



UNIVERSITÉ PARIS 8 - VINCENNES À SAINT-DENIS

Master Informatique des Systèmes Embarqués

Memoire de projet tuteuré

Fakhri YAHIAOUI - Roman BOURSIER

Date de soutenance : le 09/06/2020

Tuteur – Université : Farès BELHADJ

Résumé

Le présent document présente notre travail effectué dans la cadre de notre mémoire de fin d'étude.

Dans un premier temps, nous présentons le projet et son contexte, puis nous abordons les différentes solutions existantes à travers la section "état de l'art". Nous exposons ensuite nos propositions de solutions et leurs évaluations respectives. En conclusions nous faisons le bilan de ce travail ainsi que les points d'amélioration possibles de nos modèles.

Table des matières

Résumé	1
Introduction	4
0.1 Contexte	4
1 Etat de l'art	6
1.1 Généralités sur les GANs	6
1.2 CGANs et traduction d'image	7
1.2.1 Pix2Pix	7
1.2.2 GauGan	7
1.2.3 CycleGan	7
1.2.4 Transfert de style neuronal	8
1.2.5 Learning to Sketch	8
1.2.6 Sketching : Inferring Contour Drawings from Images .	9
2 Résultats	10
2.1 Généralités	10
2.1.1 Implémentations	10
2.1.2 Datasets	10
2.1.3 Evaluations	11
2.2 Abstraction vers peinture	12
2.2.1 Evaluation	12
2.2.2 Résultats	13
2.3 Abstraction vers photo	14
2.3.1 Evaluation	14
2.3.2 Résultats	14
2.4 Photo vers peinture	16
2.4.1 Evaluation	16
2.4.2 Résultats	16
2.5 Temps d'exécutions	18

Introduction

Nous souhaitons utiliser un modèle de Deep Learning, afin de produire un moteur de rendu capable d'adopter une stylisation « type » tel que la peinture chinoise. Dans un premier temps, il s'agira de proposer un modèle d'abstraction des peintures sélectionnées comme base d'apprentissage et d'utiliser le couple « peinture originale » / « abstraction » pour l'entraînement. Par la suite, un moteur de rendu d'abstractions sera connecté au réseau profond qui produira une peinture sur la base de l'abstraction.

Le modèle généré devra d'une part adopter la stylisation retenue mais aussi interpréter l'abstraction d'origine.

0.1 Contexte

La "traduction image-image" (Image-to-image translation) permet d'apprendre le mapping entre une image d'entrée et de sortie. En figure 1, nous testons la génération d'un paysage à partir d'un croquis simple. En figure 2, c'est un échec.

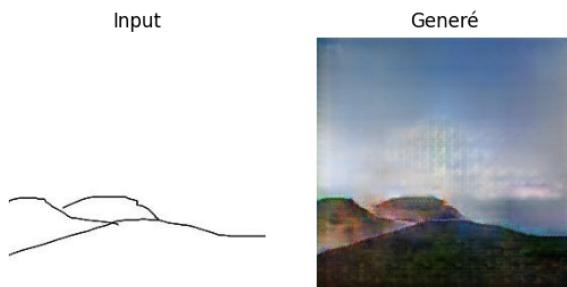


FIGURE 1 – Test du framework pix2pix [IZZE16] sur notre dataset composé de photos de paysages, labellisées en appliquant un filtre Canny [Can86] sur chacune d'elles.

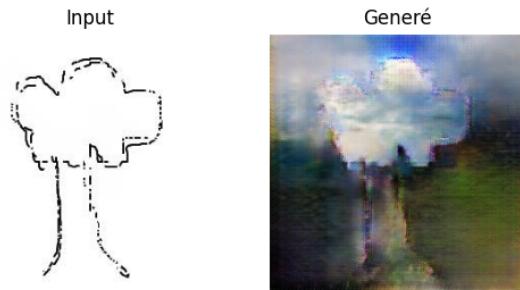


FIGURE 2 – Utilisation du même modèle avec une abstraction d'arbre

Comme évoqué dans [SPS⁺18] les algorithmes du type "traduction image-image" se basent essentiellement sur la corrélation d'une image à l'autre, et relève d'un apprentissage supervisé. Le rendu en figure 2 s'explique par la nature du dataset et par la distance importante qui sépare une abstraction d'une photo.

