranger les pixels d'un tenseur en étalant tous les pixels de la profondeur sur la base (H, W, C) augmentant ainsi la résolution spatiale de l'image d'où sa résolution rH x rW x C.

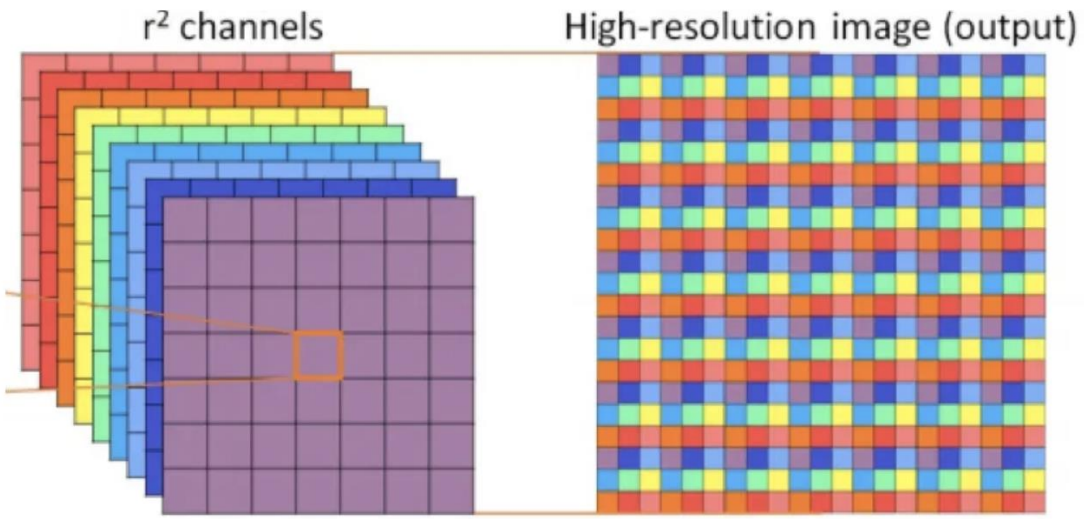


Figure 5 : Illustration du Pixel Shuffler

Le Pixel Shuffler est une technique de deep learning utilisée dans la super résolution d'images. Il est souvent utilisé dans les architectures de type GAN, telles que le SRGAN.

L'algorithme Pixel Shuffler est basé sur la manipulation des canaux de couleurs d'une image, qui sont généralement représentés sous forme de matrices 3D. L'idée principale est de réorganiser les pixels dans les canaux de couleur de l'image de manière à augmenter la résolution spatiale de l'image. Pour ce faire le PixelShuffler utilise l'algorithme de transposition fractionnée également connue sous le nom de sous-pixel convolution ou de convolution transposée, est une technique utilisée dans les réseaux de neurones pour réaliser une interpolation de l'image.

En général, l'opération de convolution réduit la dimension de l'image, car elle produit une carte de caractéristiques avec une taille plus petite que l'image d'entrée. La transposition fractionnée effectue le processus inverse, en augmentant la dimension de la carte de caractéristiques à la taille de l'image d'origine.

L'idée principale de la transposition fractionnée est de prendre chaque pixel de la carte de caractéristiques de l'image et de le transformer en un bloc de pixels qui correspond à la taille de la nouvelle image. Pour ce faire, la convolution transposée utilise une matrice de poids qui est similaire à celle de la convolution classique, mais qui est inversée.

Pour chaque pixel de la carte de caractéristiques, la transposition fractionnée calcule la somme pondérée des pixels adjacents dans l'image de sortie, en utilisant les poids de la matrice de convolution transposée. Ces sommes pondérées sont ensuite placées dans les pixels correspondants de l'image de sortie, créant ainsi une nouvelle image avec une résolution plus élevée.

Le principal avantage de la transposition fractionnée est qu'elle permet d'augmenter la résolution d'une image tout en préservant les détails et les caractéristiques de l'image d'origine. C'est pour cette raison que la transposition fractionnée est souvent utilisée dans les architectures de réseaux de neurones pour des tâches de traitement d'images telles que la super-résolution.

- **Discriminateur**

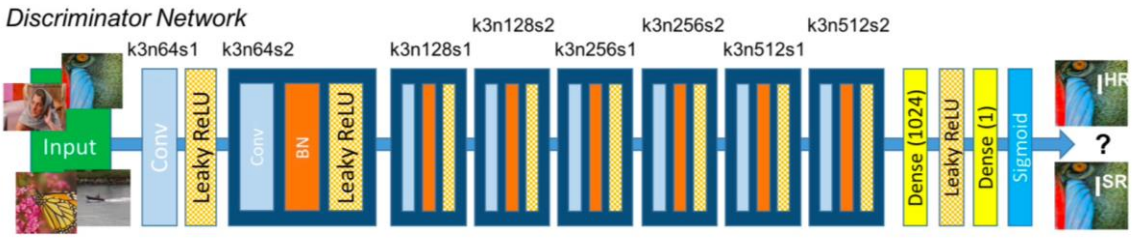


Figure 6 : Architecture du discriminateur

L'architecture d'un discriminateur dans un SRGAN est basée sur un réseau de neurones convolutionnels (CNN) profond, qui prend en entrée une image haute résolution (HR) et doit décider si elle est réelle (HR) ou générée (SR). Le discriminateur est entraîné avec une loss function d'adversité pour distinguer les images réelles des images générées par le générateur.

L'architecture du discriminateur est généralement composée de plusieurs couches convolutionnelles suivies de fonctions d'activation non linéaires telles que LeakyReLU ou ReLU. Les couches convolutionnelles sont utilisées pour extraire les caractéristiques de l'image d'entrée et les fonctions d'activation sont utilisées pour introduire des non-linéarités dans le modèle.

La plupart des architectures de discriminateurs dans les SRGAN utilisent des couches de convolution avec des kernels de taille 3x3 et un stride de 2, ce qui réduit la taille de l'image de moitié à chaque couche. Cela permet au modèle d'apprendre des caractéristiques à différentes échelles spatiales. Les couches convolutives sont souvent suivies de couches de batch normalization qui normalisent les activations de chaque couche pour améliorer la stabilité de l'entraînement et accélérer la convergence. L'architecture du discriminateur se termine généralement par une couche dense avec une activation sigmoid qui produit une sortie entre 0 et 1. Cette sortie représente la probabilité que l'image en entrée soit réelle ou générée.

La particularité des SRGAN est cependant leur fonction coût qui comme dans tout problème de machine learning nous cherchons à minimiser

$$l^{SR} = \underbrace{l_X^{SR}}_{\text{content loss}} + \underbrace{10^{-3} l_{Gen}^{SR}}_{\text{adversarial loss}}$$

Figure 7 : Fonction coût du SRGAN

C'est une combinaison linéaire de la fonction cout du générateur et du discriminateur.

La fonction cout du discriminateur est une négative log-likelihood loss qui calcule la probabilité que l'image générée soit réelle ou non.

$$l_{Gen}^{SR} = \sum_{n=1}^{N} -log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

Here, $G_{\theta G}(I^{LR})$ is the generated image and $D_{\theta D}(G_{\theta G}(I^{LR}))$ represents the probability that the generated image is a real image

Figure 8 : Fonction coût du discriminateur

La loss function du discriminateur est une Mean Square Error sur l'image en 2d qui calcule la différence entre l'image haute résolution et l'image générée

$$l_{MSE}^{SR} = \frac{1}{r^2 WH} \sum_{x=1}^{rW} \sum_{y=1}^{rH} (I_{x,y}^{HR} - G_{\theta_G}(I^{LR})_{x,y})^2$$

Figure 9 : Fonction coût du générateur

Entrainement du modèle

On entraine sur 100 epochs avec un batch size de 16 le modèle sur le train set en utilisant le val set pour comme jeu de validation.

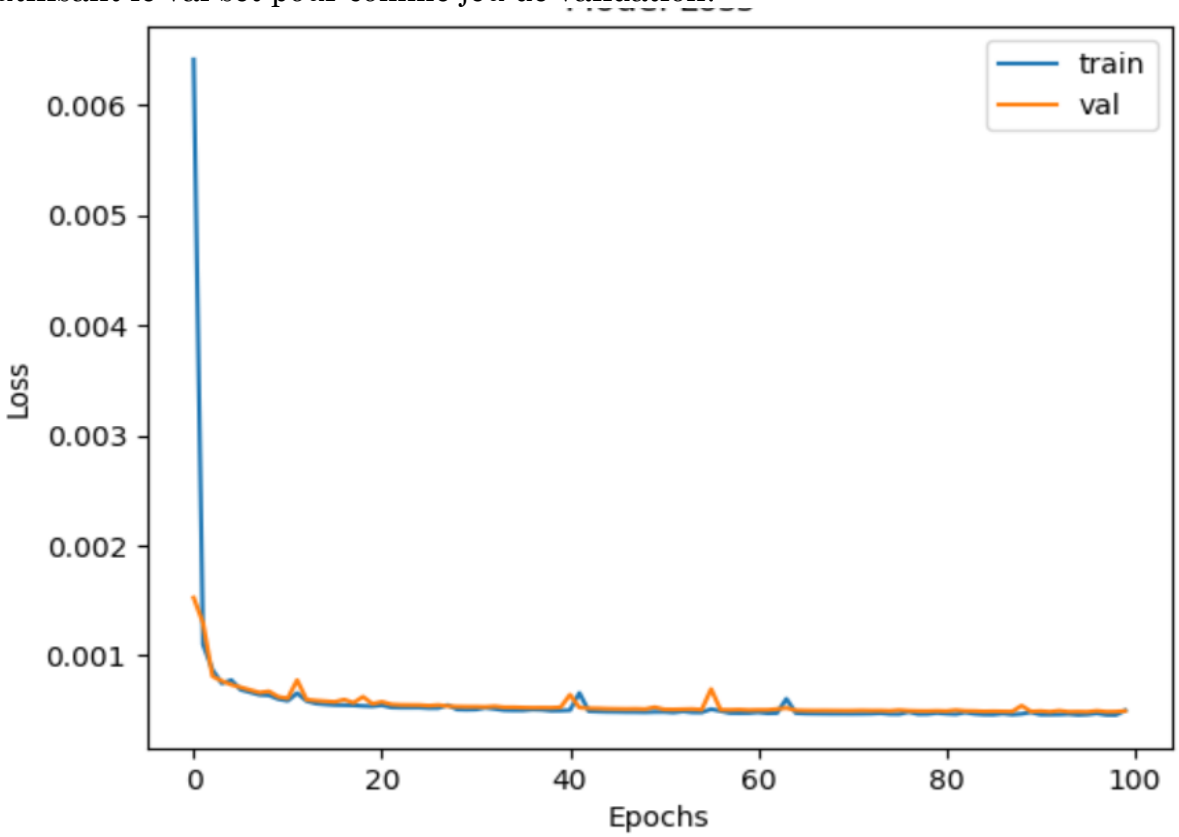


Figure 10 : Variation de la fonction coût du modèle

On constate donc après entrainement que l'on obtient effectivement une baisse de la fonction coût qui passe de 0.0065 et 0.0015 pour les jeux d'entrainement et de validation à moins près de 0.0005 pour les deux.

4. Analyse des résultats

Figure 11 : Résultat de la super résolution sur des images basse résolution du jeu de données test

Nous pouvons donc constater qu'après super résolution, nous constatons effectivement une augmentation de la netteté de l'image ainsi qu'une meilleure visibilité des détails au niveau de l'image.

La distance initiale de Fréchet (FID) est une métrique utilisée pour évaluer la qualité des images créées par un modèle génératif, comme un réseau antagoniste génératif (GAN). Le FID compare la distribution des images générées à la distribution d'un ensemble d'images réelles ("ground Truth").

La métrique FID a été introduite en 2017 et est la métrique standard actuelle pour évaluer la qualité des modèles génératifs à partir de 2020.

Elle a été utilisée pour mesurer la qualité de nombreux modèles récents, notamment le StyleGAN1 haute résolution et Réseaux StyleGAN2.

```
FID (Fréchet Incpetion Distance) Obtenue: 0.06415592287133852
```

Method	*Dataset (n. Images)*	*FID score ↓*
GAN [48]	CelebA-HQ (50K)	7.30
NVAE [60]	CelebA-HQ (50K)	40.26
PCA [61]	Ours (9358)	65.80
VAE [59]	Ours (9358)	120.44
NVAE [60]	Ours (9358)	62.98
GAN (Ours) [48]	Ours (9358)	**2.95**

Figure 11 : FID du modèle de SRGAN ainsi que celle des GAN classiques

Nous voyons donc que notre modèle nous donne un FID d'environ 0.064 ce qui montre que la qualité des images générées est effectivement très proche des images haute résolution du dataset donc les performances de notre modèle sont satisfaisantes. En comparant ce résultat à des résultats obtenus avec les GAN classiques, nous voyons que notre modèle de SRGAN performe mieux.

```
Avg. PSNR des lowres images: 30.7003
Avg. PSNR des images super résolues: 33.6997
```

Figure 12 : PSNR des images super résolues vs PSNR des images initiales

En utilisant le PSNR nous pouvons également voir que la super résolution entraine une augmentation de 3 (30 à 33) à ce niveau sur les images super résolues par rapport aux images basses résolution que nous avions initialement.

