Projet - Transfert de Style

Automatants

Thibault Nadin

2023

AUTOMATANTS

Table of Contents

Introduction

Le transfert de style neuronal est une technique d'optimisation utilisée pour former une image à partir de deux images en entrée :

- une image de contenu
- une image de référence de style (telle qu'une œuvre d'un peintre célèbre)

L'objectif est de les mélanger afin que l'image de sortie ressemble à l'image de contenu, mais dans le style de l'image de référence de style.

Ceci est mis en œuvre en optimisant l'image de sortie pour qu'elle corresponde aux statistiques de contenu d'une première image dite de contenu et aux statistiques de stylistique d'une seconde image : la référence de style.

Ces statistiques sont extraites des images à l'aide d'un réseau convolutif.

Principe

Figure: Exemple de transfert de style

Exemples

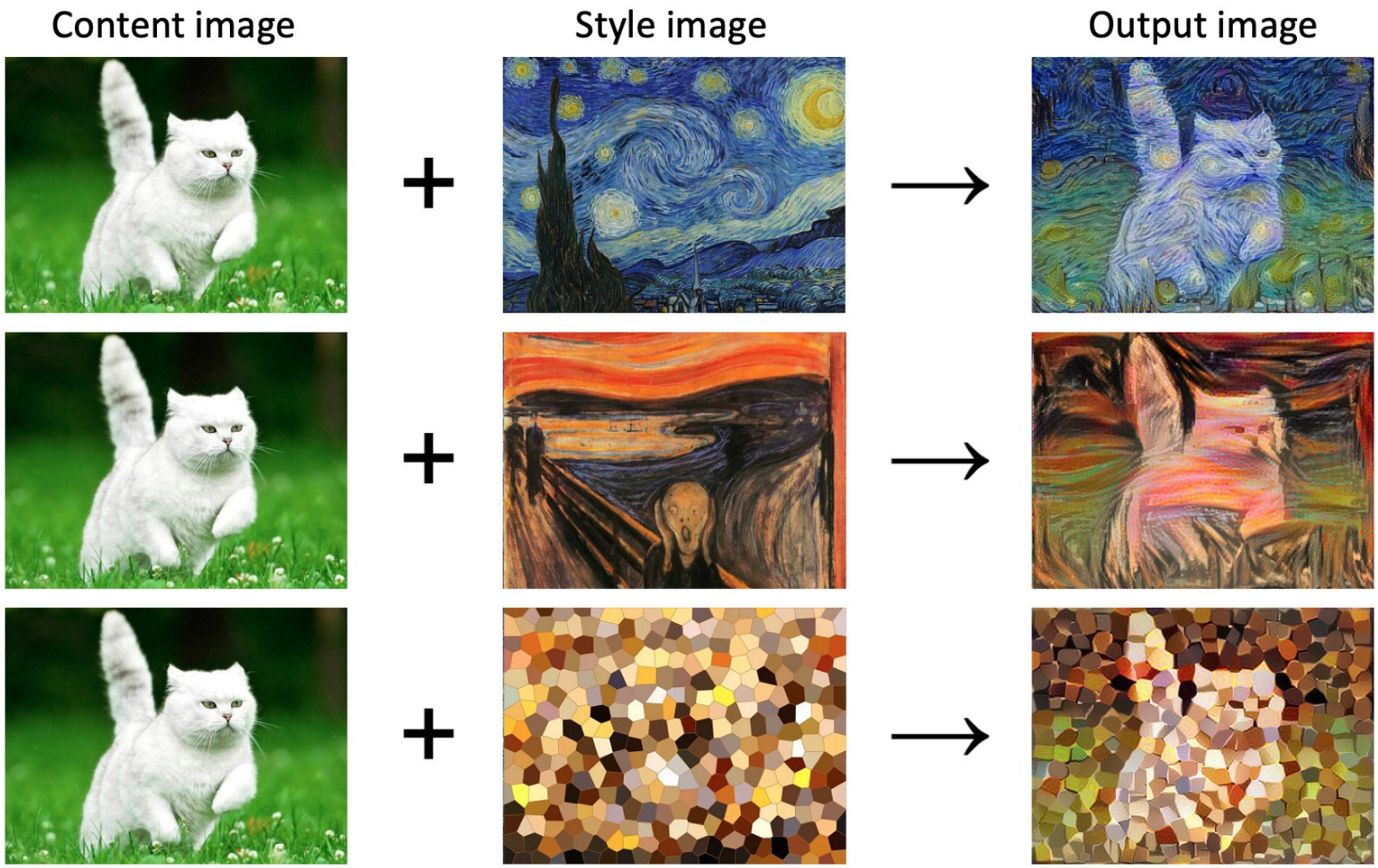


Figure: Exemples de transfert de style

Ressources utiles

1. ARTICLE by Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, **A Neural Algorithm of Artistic Style**