En figure 3 nous avons demandé à plusieurs personnes de dessiner un paysage composé de montagnes et d'arbres, éléments courants de la peinture chinoise. Ces dessins sont des abstractions, que nous souhaitons traduire en peintures chinoises.

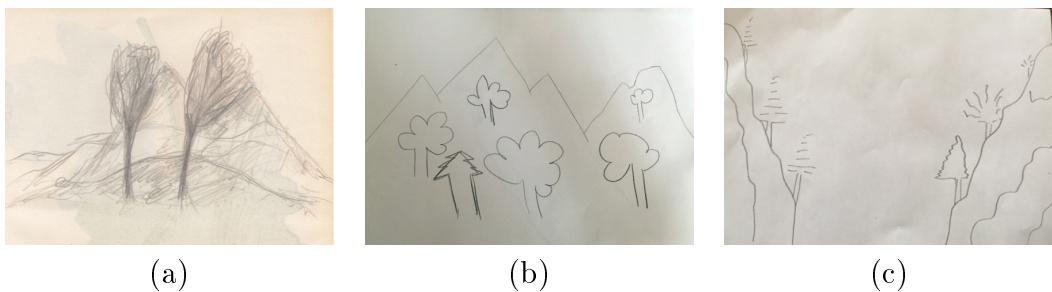


FIGURE 3 – Dessin de paysages réalisés par des personnes différentes. On note que les niveaux d'informations sont plus ou moins élevés. Le dessin (b) est très abstrait tandis que nous observons plus de détails pour (a) .

Chapitre 1

Etat de l'art

Nous présentons dans un premier temps les GANs qui sont au coeur de notre problématique. En un second lieu, nous étudions différents modèles existants basés sur les GANs conditionnels.

1.1 Généralités sur les GANs

"Les GANs (en anglais generative adversarial networks) sont une classe d'algorithmes d'apprentissage non-supervisé. Ces algorithmes ont été introduits dès 2014 par Goodfellow et permettent de générer des images avec un fort degré de réalisme." [Wik20]

Un générateur fabrique des données et les soumet au discriminateur dont le but est d'évaluer leurs degrés de crédibilité.

Le générateur $G(z, \theta_1)$ représente un réseau de neurones capable de mapper du bruit z vers l'espace désiré x . Le discriminateur $D(z, \theta_2)$ retourne la probabilité dans l'intervalle $[0, 1]$ que x vient du dataset original. θ_i représente les poids définis par chacun des modèles. Le générateur tente de maximiser la probabilité que les données x , soient classifiées comme appartenant au dataset d'origine et inversement, le discriminateur minimise la probabilité que de fausses images appartiennent au dataset d'origine.

Il existe aujourd'hui une très grande variété de travaux de recherches basés sur les GANs (BCGAN, AmbientGAN, ORGAN, Perceptual GAN). Le dépôt "The GANs Zoo" [hin18] référençait déjà en 2018 plus de 502 noms de GANs différents !

1.2 CGANs et traduction d'image

Les GANs conditionnels ajoutent une information supplémentaire y , partagée par le discriminateur et le générateur. Grâce aux cGANs il est possible de générer des images réalistes basées sur des labels de classes, des textes ou des images.

1.2.1 Pix2Pix

[GEB15] nous montre en quoi les cGANs peuvent permettre de résoudre efficacement les problèmes de traductions d'images et propose un framework applicable à n'importe quel domaine.

1.2.2 GauGan

[PLWZ19] permet de générer des paysages réalistes basés sur des masques de segmentation sémantiques. Les auteurs introduisent la "normalisation conditionnel" ou "adaptative", permettant de prendre en compte notamment les informations spatiales. Les couches de normalisations ont tendance à faire perdre de l'information contenu dans les masques sémantiques d'entrées car ils ne dépendent pas de données externes.