II. Matlab : Super Résolution appliquée au Neural Style Transfer

1. Workflow de l'application

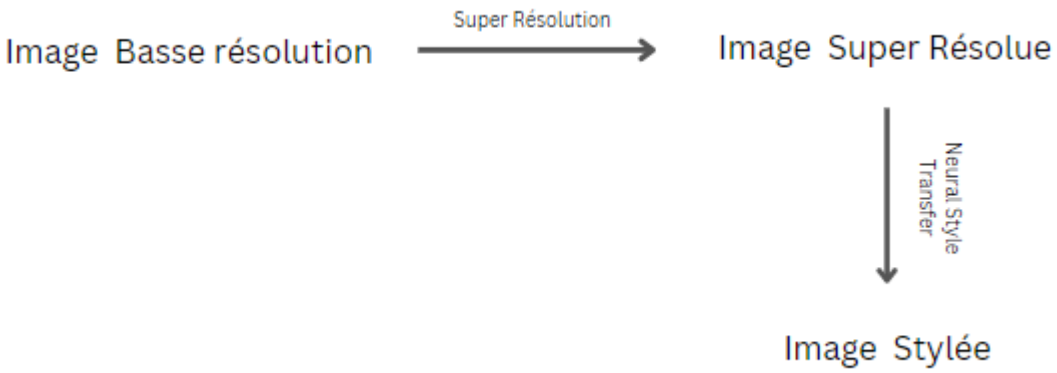


Figure 13 : Workflow de l'application de super résolution appliquée au NST

L'architecture du modèle se présente comme suit. Partant d'une image basse résolution et en utilisant la méthode de super résolution explicitée dans la partie précédente, on obtient une image super résolue. A cette dernière image, on lui appliquera le Neural Style Transfer, pour obtenir à la fin une Image avec du style ou image stylée.

2. Neural Style Transfer

a. Présentation Générale

Dans le domaine des beaux-arts, et plus particulièrement de la peinture, l'homme est passé maître dans l'art de créer des expériences visuelles uniques en composant une interaction complexe entre le contenu et le style d'une image. Jusqu'à présent, la base algorithmique de ce processus est inconnue et il n'existe pas de système artificiel doté de capacités similaires. Cependant, dans d'autres domaines clés de la perception visuelle tels que la reconnaissance des objets et des visages, des performances quasi humaines ont récemment été démontrées par une classe de modèles de vision d'inspiration biologique appelés réseaux neuronaux profonds. Nous présentons ici un système artificiel basé sur un réseau neuronal profond qui crée des images artistiques d'une grande qualité perceptive. Le système utilise des représentations neuronales pour séparer et recombiner le contenu et le style d'images arbitraires, fournissant ainsi un algorithme neuronal pour la création d'images artistiques. Notre travail s'inspire du papier de recherche nommé **A Neural Algorithm of Artistic Style** des auteurs **Leon A. Gatys, Alexander S. Ecker, Matthias Bethge**.
L'application que nous avons eu à implémenter met en œuvre le Neural Style Transfer ou transfert de style qui consiste à combiner le style d'une seule image avec le contenu d'une autre image pour générer une nouvelle image avec le style de la première et le contenu de la deuxième image.

b. Etapes de l'implémentation

Méthode utilisée

La classe de réseaux neuronaux profonds la plus puissante pour les tâches de traitement d'images s'appelle les réseaux neuronaux convolutifs. Les réseaux neuronaux convolutifs sont constitués de couches de petites unités de calcul qui traitent les informations visuelles de manière hiérarchique, selon une méthode d'anticipation. Chaque couche d'unités peut être considérée comme un ensemble de filtres d'image, dont chacun extrait une certaine caractéristique de l'image d'entrée. Ainsi, la sortie d'une couche donnée consiste en ce que l'on appelle des cartes de caractéristiques ou features maps : des versions filtrées différemment de l'image d'entrée. Lorsque les réseaux neuronaux convolutifs sont entraînés à la reconnaissance d'objets, ils développent une représentation de l'image qui rend l'information sur l'objet de plus en plus explicite le long de la hiérarchie de traitement. Par conséquent, le long de la hiérarchie de traitement du réseau, l'image d'entrée est transformée en représentations qui se soucient de plus en plus du contenu réel de l'image par rapport à ses valeurs de pixels détaillées. Nous pouvons visualiser directement les informations que chaque couche contient sur l'image d'entrée en reconstruisant l'image uniquement à partir des cartes de caractéristiques (features maps) de cette couche. Les couches supérieures du réseau capturent le contenu de haut niveau en termes d'objets et de leur disposition dans l'image d'entrée, mais ne contraignent pas les valeurs exactes des pixels de la reconstruction. En revanche, la reconstruction des couches inférieures reproduisent simplement les valeurs exactes des pixels de l'image originale.

Nous désignons donc les réponses aux caractéristiques (features responses) dans les couches supérieures du réseau comme la représentation du contenu. Pour obtenir une représentation du style d'une image d'entrée, nous utilisons un espace de caractéristiques conçu à l'origine pour capturer des informations sur la texture. Cet espace de caractéristiques est situé au-dessus des réponses des filtres dans chaque couche du réseau. Il se compose des corrélations entre les différentes réponses des filtres sur l'étendue spatiale des cartes de caractéristiques (features maps).

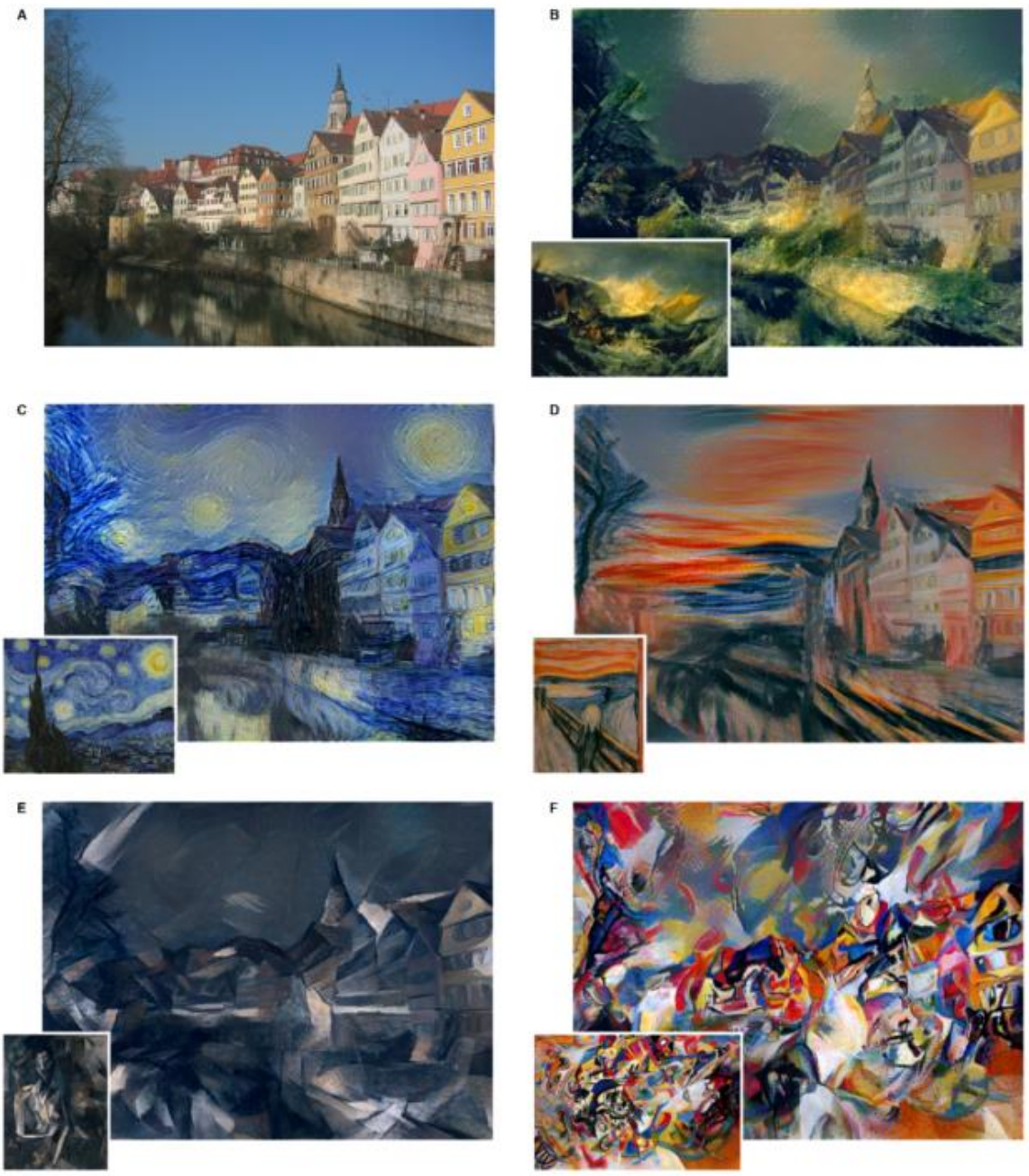


Figure 14 : Images combinant le contenu d'une photographie et le style de plusieurs œuvres d'art célèbres.

Les images ont été créées en trouvant une image qui correspond simultanément à la représentation du contenu de la photographie et à la représentation du style de l'œuvre d'art :

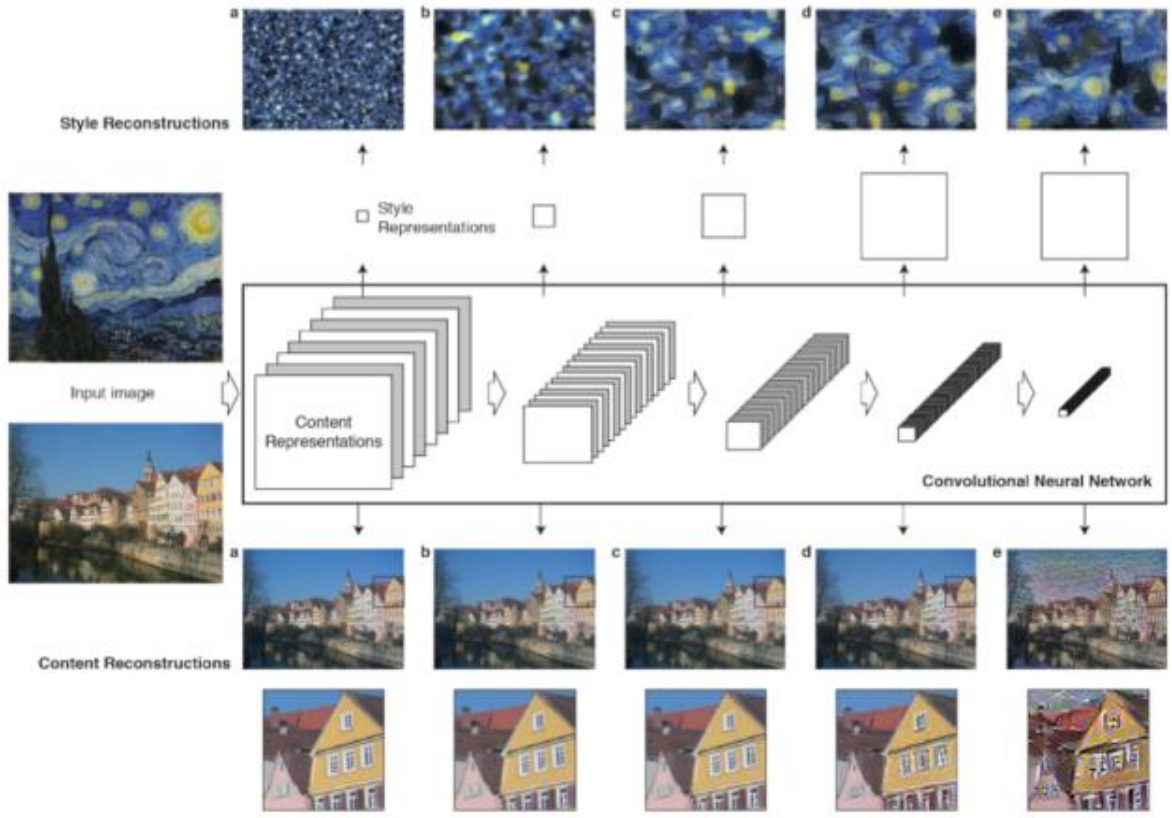


Figure 15 : Réseau de neurones convolutifs pour le Neural Style Transfer

Une image d'entrée donnée est représentée sous la forme d'un ensemble d'images filtrées à chaque étape du traitement dans le CNN. Alors que le nombre de filtres différents augmente le long de la hiérarchie de traitement, la taille des images filtrées est réduite par un mécanisme de sous-échantillonnage (par exemple, max-pooling), ce qui entraîne une diminution du nombre total d'unités par couche du réseau.