2. Quelques explications du transfert de style à partir de l'article précédent
3. **Didactiel Tensorflow** : Transfert de style neuronal

Table of Contents

CNN - Réseaux neuronaux à convolutions

- **Données :** Dataset Kaggle contenant des visages et des attribus à prédire
- **Objectif :** prédire les labels de `list_attr_celeba.csv`.
- **Code dispo :** baseline disponnible (coder le réseau sois-même).

GAN - Réseaux antagonistes

- **Données :** Dataset Kaggle contenant des visages
- **Objectif :** générer des images réalistes de visages humains en utilisant un GAN
- **Code dispo :** baseline disponnible (coder le réseau sois-même).

Table of Contents

Extraction des caractéristiques dans un CNN

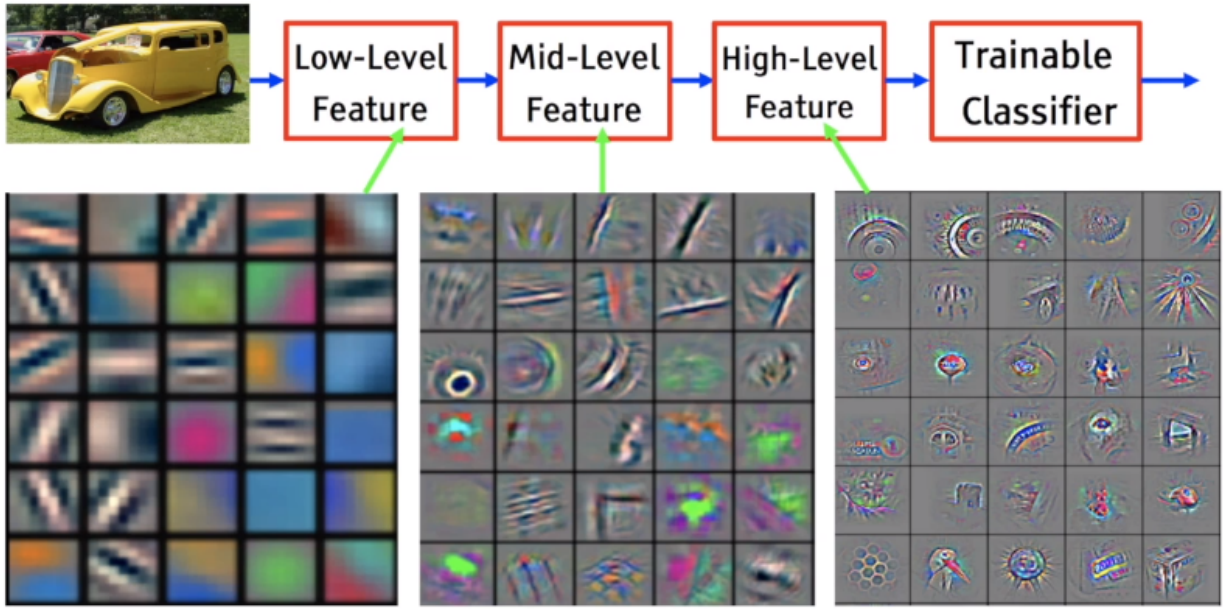


Figure: Extraction progressive des caractéristiques dans un CNN

- Les couches de convolution initiales capturent généralement des caractéristiques de bas niveau telles que les bords et les textures,
- les couches plus profondes peuvent détecter des formes plus complexes comme des motifs, des objets partiels ou des textures spécifiques.