(a) Input - masques de segmentation sémantiques



(b) Output

FIGURE 1.1 – Test de rendu de paysage à partir de l'application Gaugan : <http://nvidia-research-mingyliu.com/gaugan/>

1.2.3 CycleGan

CycleGan [ZPIE17], présente une approche pour la traduction d'une image d'un domaine source vers un domaine cible lorsque le dataset n'est pas apparié.

Le modèle possède deux générateurs $G : X \rightarrow Y$ et un second $F : Y \rightarrow X$ ainsi que deux discriminateurs D_Y and D_X

Les auteurs nous expliquent que si nous pouvons passer du domaine X vers le domaine Y et vice-versa, alors le résultat final devrait être identique à l'entrée initiale X .

1.2.4 Transfert de style neuronal

Introduit par Leon A. Gatys [GEB15] le TDN consiste à transférer un style à partir d'une image de référence vers une image de contenu. L'objectif est de transformer l'image d'entrée (bruit) en minimisant la distance avec l'image de contenu et avec l'image de style. On obtient alors par rétropropagation, une image qui correspond au contenu de l'image d'origine et au style souhaitée.

L'avantage de cette technique est qu'elle ne nécessite pas de dataset, seules deux images sont nécessaires.

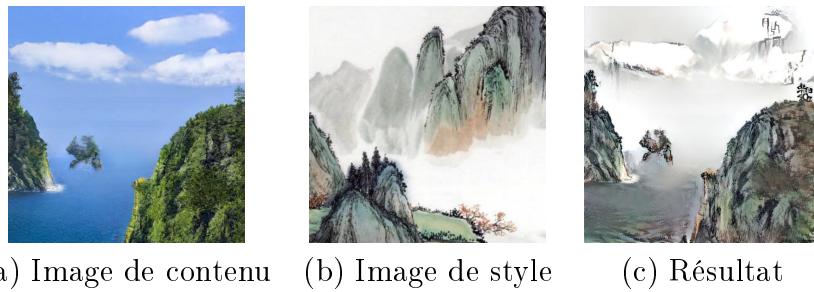


FIGURE 1.2 – Test de transfert de style réalisé à partir de la figure 2.8(b)

1.2.5 Learning to Sketch

[SPS⁺18] présente une méthode permettant de transformer une photo en croquis, en essayant d'imiter la façon de faire d'un humain. Le modèle est capable de réaliser le croquis séquentiellement, c'est-à-dire trait par trait. Les auteurs proposent de résoudre le problème des styles subjectifs et variés des dessins fait à la main en utilisant un modèle hybride supervisé/non-supervisé. L'objectif étant de palier "au signal de supervision faible et bruyante" induit par l'écart important entre un croquis et sa photo correspondante.

L'architecture est décomposé en 4 sous-modèles contenant chacun leurs propres sous-réseaux d'encodeurs et de décodeurs. Deux réseaux supervisés traduisent respectivement une photo en croquis $D(E(photo)) \rightarrow sketch$ et un croquis en photo $(D(E(sketch)) \rightarrow photo)$. Deux autres réseaux non-supervisés se chargent de la reconstruction. $D(E(photo)) \rightarrow photo$ et

$$D(E(\text{sketch})) \rightarrow \text{sketch}$$

1.2.6 Sketching : Inferring Contour Drawings from Images

[LLM⁺19] propose une nouvelle approche concernant la détection des contours dans une image. L'article montre que les solutions traditionnelles comme Canny [Can86], captent uniquement les signaux de haute fréquence dans l'image sans la comprendre. Les auteurs ont collecté un dataset de 5000 pairs "croquis humain/photos", créé manuellement via la plateforme de crowdsourcing "Amazon Mechanical Turk". En effet aucun dataset existant ne convenait (nombre d'éléments dans l'image, limites internes manquantes, le contenu non reconnaissable, les zones ombrées vides etc ..).