- **Reconstruction du contenu**

Nous pouvons visualiser les informations à différents stades de traitement dans le CNN en reconstruisant l'image d'entrée à partir de la seule connaissance des réponses du réseau dans une couche particulière. Nous reconstruisons l'image d'entrée à partir des couches "conv1 1" (a), "conv2 1" (b), "conv3 1" (c), "conv4 1" (d) et "conv5 1" (e) du réseau VGG original. Nous constatons que la reconstruction des couches inférieures est presque parfaite (a, b, c). Dans les couches supérieures du réseau, les informations NST détaillées sur les pixels sont perdues alors que le contenu de haut niveau de l'image est préservé (d,e).

- **Reconstruction stylistique**

En plus des représentations CNN originales, nous avons construit un nouvel espace de caractéristiques qui capture le style d'une image d'entrée. La représentation du style calcule les corrélations entre les différentes caractéristiques dans les différentes couches du CNN. Nous reconstruisons le style de l'image d'entrée à partir des représentations de style construites sur

différents sous-ensembles de couches du CNN ("conv1 1" (a), "conv1 1" et "conv2 1" (b), "conv3 1" (c), "conv4 1" (d) et "conv5 1" (e)). Cela permet de créer des images qui correspondent au style d'une image donnée sur une échelle croissante, tout en ignorant les informations relatives à l'agencement global de la scène.
En général, chaque couche du réseau définit un banc de filtres non linéaires dont la complexité augmente avec la position de la couche dans le réseau. Ainsi, une image d'entrée donnée $\vec{x}$ est codée dans chaque couche du CNN par un banc de filtres non linéaires.
Une couche avec N_l filtres distincts possède N_l cartes de caractéristiques (features maps), chacune de taille M_l est la hauteur multipliée par la largeur de la carte de caractéristiques (features maps).
Nous définissons ensuite la fonction coût de contenu par erreur quadratique entre les deux représentations des caractéristiques
Les réponses d'une couche l peuvent donc être stockées dans une matrice $F_l \in \mathrm{R}$.
Pour visualiser l'information sur l'image qui est encodée à différentes couches de la hiérarchie, nous effectuons une descente de gradient sur une image de bruit blanc pour trouver une autre image qui corresponde aux réponses des caractéristiques de l'image d'origine.
Soit $\vec{p}$ et $\vec{x}$ l'image originale et l'image générée, et P_l et F_l leur représentation respective des caractéristiques dans la couche l. Nous définissons ensuite la perte par erreur quadratique entre les deux représentations des caractéristiques

$$\mathcal{L}_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F_{ij}^l - P_{ij}^l\right)^2$$

Figure 16 : Fonction coût pour le contenu de l'image

La dérivée de cette perte par rapport aux activations de la couche l est égale.

$$\frac{\partial \mathcal{L}_{content}}{\partial F_{ij}^l} = \begin{cases} \left(F^l - P^l\right)_{ij} & \text{if } F_{ij}^l > 0 \\ 0 & \text{if } F_{ij}^l < 0 \,. \end{cases}$$

Figure 17 : Dérivée de la fonction coût contenue

à partir de laquelle le gradient par rapport à l'image $\vec{x}$ peut être calculé à l'aide d'une rétropropagation à erreur standard. Nous pouvons donc modifier l'image initialement aléatoire $\vec{x}$ jusqu'à ce qu'elle génère la même réponse dans une certaine couche du CNN que l'image originale $\vec{p}$ Les cinq reconstructions de contenu de la figure 1 proviennent des couches "conv1 1" (a), "conv2 1" (b), "conv3 1" (c), "conv4 1" (d) et "conv5 1" (e) du réseau VGG original.
Au-dessus des réponses CNN dans chaque couche du réseau, nous avons construit une représentation de style qui calcule les corrélations entre les différentes réponses de filtre, où l'espérance est prise sur l'extension spatiale de

l'image d'entrée. Ces corrélations sont données par la matrice de Gram $G^l \in \mathrm{R}$ où G^l_{ij} est le produit intérieur entre la carte vectorielle des caractéristiques (features maps vectorisées) i et j dans la couche l :

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk}.$$

Figure 18 : Corrélation entre les différentes réponses des filtres

Pour générer une texture correspondant au style d'une image donnée, nous utilisons la descente de gradient à partir d'une image de bruit blanc pour trouver une autre image correspondant à la représentation du style de l'image originale. Pour ce faire, nous minimisons la distance moyenne quadratique entre les entrées de la matrice de Gram de l'image originale et la matrice de Gram de l'image à générer. Soit $\vec{a}$ et $\vec{x}$ l'image originale et l'image générée et A^l et G^l leurs représentations de style respectives dans la couche l. La contribution de cette couche à la perte totale est alors la suivante :

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G^l_{ij} - A^l_{ij}\right)^2$$

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l$$

Figure 19 : Contribution à la perte totale et perte totale

où w_l sont des facteurs de pondération de la contribution de chaque couche à la perte totale. La dérivée de E_l par rapport aux activations de la couche l peut être calculée analytiquement :

$$\frac{\partial E_l}{\partial F_{ij}^l} = \begin{cases} \frac{1}{N_l^2 M_l^2} \left((F^l)^{\mathrm{T}} \left(G^l - A^l \right) \right)_{ji} & \text{if } F_{ij}^l > 0 \\ 0 & \text{if } F_{ij}^l < 0 \,. \end{cases}$$

Figure 20 : Dérivée par rapport aux activations de la couche l

Les gradients de E_l par rapport aux activations des couches inférieures du réseau peuvent être facilement calculés à l'aide de la rétropropagation à erreur standard. Les cinq reconstructions de style de la figure 1 ont été générées en faisant correspondre les représentations de style sur la couche "conv1 1" (a), "conv1 1" et "conv2 1" (b), "conv1 1", "conv2 1" et "conv3 1" (c), "conv1 1", "conv2 1", "conv3 1" et "conv4 1" (d), "conv1 1", "conv2 1", "conv3 1", "conv4 1" et "conv5 1" (e).

Pour générer les images qui mélangent le contenu d'une photographie et le style d'une peinture, nous minimisons conjointement la distance d'une image de bruit blanc par rapport à la représentation du contenu de la photographie dans une couche du réseau et la représentation du style de la peinture dans un certain nombre de couches du CNN. Soit la photographie $\vec{p}$ et $\vec{a}$ l'œuvre d'art. La fonction coût globale que nous minimisons est :

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$

Figure 21 : Fonction coût globale

où α et ß sont les facteurs de pondération pour la reconstruction du contenu et du style respectivement.

Entrainement du modèle

L'entrainement du modèle consiste à passer en entrée du modèle les images de notre modèle et l'image de style afin d'apprendre les features des images à différents niveaux. Notre objectif étant de minimiser la fonction coût globale. Nous entrainons donc le modèle pour 300 itérations.

Figure 22 : Images de Transfer par intervalles de 50 itérations

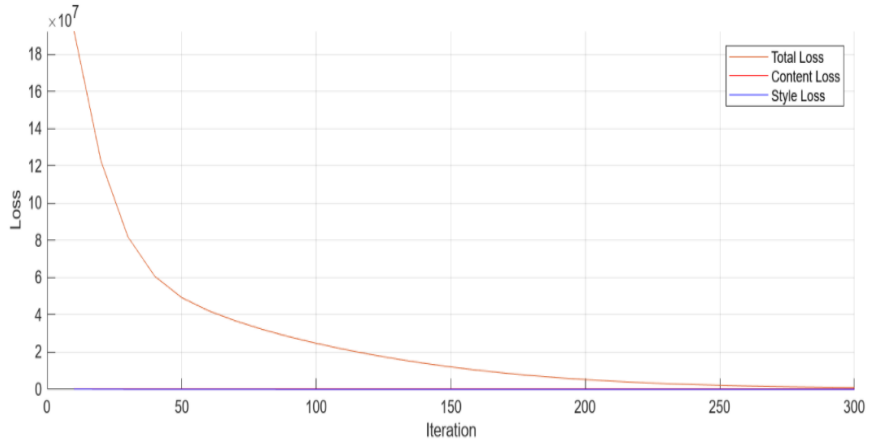


Figure 23 : Variation de la fonction coût

En regardant pour la dernière image du jeu d'entraînement l'image de sortie après chaque itération nous voyons donc qu'à mesure que la fonction coût décroît mieux, l'image de sortie est de plus en plus meilleure en termes de qualité.
A la fin de l'entraînement nous voyons donc que la fonction coût global a été considérablement réduite et l'image obtenue est bien stylisée et contient effectivement le style de l'image initiale et le contenu de l'image de style.

c. Analyse des résultats

Figure 24 : Résultats du Neural Style Transfer sur des images tests super résolues

Nous voyons donc qu'après super résolution, en appliquant le Neural Style Transfer les images obtenues ont effectivement un style artistique similaire à l'image de style tout en conservant le contenu de l'image originale.
La perte de perception (Perceptual Loss) est une métrique couramment utilisée pour évaluer la qualité du NST. Elle mesure la différence entre les caractéristiques de l'image générée et les caractéristiques de l'image cible, généralement à l'aide de réseaux de neurones profonds préformés.
La perte de perception s'est avérée efficace pour produire des résultats visuellement attrayants qui préservent le contenu de l'image d'entrée tout en générant une image de sortie avec un style différent.

```
Avg. Perceptual Loss NST:  0.9460041522979736
```

Figure 25 : Perte de perception moyenne

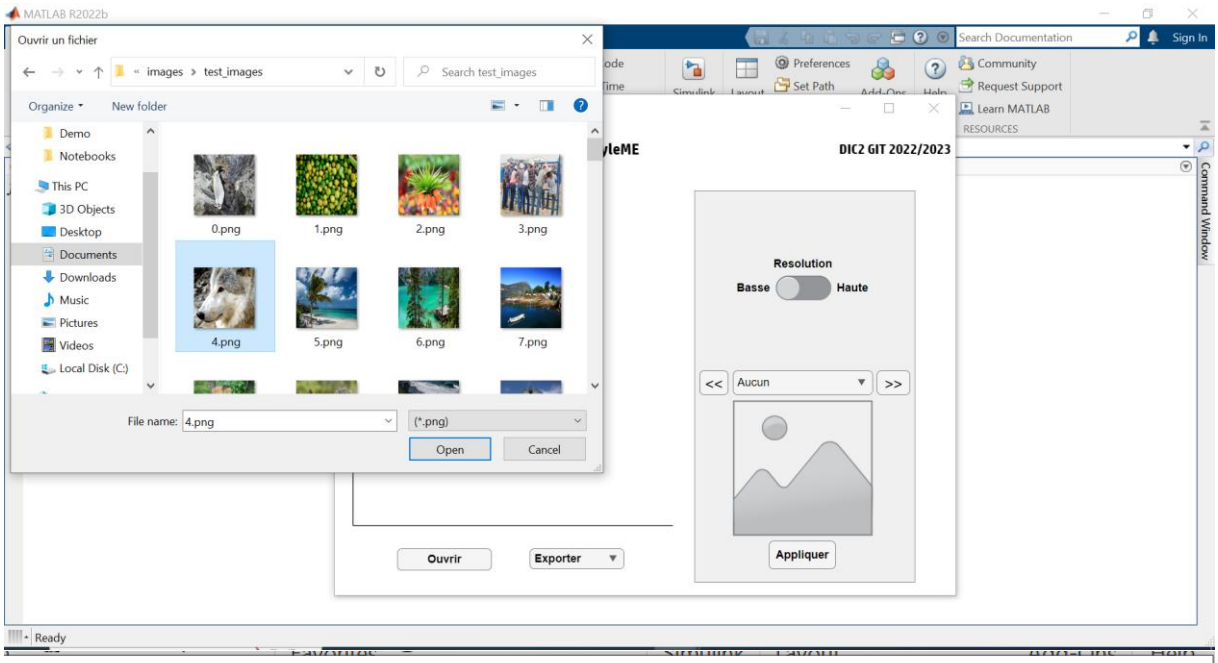
Nous pouvons donc voir que nous avons une perte de perception de 0.95 en moyenne qui n'est pas très élevée.