Exemple

Regardons les différentes couches de VGG19 pour l'entrée suivante.

Exemple

Figure: Une couche de bas niveau

Exemple

Figure: Une couche de niveau moyen

Exemple

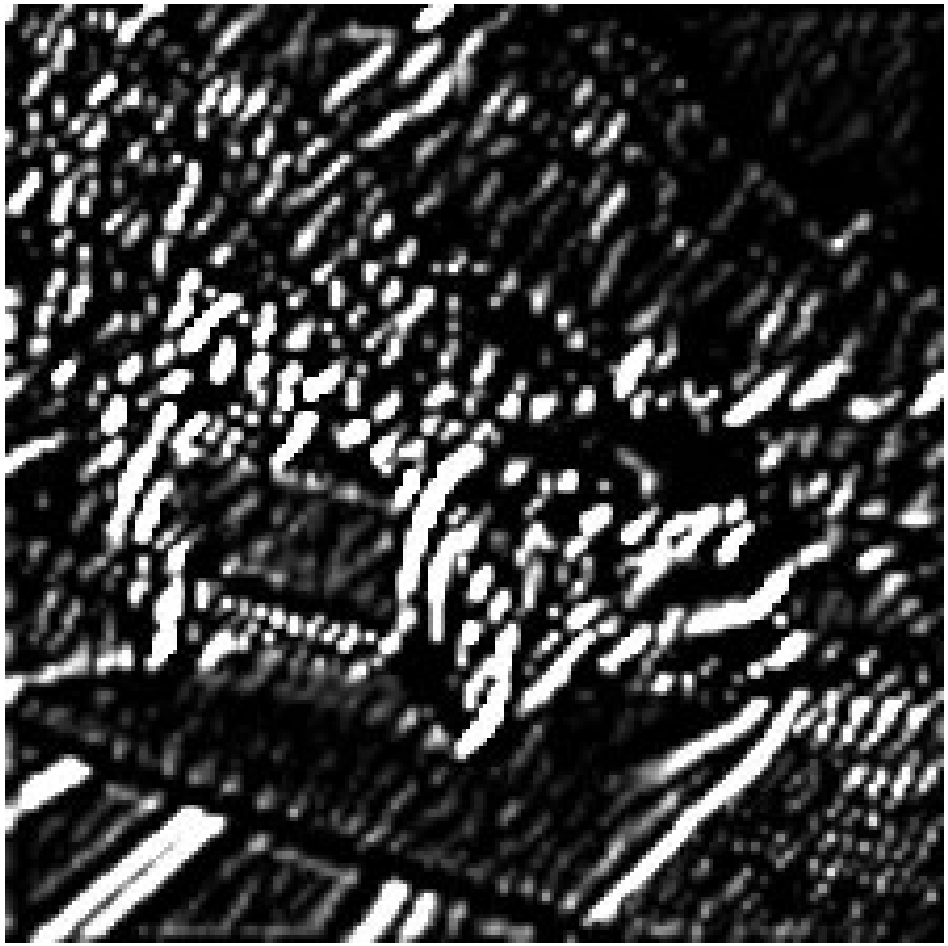


Figure: Une couche de haut niveau

Eléments mathématiques

Mean Squared Error (MSE)