Ce modèle également du type "traduction image-image", permet d'avoir plusieurs labels différents pour la même photo, donc plusieurs interprétations différentes.

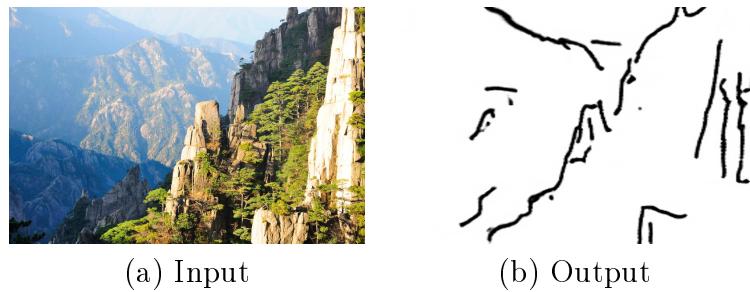


FIGURE 1.3 – Test du modèle pré-entraîné sur une photo de paysage

Chapitre 2

Résultats

2.1 Généralités

2.1.1 Implémentations

Nous proposons dans les sections suivantes un ensemble de tests effectués selon des configurations différentes. Nous avons retenu les cGANs et en particulier plusieurs implémentations des cycleGan et de pix2pix.

Chaque étapes de l'apprentissage consiste d'abord à sélectionner un lot d'exemples réels, puis à utiliser le générateur pour générer de faux échantillons correspondants. Le discriminateur est ensuite mis à jour avec le lot d'images réelles, puis de fausses images. Ensuite, le générateur est mis à jour en fournissant les images sources réelles comme entrée et les prédictions de sortie du discriminateur.

Le générateur possède deux scores de pertes ainsi que le score de somme pondérée retourné de l'appel. Ce score pondéré est utilisé pour mettre à jour les poids du modèle.

2.1.2 Datasets

Malgré nos recherches, nous n'avons pas trouvé de datasets correspondants exactement à nos besoins, c'est-à-dire des paires d'abstractions/peintures chinoises. Pour les croquis, les plus grosses bases existantes sont TU-Berlin [EHA12], Sketchy [SBHH16] et Quickdraw. Leurs utilisations est délicate car il s'agit le plus souvent d'objets isolés et non appairés à une image réaliste. Pour la peinture chinoise, nous avons récupéré un premier dataset de 5000 images ([ych18]) et scrappés des plateformes comme google/baidu et pinterest. Chaque dataset est répartie à 90% pour l'entraînement et 10% pour les tests. Nous avons ensuite réduit le nombres d'exemples pour des

raisons de temps et d'apprentissage trop longs. Chaque image a été par la suite redimensionnée et recadrée au format 256x256.

2.1.3 Evaluations

En général, les modèles GAN ne convergent pas ; au lieu de cela, un équilibre est trouvé entre le générateur et les modèles discriminateurs.

```
>1, d1[0.505] d2[0.651] g[75.595]
>2, d1[0.479] d2[0.516] g[69.595]
>3, d1[0.621] d2[0.478] g[75.868]
>4, d1[0.478] d2[0.538] g[65.794]
>5, d1[0.415] d2[0.425] g[70.626]
```

- d1 : La perte de discrimination sur des exemples réels
- d2 : La perte de discrimination sur des exemples générés ou faux
- g : La perte du générateur (moyenne pondérée des pertes liées à l'affrontement de d1 et d2).

Par conséquent, nous ne pouvons pas juger facilement quand la formation devrait cesser. Nous avons donc enregistrer le modèle et généré des exemples de traductions image-à-image périodiquement pendant la formation. Les rendus peuvent être évalués à la fin de l'analyse et utilisés pour sélectionner un modèle de générateur final basé sur la qualité d'image générée.

2.2 Abstraction vers peinture

L'idée ici est de décomposer le passage de l'abstraction à la peinture en deux étapes, c'est à dire de l'abstraction au dessin détaillé puis du dessin détaillé vers la peinture chinoise.