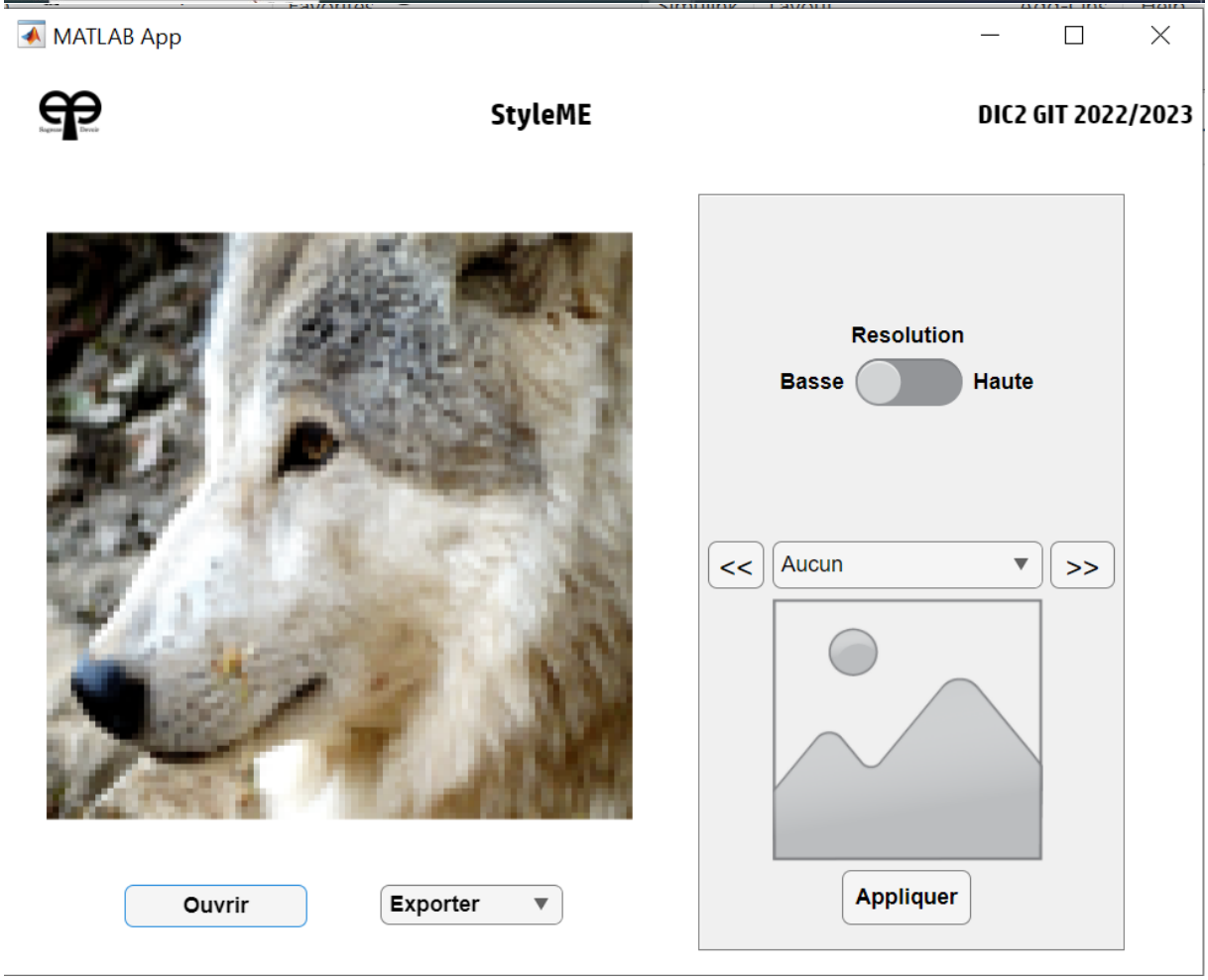
3. Application

L'application est une interface Matlab destiné au grand public dont l'objectif est de permettre à quelqu'un qui veut embellir une image d'obtenir une image

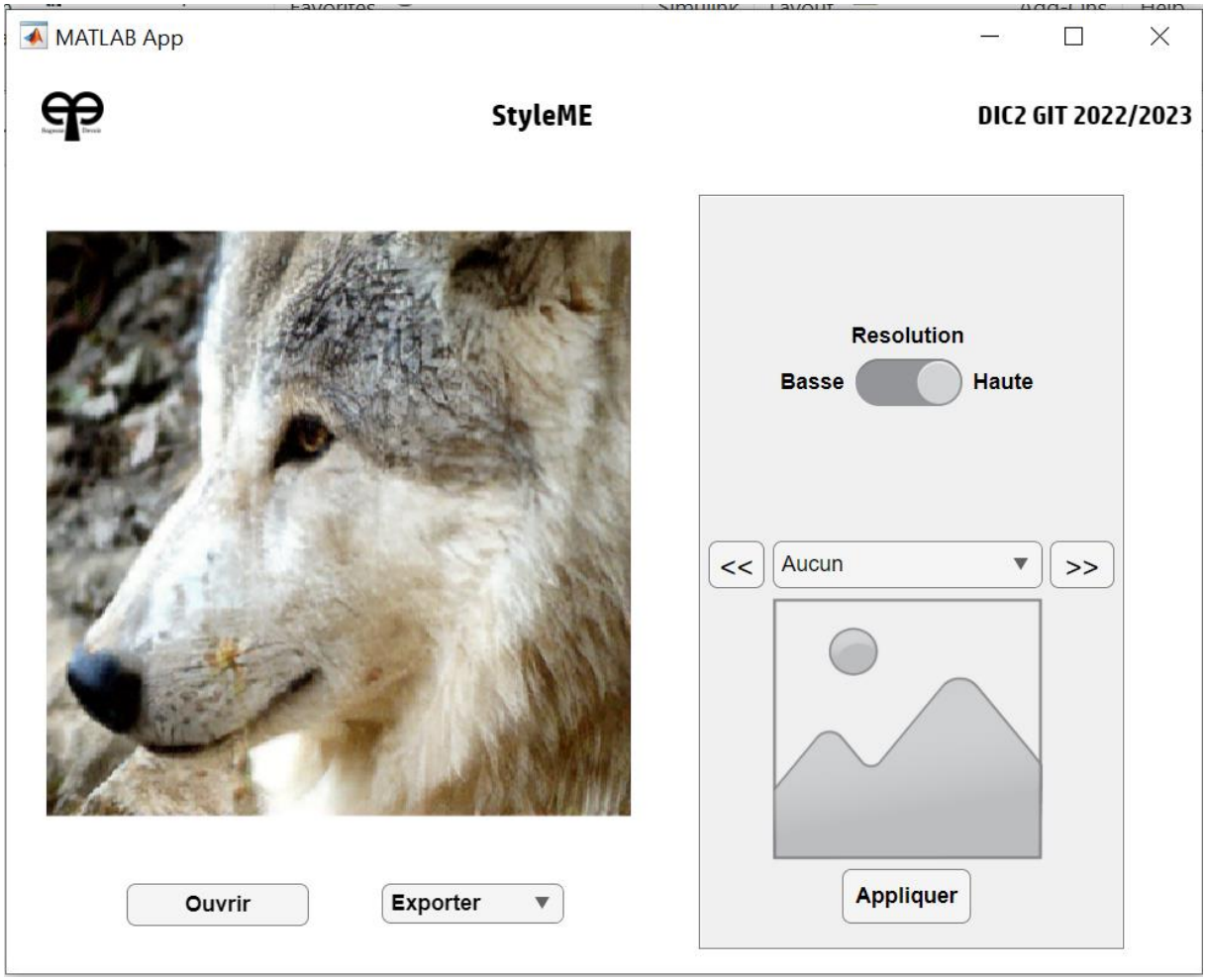
rendue sous un style artistique similaire à une image de style choisie. L'image chargée basse résolution est d'abord super résolue puis l'utilisateur a le choix de trois styles et après avoir choisi l'image lui sera affichée sous le format choisi puis il pourra l'exporter.