Pour $\mathcal{I}, \mathcal{J} \in \mathbb{R}^{np}$

$$\text{MSE}(\mathcal{I}, \mathcal{J}) = \frac{1}{np} \sum_{i,j} (\mathcal{I}_{ij} - \mathcal{J}_{ij})^2$$

Matrice de Gram

Pour $x_1, ..., x_p \in \mathbb{R}^n$, $G(x_1, ..., x_p) = (x_1^\top x_p)_{1 \leq i,j \leq p}$ est la matrice de Gram de la famille $(x_1, ..., x_p)$

Propriété :
La matrice de Gram des colonnes de $M \in \mathcal{M}_n(\mathbb{R})$ est $G(M) = M^\top M$

Comment mesurer la fidélité du contenu ?

- Première idée : utiliser la MSE entre l'image de contenu et l'image générée,
- Si on fait cela, le style sera lui aussi copié,
- Il faut donc prendre en compte le contenu d'une image mais pas son style. Pour cela, on compare les cartes de caractéristiques du contenu souhaité et de l'image obtenue avec la MSE par exemple.

Métrique de contenu

Pour extraire les features de l'image, on propose d'utiliser la sortie `conv4_2` du CNN `VGG_19` pré-entrainé sur le dataset ImageNet.

Il faut donc importer `VGG_19` et créer un modèle qui retourne la sortie de la couche `conv4_2` de `VGG_19` (on le note $\mathcal{V}_{42}$).

La métrique utilisée pour quantifier la fidélité au contenue est :

Content Loss

$$\mathcal{L}_{\text{content}}(\mathcal{Y}, \mathcal{X}_{\text{content}}) = \text{MSE}(\mathcal{V}_{42}(\mathcal{Y}), \mathcal{V}_{42}(\mathcal{X}_{\text{content}}))$$

Comment mesurer la cohérence stylistique ?

Objectif : prendre en compte uniquement le style dans la comparaison de deux images.

Solution :

- Les matrices de Gram capturent des informations sur les motifs et les textures présents dans une image, ce qui est lié à la cohérence stylistique.
- En effet, plus il y a de ressemblance entre les coordonnées d'un vecteurs plus ils sont proche de la co-linéarité et leur produit scalaire est grand en valeur absolue.
- Utiliser les couches d'un CNN pour extraire à différents niveaux des caractéristiques (contours, textures, motifs ...) et comparer leur matrice de Gram

Métrique de style

Pour extraire les features de l'image à différents niveaux, on propose d'utiliser les sorties des couches `conv1_1`, `conv2_1`, `conv3_1`, `conv4_1` et `conv5_1` du CNN `VGG_19` pré-entrainé sur le dataset ImageNet.

Il faut donc importer `VGG_19` et créer des modèles qui retournent la sortie de chacune des couches (`conv1_1`, `conv2_1`, `conv3_1`, `conv4_1` et `conv5_1`) de `VGG_19` (notés $\mathcal{V}_{11}$, $\mathcal{V}_{21}$, $\mathcal{V}_{31}$, $\mathcal{V}_{41}$ et $\mathcal{V}_{51}$).

La métrique utilisée pour quantifier la cohérence stylistique est :

Style Loss

$$\mathcal{L}_{\text{style}}(\mathcal{Y}, \mathcal{X}_{\text{style}}) = \sum_{i=1}^{5} \text{MSE}(G(\mathcal{V}_{i1}(\mathcal{Y})), G(\mathcal{V}_{i1}(\mathcal{X}_{\text{style}})))$$

Métrique finale

La métrique utilisée pour quantifier la cohérence stylistique et la fidélité au contenu est :

Loss

$$\mathcal{L}(\mathcal{Y}, \mathcal{X}_{\text{style}}, \mathcal{X}_{\text{content}}) = \lambda\, \mathcal{L}_{\text{content}}(\mathcal{Y}, \mathcal{X}_{\text{content}}) + \mathcal{L}_{\text{style}}(\mathcal{Y}, \mathcal{X}_{\text{style}})$$

Où λ est un paramètre à choisir selon le rendu souhaité.