L'entraînement utilise l'architecture pix2pix décrites dans [GEB15], implémentée sous keras 2.4.0.

Le premier dataset de 654 images a été labellisé manuellement, et le second a été obtenu en appliquant l'algorithme de détection de contours nommé "holistically-nested edge detection" [XT15] sur les peintures chinoises. HED est basé sur un modèle d'apprentissage profond et permet de mieux résoudre l'ambiguïté liée à la détection des contours et des objets.

2.2.1 Evaluation

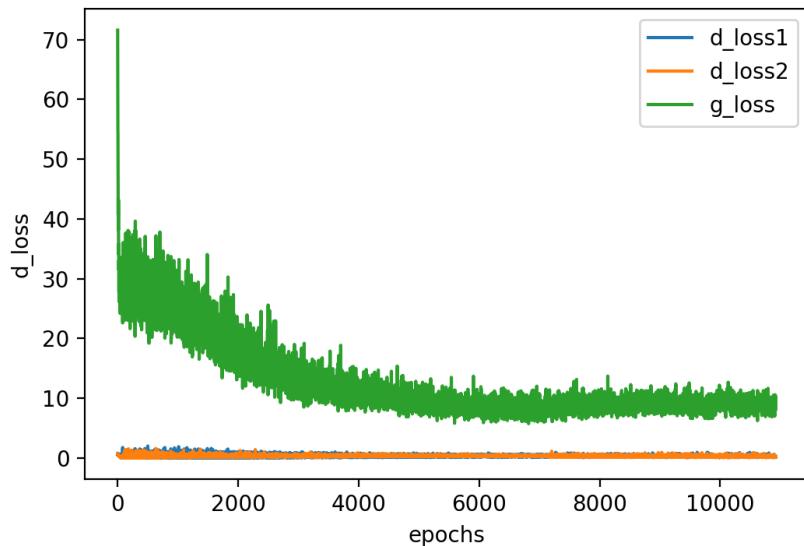


FIGURE 2.1 – Scores de pertes pendant l'entraînement 2.8(b)

2.2.2 Results

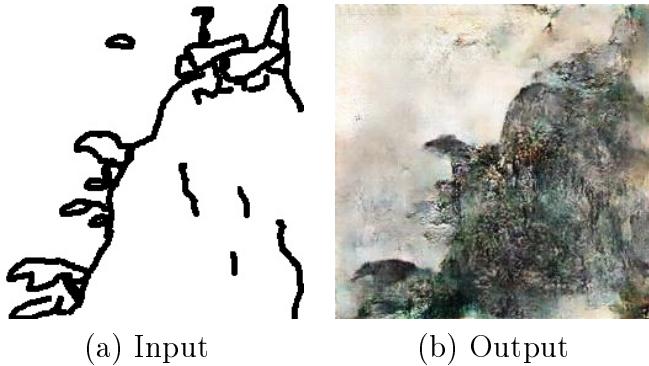


FIGURE 2.2 – Abstraction vers peinture



FIGURE 2.3 – Abstraction vers peinture 2.8(b)

2.3 Abstraction vers photo

Au départ, nous avions directement fait nos tests sur des pairs croquis/-photos. Les résultats étant décevants, nous avons alors avons tenté un nouvel apprentissage en générant les abstractions via un filtre Canny. L'apprentissage à été effectué grâce à un dataset composé de 583 pairs canny/photos et une implémentation de pix2pix sous pytorch 0.4.1.