A. L'utilisateur charge une image basse résolution

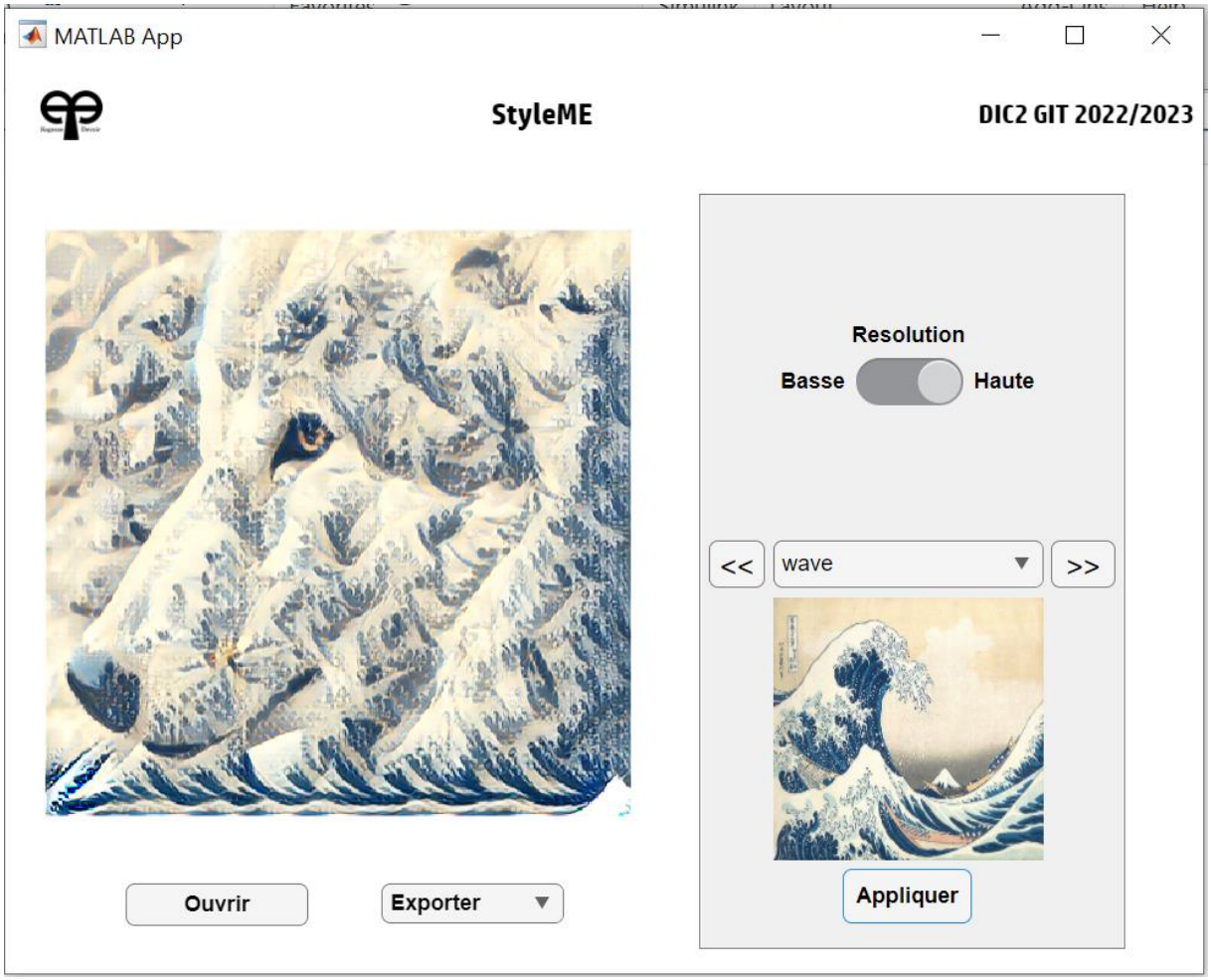




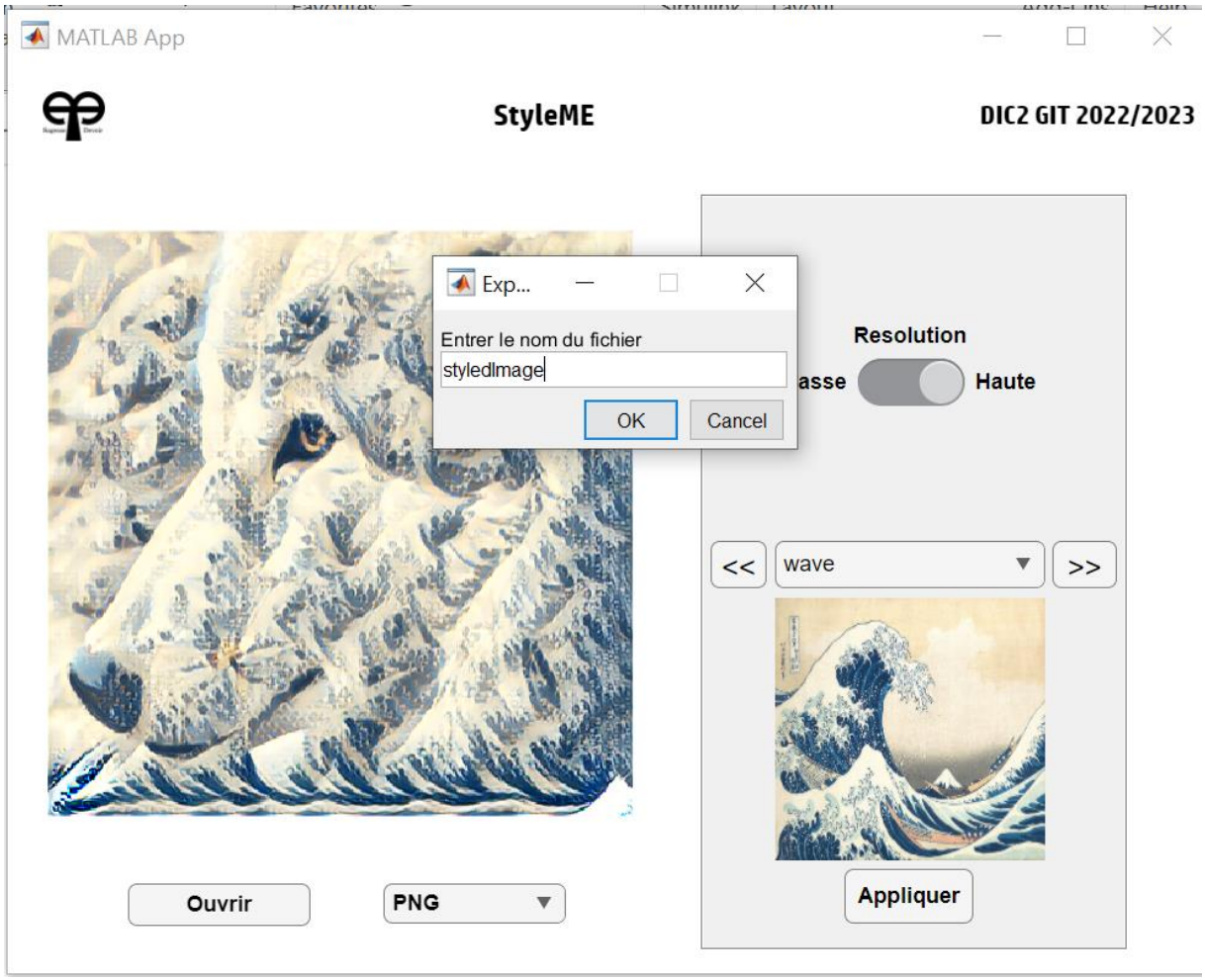
B. L'image est super résolue lorsque l'utilisateur passe le switch vers haute



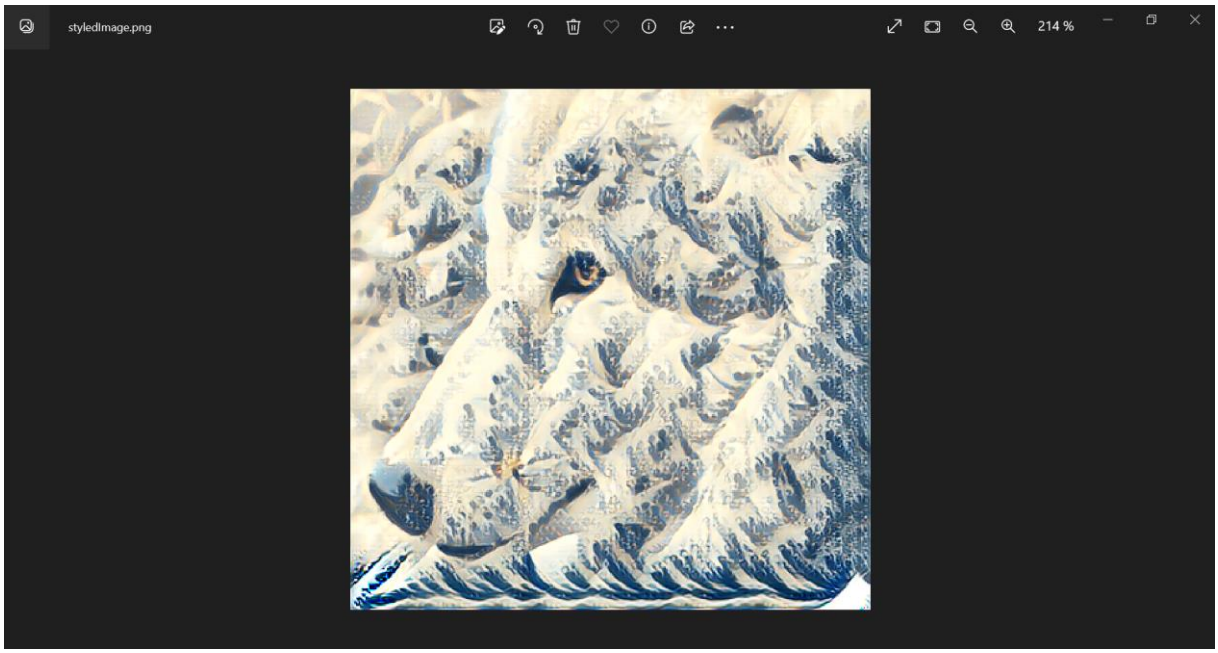
C. L'utilisateur choisit un style et qui est ensuite appliquée à l'image



D. L'utilisateur exporte l'image au format de son choix



E. L'image obtenue



III. Python : Super Résolution appliquée au diagnostic des tumeurs du cerveau

1. Workflow de l'application

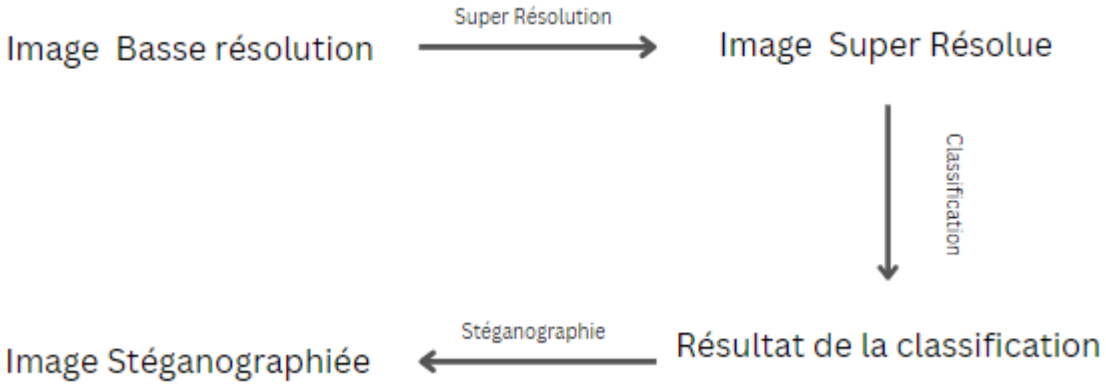


Figure 26 : Workflow de l'application de super résolution appliquée à la classification des tumeurs du cerveau

L'objectif est de proposer une solution pour l'utilisation de la super résolution dans le diagnostic du cancer du cerveau.
On commence par appliquer la super résolution sur l'image basse résolution qui correspond ici à une image d'un CT Scan. Ensuite on utilise un modèle de classification afin de déterminer la nature de la tumeur.
Pour finir après validation par le médecin on encode le résultat obtenu ainsi que les informations sur le patient en utilisant la stéganographie avec la méthode LSB pour obtenir une image stéganographiée.
Cette image sera ensuite envoyée au patient qui pourra à partir de l'application la décoder afin de voir les résultats.

2. Brain Tumor Classification

a. Présentation Générale

La détection de tumeur du cerveau est une tâche importante en imagerie médicale qui consiste à desceller sur les images de scanner ou d'IRM du cerveau la présence ou non d'une tumeur, ainsi que de son type et de sa gravité. Cette tâche est cruciale pour le diagnostic et la planification du traitement des patients atteints de tumeurs cérébrales.
Pour l'effectuer correctement il est nécessaire de disposer d'images IRM ou scanner de très bonne qualité qui sont données par les IRM dernier modèle et sont très coûteuse.
En 2017, le Sénégal ne disposait que deux machines IRM et ces dernières n'était même pas les derniers modèles.
Ainsi notre objectif est qu'en utilisant la super résolution, nous pouvons améliorer la qualité des images obtenues avec ces machines de mettre en place un modèle de classification d'images basé sur le deep learning ce qui permettra d'améliorer les résultats en les rendant plus précis et plus rapides. Ceci permet aux médecins de prendre des décisions plus éclairées en matière de diagnostic et

de traitement, ce qui peut améliorer les résultats pour les patients atteints de tumeurs cérébrales.

b. Dataset utilisé

Le dataset utilisé est le Brain Tumor Classification MRI de Kaggle qui contient 3238 images de classes glioma_tumor, no_tumor, pituary_tumor, meningioma_tumor.

c. Etapes de l'implémentation

Prétraitement des Images

- Egalisation d'histogramme et débruitage

On commence par effectuer une égalisation de l'histogramme ce qui nous permet d'augmenter la visibilité des structures. En effet au niveau de la deuxième image, il y a quelques détails fins qui sont plus visibles.
On utilise un filtre médian pour réduire le bruit ce qui permet d'enlever les effets liés au système d'imagerie tout en ajoutant du flou dans l'image comme nous pouvons le voir au niveau de la troisième image de la figure.

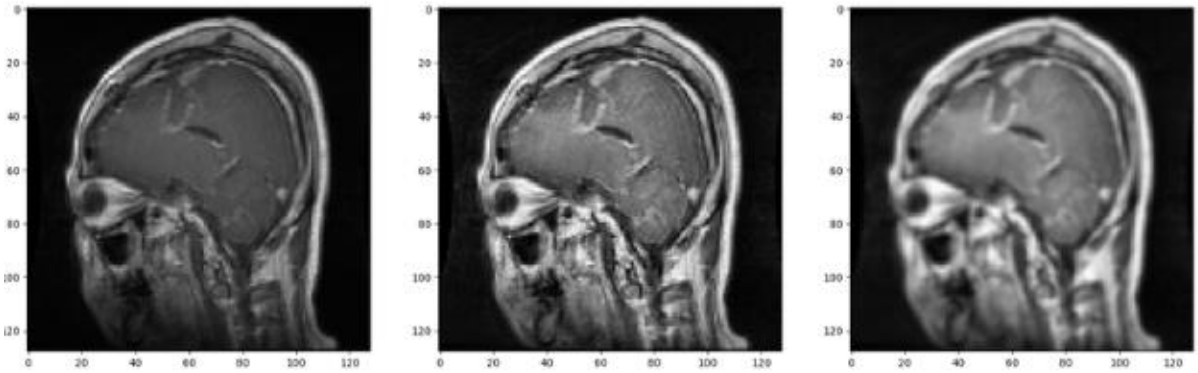


Figure 27 : Image originale, Image égalisée et Image débruitée

- Augmentation du contraste au niveau de l'image

On utilise CLAHE (Contrast Histogram Equalization) qui est une technique d'égalisation adaptive de l'histogramme qui peut être utilisée pour augmenter le contraste d'une image. Contrairement à l'égalisation d'histogramme classique qui est une méthode globale, elle divise l'image en blocs et applique l'égalisation sur chaque bloc ce qui permet une adaptation locale. Elle utilise également une limitation du contraste pour éviter l'amplification du bruit dans les zones à faible luminosité.
Nous voyons donc dans la figure ci-dessous que le contraste de l'image a bien augmenté et que certaines structures plus fines sont devenues plus visibles. Il y a cependant l'apparition de bandes sur l'image qui sont dues à la variation d'égalisation.

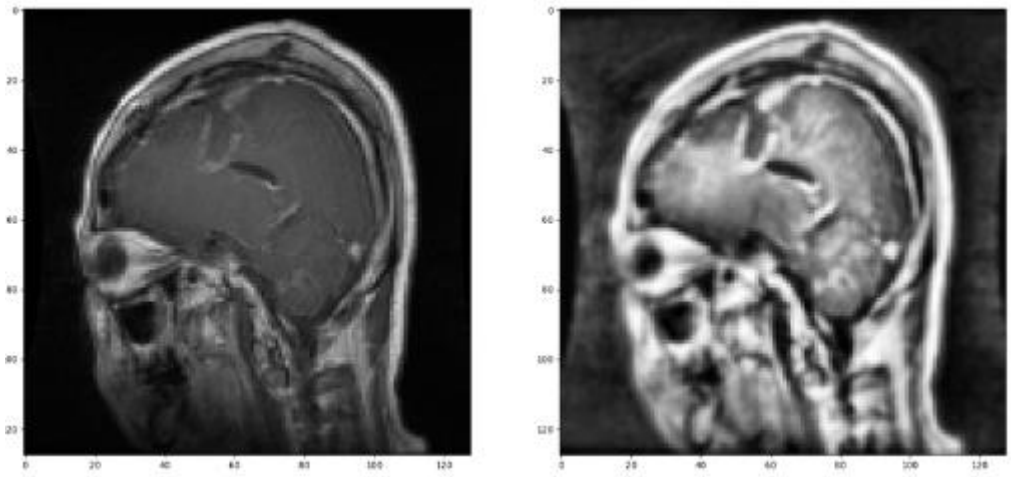


Figure 28 : Image originale et Image avec contraste augmentée

- Normaliser et mettre à l'échelle de l'intensité l'image

Pour finir on normalise les images ce qui permet de ramener les valeurs des pixels de [0,255] à un intervalle [0,1] ce qui permet d'éliminer les bandes obtenues précédemment et facilite les traitements pour le modèle de deep learning comme montré dans la figure ci-dessous.
Nous voyons donc qu'après prétraitement on aune image où les structures sont plus visibles et distinctes.