Figure: Influence du paramètre λ de gauche à droite, $\lambda = 10^{-5}$, 10^{-4}, 10^{-3} et 10^{-2}

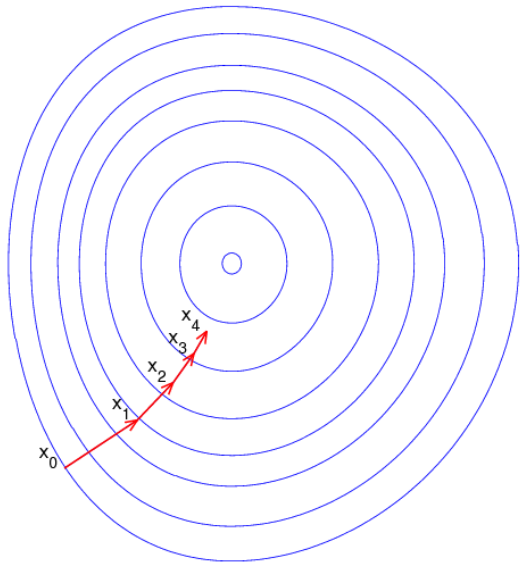
Table of Contents

Raffinement itératif

Principe : Appliquer l'algorithme de la descente de gradient directement sur les paramètres (pixels de l'image) et non un modèle de deep-learning.

Solution :

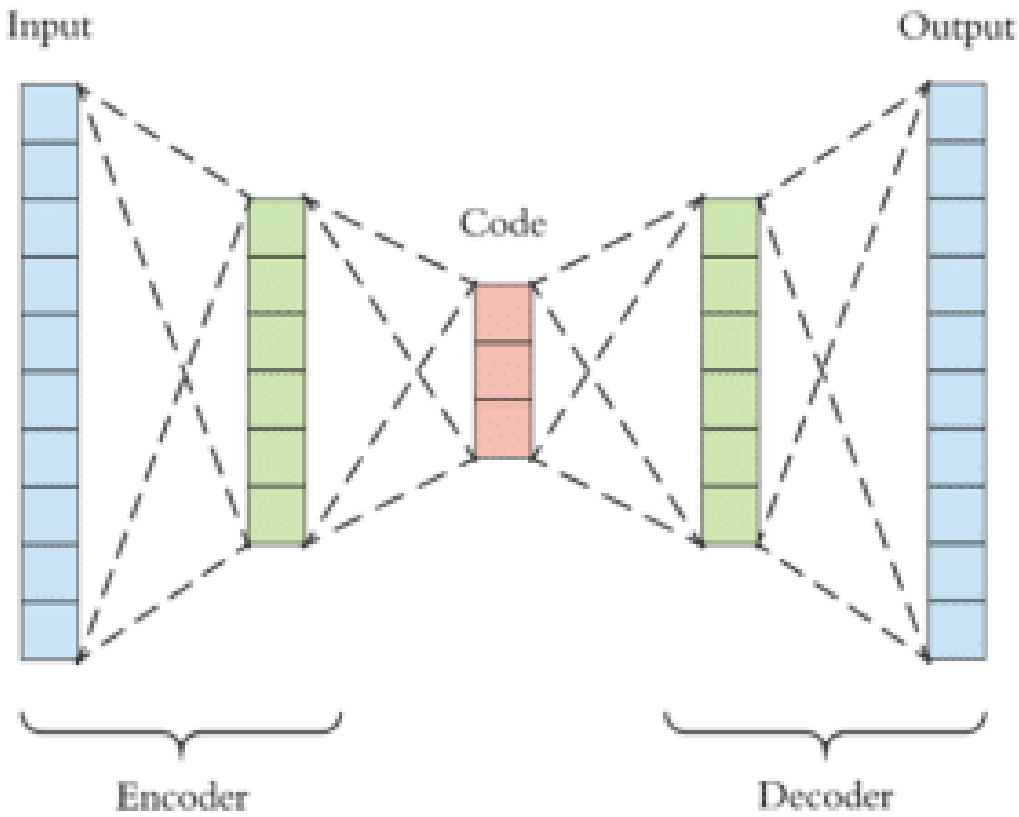
- Ainsi pour une image de taille `(n, m, 3)`, on a $3nm$ paramètres à optimiser (entraînement beaucoup plus court qu'un modèle de Deep Learning : CNN, GAN ...).
- Descente de gradient :