2.3.1 Evaluation

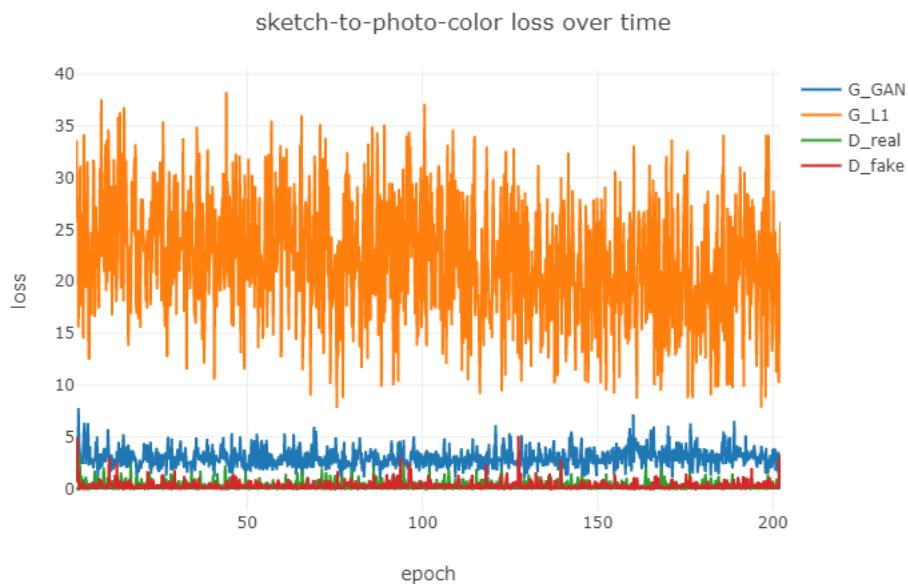


FIGURE 2.4 – Scores de pertes pendant l'entraînement 2.8(b)