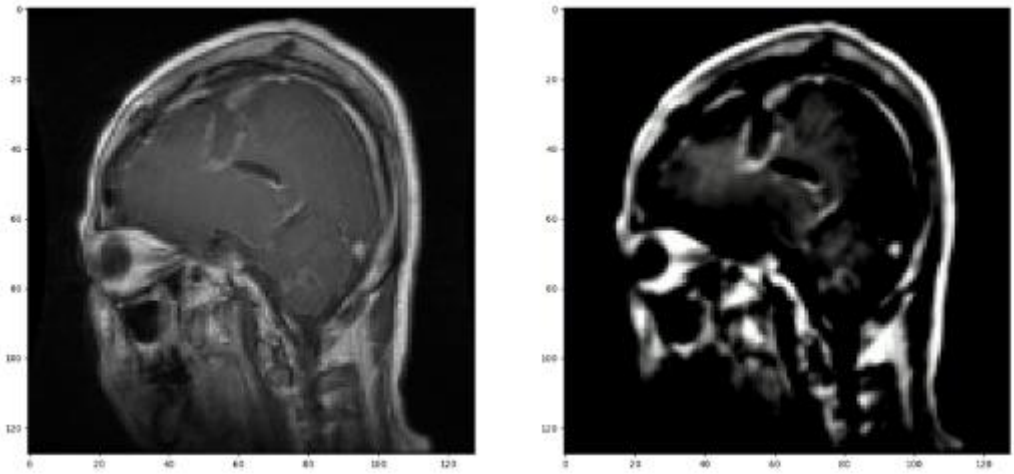


Figure 29 : Image originale et Image après prétraitement

Modèle utilisé

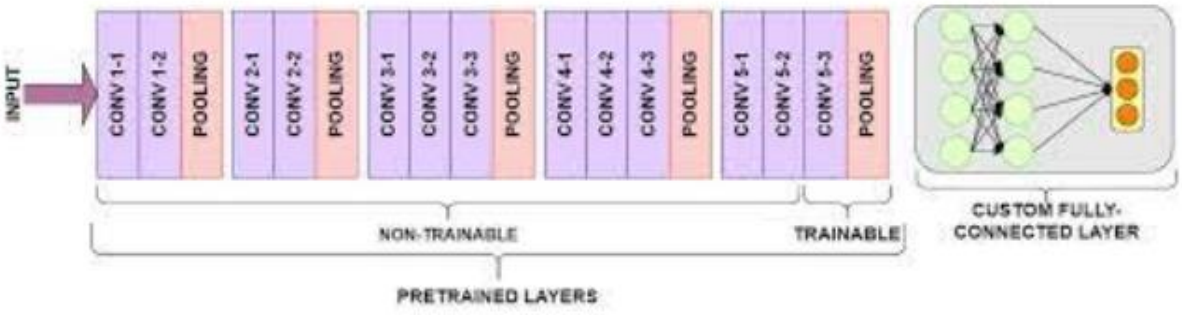


Figure 30 : Architecture du modèle de transfert learning

Le transfer learning ou apprentissage par transfert est une technique de machine learning dans laquelle un modèle pré-entrainé sur une tâche est réutiliser comme point de départ pour résoudre une nouvelle tâche similaire ou connexe.
Nous utilisons cette technique avec le modèle VGG16 dont nous rendons allons geler les couches afin d'utiliser les poids du modèle existant. Nous rajoutons des couches de Convolution et de Pooling dont les poids seront obtenus lors de l'apprentissage sur nos données.
Ce qui nous permet d'améliorer la précision et la performance sur la nouvelle tâche.

Entrainement du modèle

On sépare le jeu de données en train set, val set et test set. On entraine sur 20 epochs avec un batch size de 16 le modèle sur le train set en utilisant le val set pour comme jeu de validation.

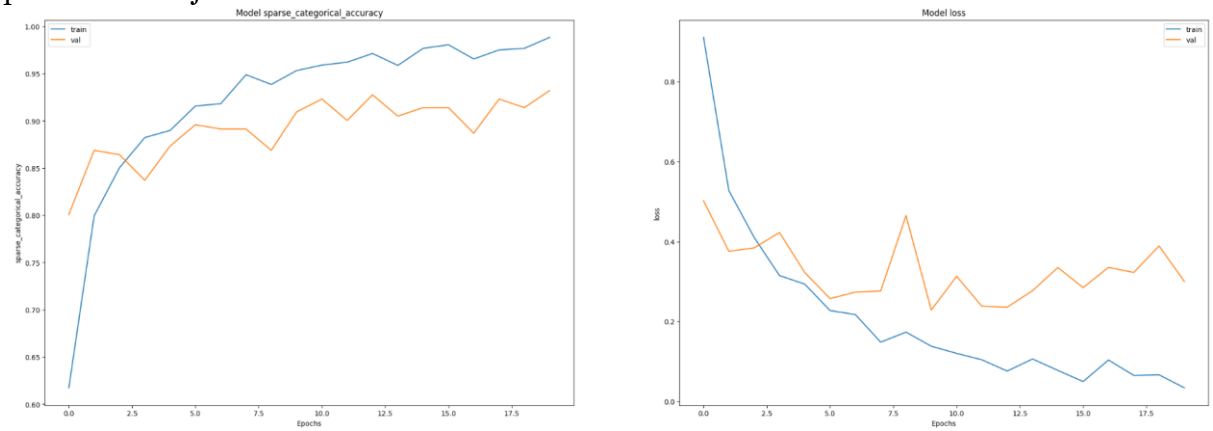
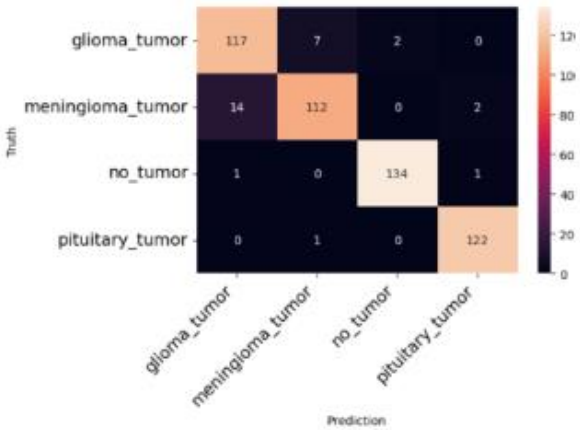


Figure 31 : Variation de la fonction coût et de la précision du modèle

Nous voyons que lors de l'apprentissage on a l'augmentation de la précision du modèle ainsi que la diminution de la fonction coût sur le training set qui avoisinent respectivement 0.98 et 0.3 à la fin. Au niveau du validation set, les résultats fluctuent suivant les epochs pour atteindre respectivement à la fin 0.93 et 0.8 de l'entrainement.

d. Analyse des résultats

Les prédictions sur le test set nous donnent les résultats suivants.



	precision	recall	f1-score	support
0.0	0.89	0.93	0.91	126
1.0	0.93	0.88	0.90	128
2.0	0.99	0.99	0.99	136
3.0	0.98	0.99	0.98	123
accuracy			0.95	513
macro avg	0.95	0.95	0.94	513
weighted avg	0.95	0.95	0.95	513

[0.45823606848716736, 0.8440545797348022]

Figure 32 : Matrice de confusion, classification report de notre modèle vs accuracy score d'un modèle classique

Nous pouvons donc voir à travers le « classification report » que le modèle a de bonnes performances avec une précision de 95% et des f1-score tournant autour de 0.90 à 0.99 pour les différentes classes.
La matrice de confusion nous montre également que les erreurs de classification ne sont que ces résultats se vérifient car la bonne classe est généralement prédite et les erreurs de classification ne sont pas nombreuses. A rappeler que ce modèle n'est qu'un outil d'aide à la prise de décision et que le diagnostic final revient évidemment au médecin qui dans des cas pareils interviendra.
Nous constatons également que cette précision dénote une augmentation 11% par rapport à un modèle entrainé sur des images basses résolutions d'où la pertinence d'utiliser un modèle de super résolution.
Le modèle répond également au seuil nécessaire pour être vu comme un modèle performant car une étude qui a été faite en 2018 à Beijing sur la classification des tumeurs du cerveau a montré qu'une équipe de 15 médecins spécialistes en neurologie avait atteint une précision de 66% (*source : https://aibusiness.com/verticals/chinese-ai-beats-human-doctors-in-diagnosing-brain-tumours).*

3. Stéganographie

a. Présentation Générale

La stéganographie est l'art de cacher des informations à l'intérieur d'un support sans que cela soit détecté. Cela peut inclure l'utilisation de techniques telles que la modification des bits de données, l'insertion de données dans des fichiers multimédias tels que des images ou des vidéos, ou l'utilisation de caractères spéciaux dans du texte pour cacher un message.
Dans notre projet on utilise la stéganographie pourrait être utilisée pour cacher les résultats de l'examen de façon à ce que seul le médecin qui analyse les résultats et le patient qui en est le destinataire puisse les voir. Ce qui peut être important pour protéger la vie privée et la confidentialité du patient, car les résultats d'un scanner de cerveau peuvent révéler des informations personnelles.

b. Etapes de l'implémentation

Nous utilisons donc la stéganographie LSB (Least Significant Bit) qui est une technique de stéganographie qui consiste à cacher des données dans les bits de poids faible (moins significatifs) d'une image.
En effet les images ont souvent une grande quantité de données inutilisées dans les bits de poids faible qui peuvent être utilisées pour cacher des données supplémentaires.
Le processus est le suivant :

- Convertir le message en bloc de 8 bits par caractères en utilisant les Unicode
- Ajouter un bit sentinelle (délimiteur) à la fin du message ainsi que du padding
- Récupérer la liste des pixels de l'image
- Récupérer les valeurs des différents canaux de l'image (R, G, B) et on convertit chaque valeur de pixel en binaire, en utilisant 8 bits pour représenter chaque canal.
- Insérer chaque bit du message dans le bit de poids faible de chaque pixel de l'image hôte, en commençant par le premier pixel de l'image. Par exemple, pour le premier octet du message, remplacer les 2 derniers bits du pixel rouge

du premier pixel, les 3 derniers bits du pixel vert du premier pixel, et les 3 derniers bits du pixel bleu du premier pixel avec les 8 bits du message.