Auto-encodeur

Objectif : Construire une nouvelle représentation d'un jeu de données (ici batch images contenu et style) plus compacte avec moins de descripteurs. La dimensionnalité des données est réduite.

Solution :



Connections résiduelles

Objectif : Améliorer la reconstruction de l'image dans la phase de "décodage" pour rester fidèle au contenu.

Solution :

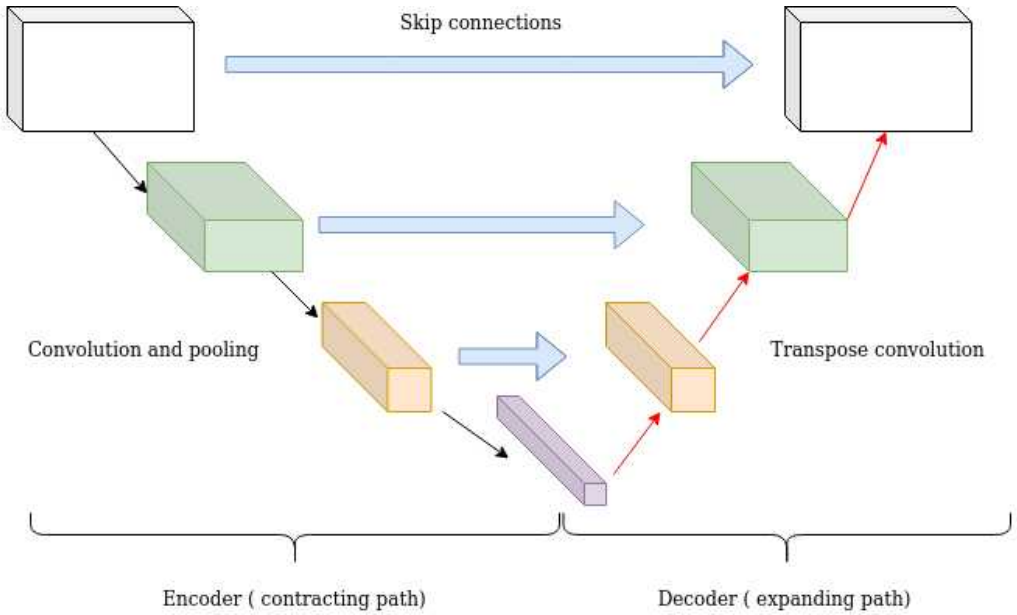


Figure: Schéma général de l'auto-encodeur avec des connections résiduelles

Ajout d'une loss adversariale

Création d'un discriminateur entraîné à distinguer les images généréés des images artistiques. Intuitivement : ajout d'un critique d'art qui doit distinguer l'original et la copie.

Nouvelle Loss

$$\mathcal{L}(\mathcal{Y}, \mathcal{X}_{\text{style}}, \mathcal{X}_{\text{content}}) = \lambda\, \mathcal{L}_{\text{content}}(\mathcal{Y}, \mathcal{X}_{\text{content}}) + \mathcal{L}_{\text{style}}(\mathcal{Y}, \mathcal{X}_{\text{style}}) + \beta \mathcal{L}_{\text{disc}}(\mathcal{Y})$$

Ressources utiles

- Entraînement sur Google Colab
- Me contacter : @thibaultndn
- Internet : stackoverflow, documentation pytorch ...