2.3.2 Résultats

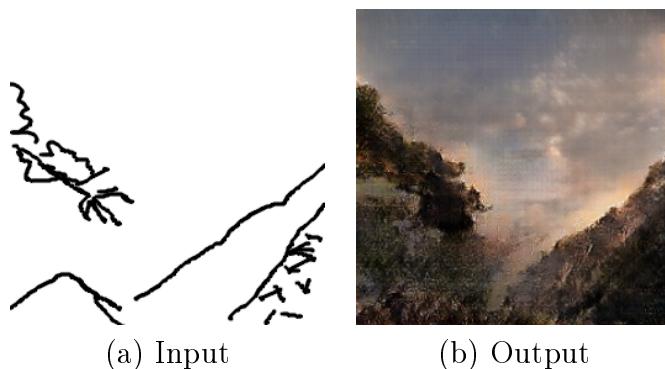


FIGURE 2.5 – Test du modèle pré-entraîné sur une photo de paysage

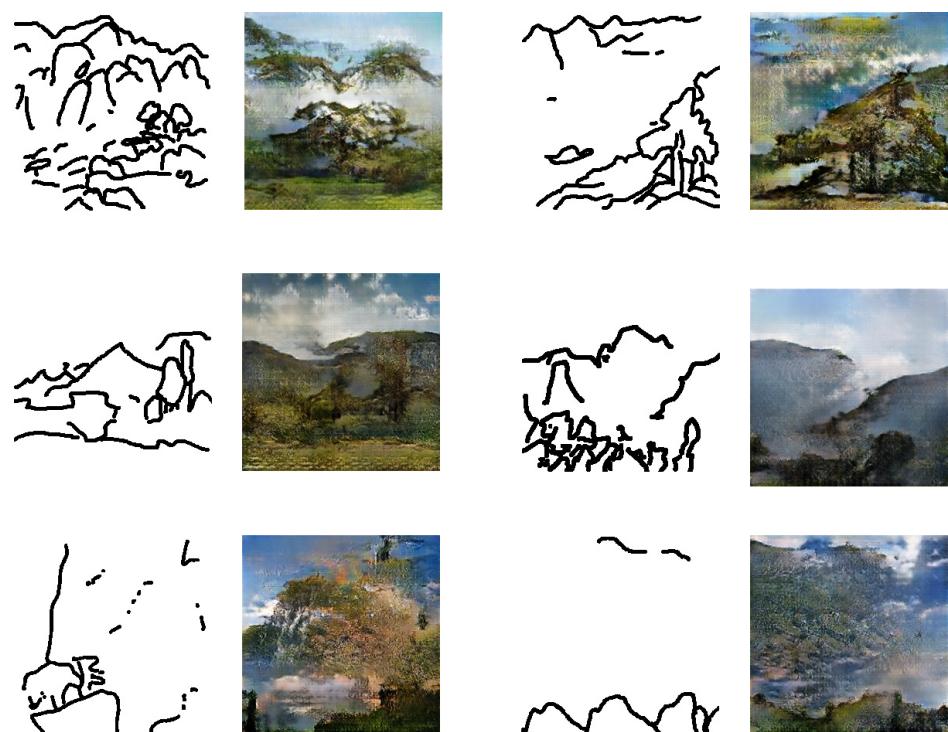


FIGURE 2.6 – Test du modèle pré-entraîné sur des photos de paysages 2.8(b)

2.4 Photo vers peinture

Pour ce dernier test, nous avons utilisé une implémentation des cycle GANs, ce qui nous a permis d'utiliser un dataset non appairé de 590 peintures chinoises et 583 photos. Cette solution est très utilisée pour la génération de peintures à partir d'une image déjà réaliste. L'idée pour nous est d'appliquer le style propre aux peintures chinoises sur des photos n'appartenant pas à ce domaine.

Pour plus de rapidité nous avons réduit le nombre de d'éléments du dataset ce qui a un impact significatif sur les résultats. Malheureusement, l'entraînement fut interrompu à la 45ème epochs en raison de restrictions sur google colab.

2.4.1 Evaluation

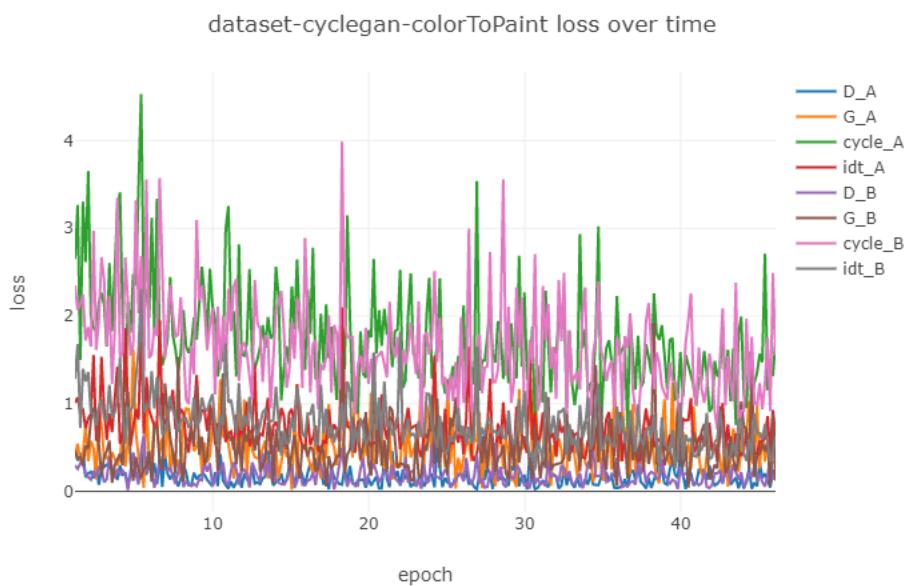


FIGURE 2.7 – Scores de pertes pendant l'entraînement 2.8(b)

2.4.2 Résultats

Malgré un rendu plutôt réussi, on constate une application des couleurs parfois hasardeuse et incohérentes. Cela est probablement dû au nombre limité d'epochs.