c. Analyse des résultats

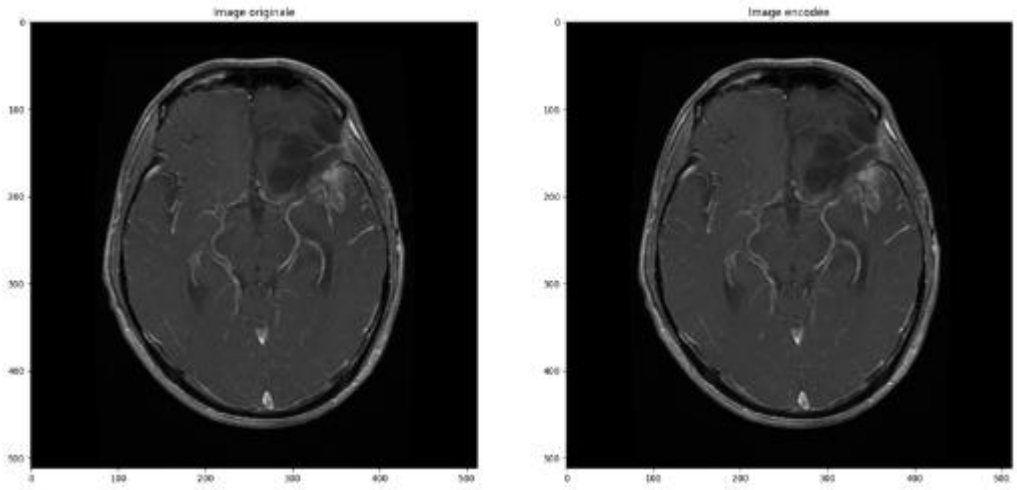


Figure 33 : Image encodée et Image originale sans stéganographie

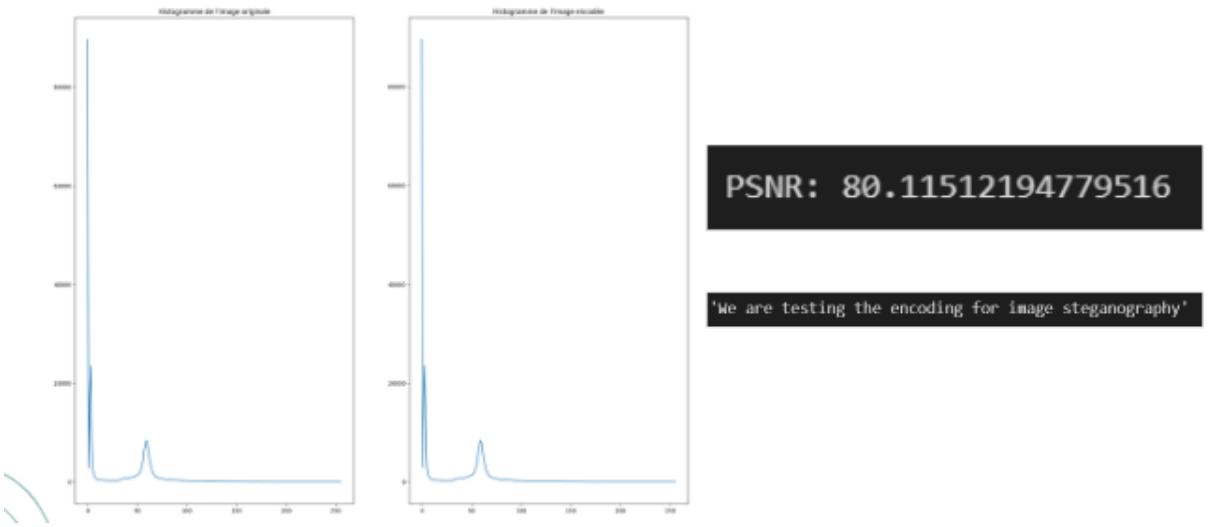


Figure 34 : Histogramme des deux images, PSNR et Décodage du message

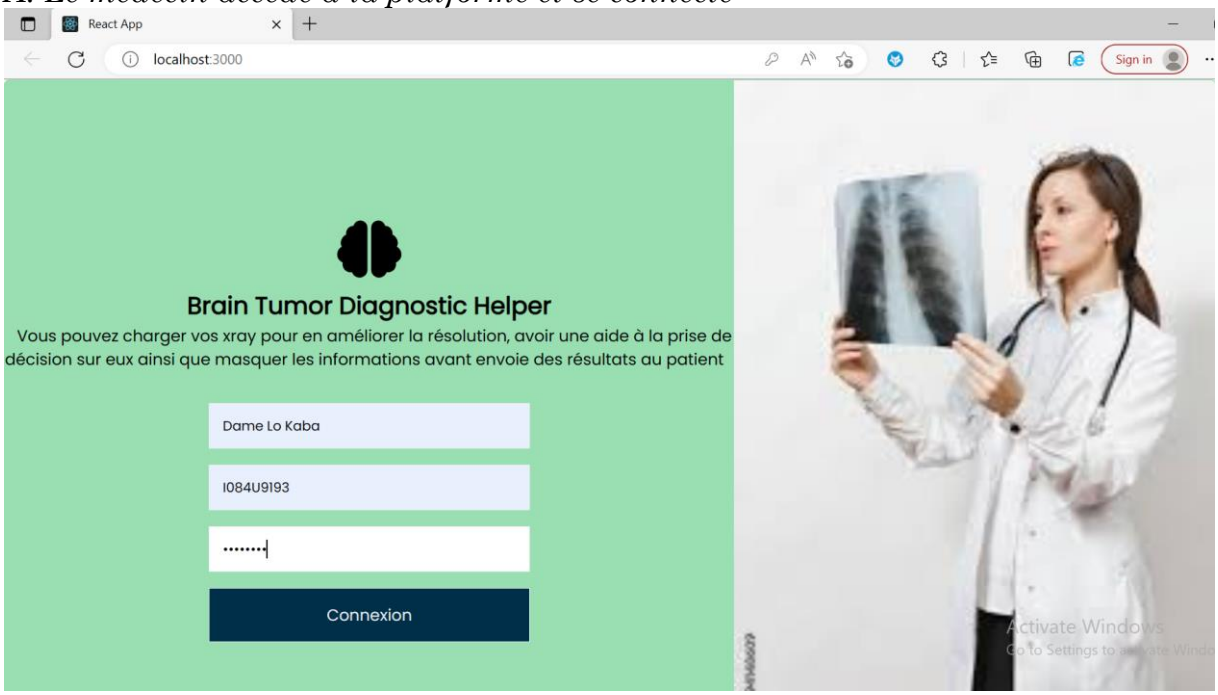
Nous voyons donc que le message est encodé dans l'image et qu'il n'y a pas de modification apparente sur l'image ni sur l'histogramme ce qui montre que la quantité d'information préservé est haute par rapport au bruit introduit par l'ajout du message. Ceci est confirmé par la valeur du PSNR qui est de 80.11. En décodant l'image on retrouve donc le message que l'on y avait initialement inséré.

4. Application

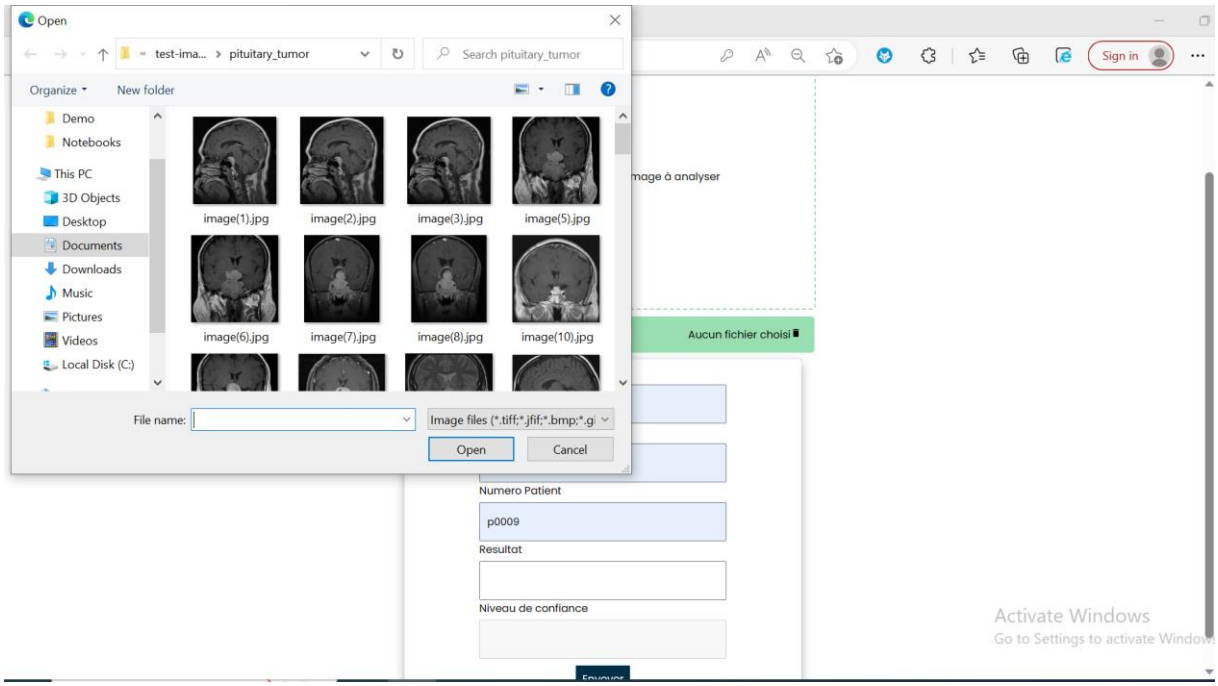
L'application est développée en utilisant FastAPI comme technologie de backend permettant de créer une API que l'on pourra accéder afin d'effectuer la super résolution, la classification et la stéganographie (codage et décodage) grâce à des méthodes que nous y avons défini qui utilise les modèles que nous avons précédemment entrainé et enregistrer. On utilise React comme technologie Frontend afin de créer l'interface utilisateur.
L'application est une plateforme d'aide au diagnostic des tumeurs du cerveau où les médecins d'un hôpital peuvent après s'être connecté charger les informations

une image d'un scanner basse résolution et les informations sur le patient. Cette image sera ensuite super résolue en faisant appel à l'endpoint défini dans notre API pour retourner une image plus visible pour le médecin qui pourra soit faire son diagnostic à l'aide de celle-ci ou envoyer cette image à l'endpoint de classification qui lui donnera en retour une prédiction sur la présence ou non de tumeur, le type de tumeur ainsi que son niveau de confiance. Le médecin peut choisir de garder ou modifier ce diagnostic suivant son opinion médicale.
Les informations sur le patient et le diagnostic final sont ensuite encodées à l'intérieur de l'image super résolue et un mail est envoyé au patient contenant une URL où il pourra voir l'image stéganographiée.
Une fois sur la plateforme le patient pourra en appuyant sur le bouton afficher faire appel à l'endpoint permettant de décoder les informations de l'image stéganographiée et obtenir les résultats.

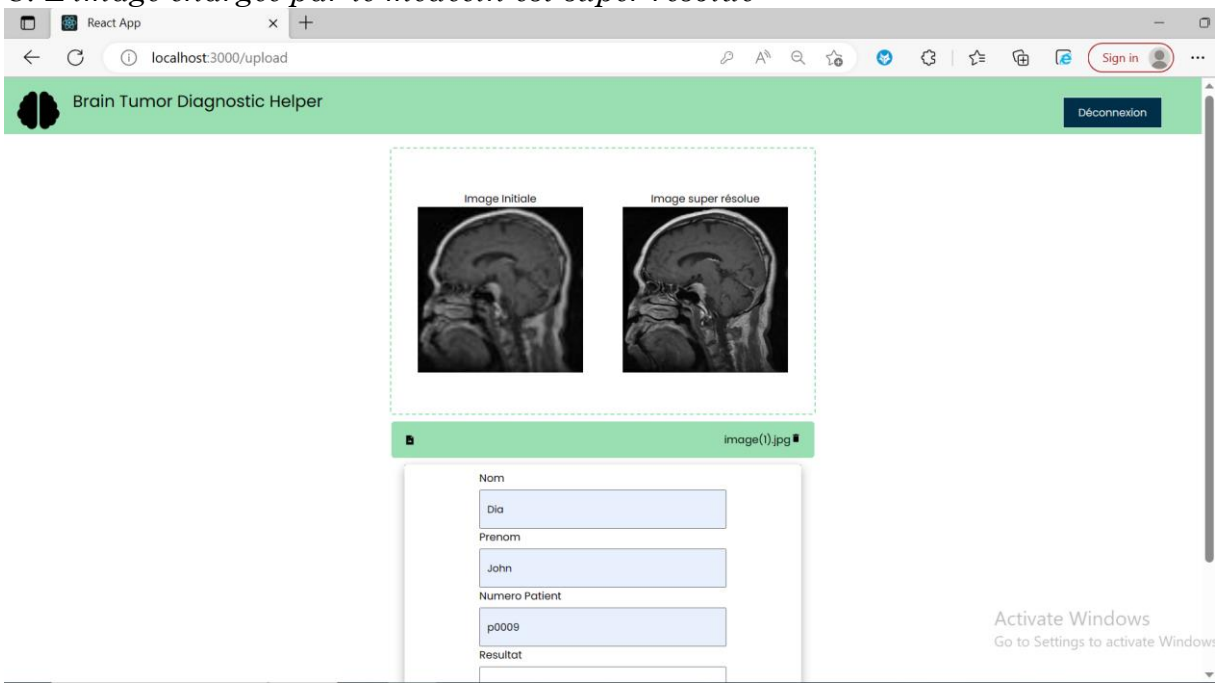
A. Le médecin accède à la platforme et se connecte



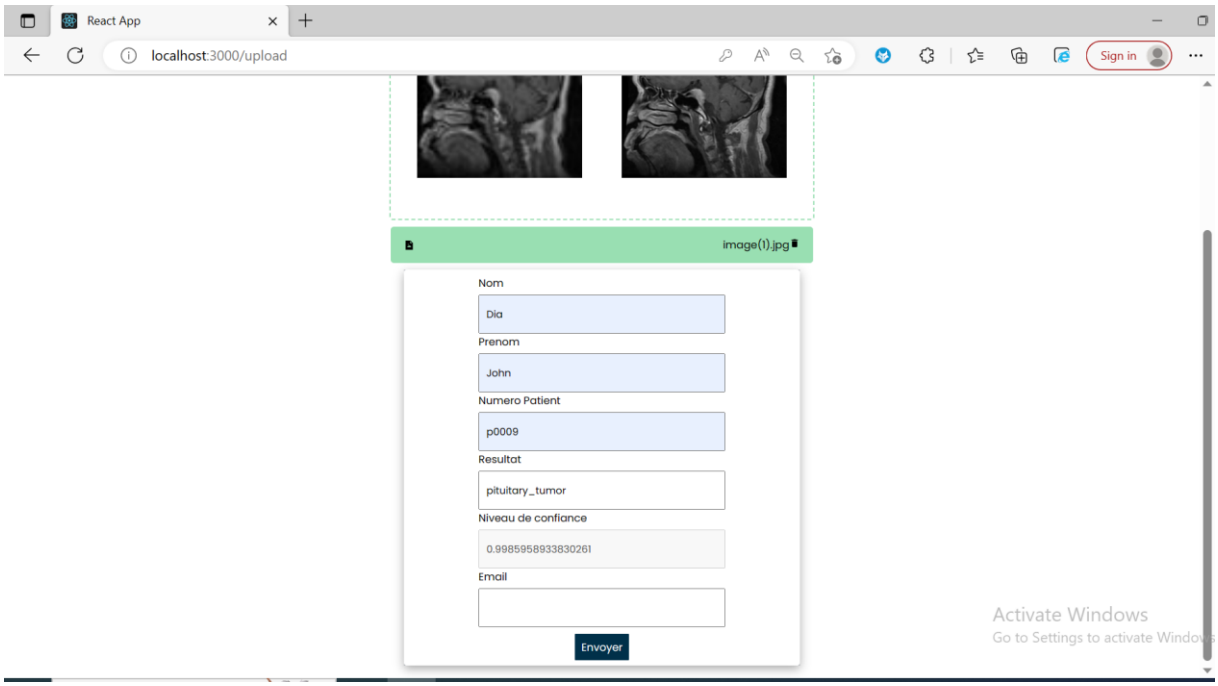
B. Le médecin renseigne les informations du patient et charge le scanner



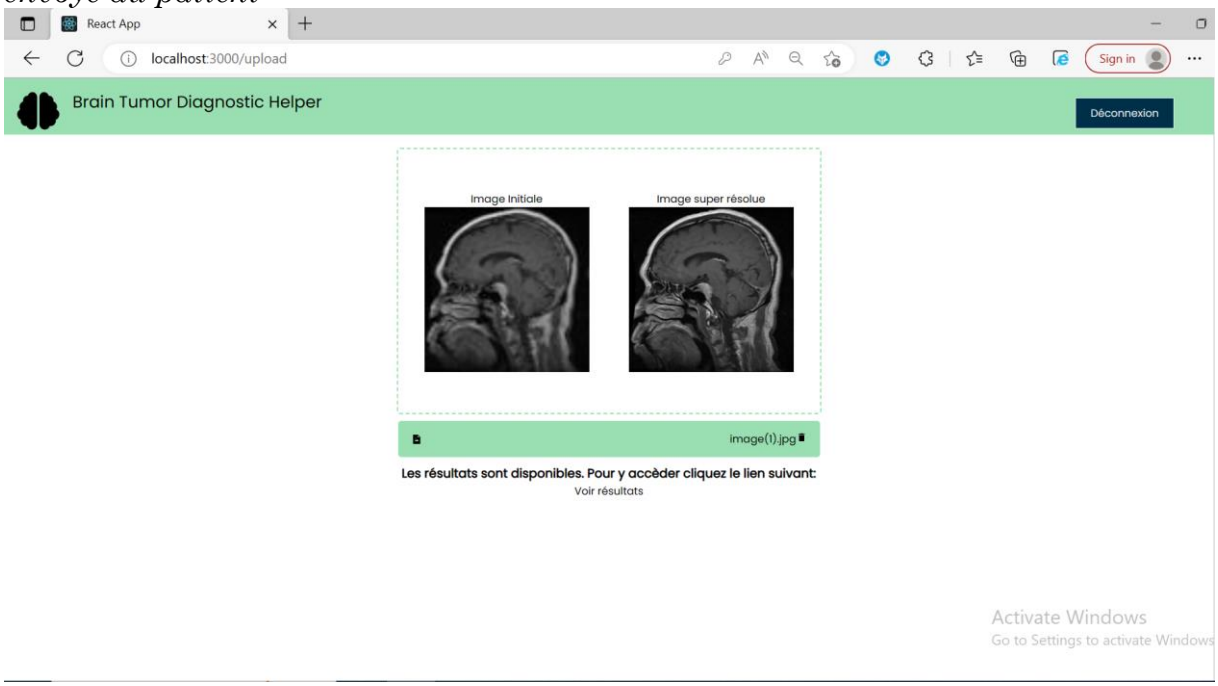
C. L'image chargée par le médecin est super résolue



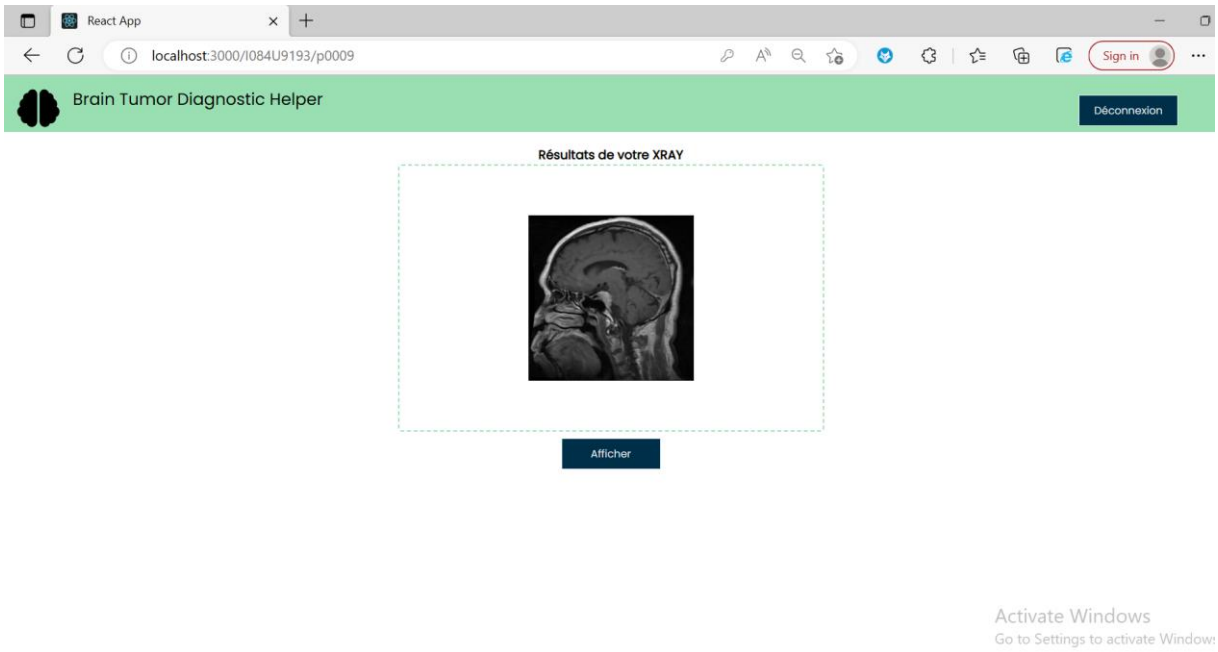
D. Le résultat du modèle de classification et le niveau de confiance est affiché sur l'image et le résultat affiché au médecin qui peut soit modifier soit conservé suivant son opinon



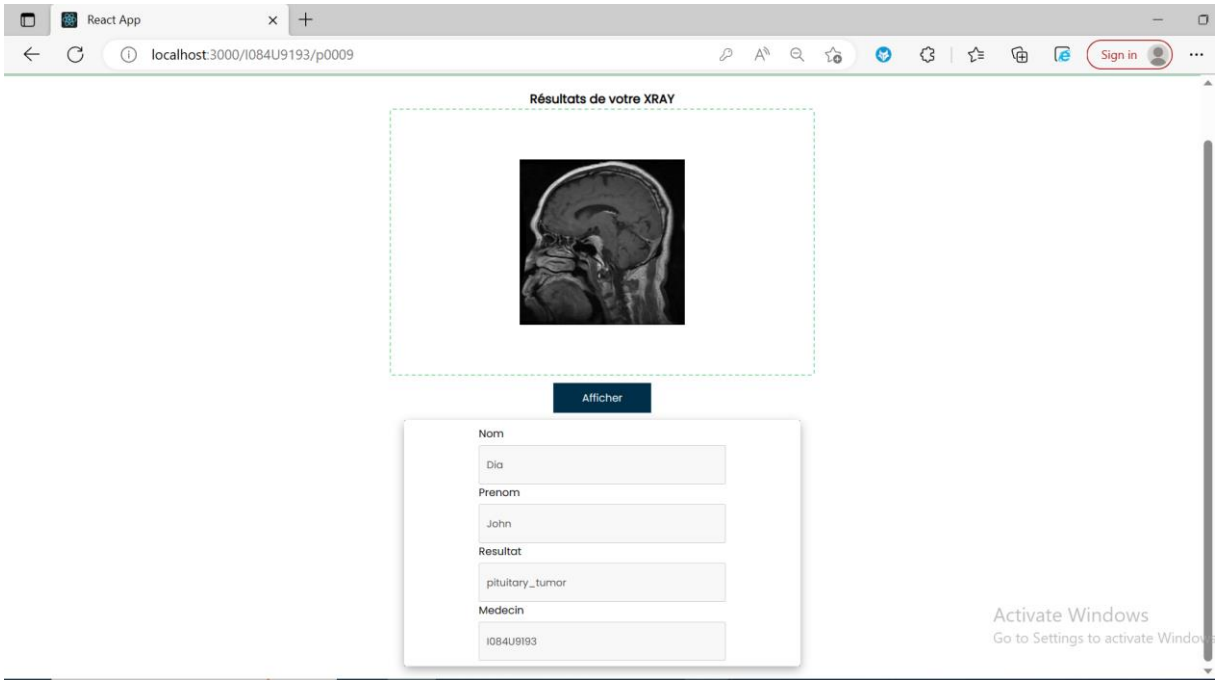
E. Les résultats sont encodés dans l'image et l'URL d'accès à la plateforme est envoyé au patient



F. Le patient accède à la platforme et on lui montre l'image stéganographiée



G. Il appuie sur le bouton « Afficher » et l'image est décodée et les informations qui s'y trouve lui sont montrées



Conclusion et Perspectives

En conclusion, nous avons pu remarquer l'efficacité de l'utilisation de la super-résolution pour améliorer la qualité de l'image dans le cadre du neural style transfer ainsi que la qualité visuelle de détection de tumeurs du cerveau. Elle conduit à une nette amélioration de la qualité des images traitées, ainsi qu'une augmentation significative de la précision de la détection de tumeurs.
Les perspectives de ce projet sont :

- Appliquer la super résolution couplée au NST à des images d'artistes africains afin de valoriser l'art locale
- Appliquer la super résolution d'images à la restauration d'images anciennes présentant du flou ou une dégradation
- D'augmenter le nombre de maladies que nous pouvons classifier avec le modèle de classification
- Améliorer la qualité des images issues des caméras de surveillance$

Nous avons eu à démarrer les deux premières mais les jeux de données dont nous disposions ne nous permettaient pas d'avoir de bons résultats constamment. Nous avons cependant eu à effectuer quelques tests qui nous laisse envisager qu'il serait effectivement possible d'y parvenir avec un jeu de données adapté

Figure 35 : Restauration d'une image des tirailleurs sénégalais

Figure 36 : Valorisation des œuvres d'art africaines

Webographie

[1] Brain Tumor MRI Dataset. (2021, September 24). Kaggle. https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset
[2] Neso Academy. (2021, August 12). *LSB Steganography - Demo* [Video]. YouTube. https://www.youtube.com/watch?v=yNo58UiIMKU
[3] Adeshina, A. A. (2022). Developing a Single Page App with FastAPI and React. *TestDriven.io*. https://testdriven.io/blog/fastapi-react/
[4] Marimuthu, P. (2022). Image Contrast Enhancement Using CLAHE. *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2022/08/image-contrast-enhancement-using-clahe/
[5] Wikipedia contributors. (2023). Fréchet inception distance. *Wikipedia*. https://en.wikipedia.org/wiki/Fr%C3%A9chet_inception_distance
[6] Wang, X. (2018, September 1). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1809.00219

[7] *Train Fast Style Transfer Network - MATLAB & Simulink*. (n.d.). https://www.mathworks.com/help/deeplearning/ug/train-fast-style-transfer-network.html

[8] Gatys, L. A. (2015, August 26). *A Neural Algorithm of Artistic Style*. arXiv.org. https://arxiv.org/abs/1508.06576