(a) Input (b) Output

FIGURE 2.8 – Application du modèle sur une photo

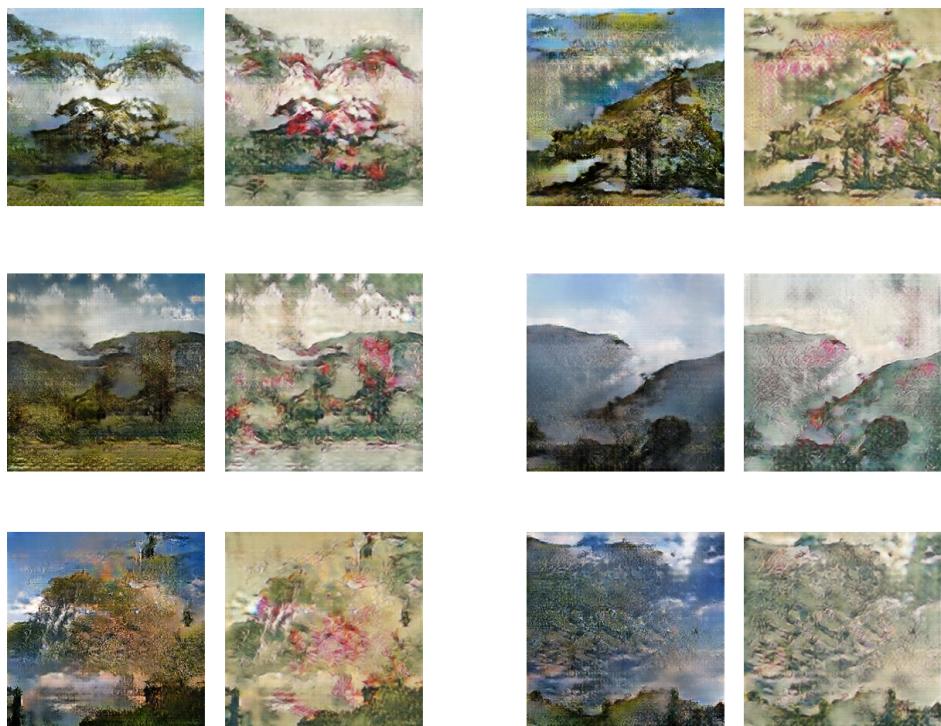


FIGURE 2.9 – Application du modèle sur une photo 2.8

2.5 Temps d'exécutions

Afin d'évaluer la possibilité d'une application temps réel,nous avons testé le modèle de la section "Abstraction vers photo" sous google collab en mode GPU (Tesla K80) sur des images 256*256px. Seulement les prédictions sont pris en compte et non le chargement des images et du modèle.

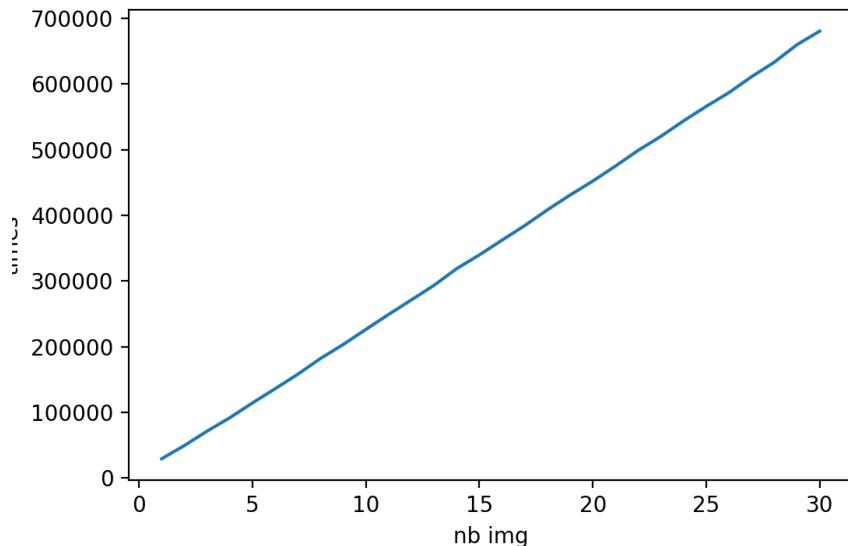


FIGURE 2.10 – Temps d'exécutions - GPU - Tesla K80 2.8(b)

Le modèle est donc capable de générer 30 image en 0.7 sc ce qui semble un bon score. A terme, il serait intéressant de regrouper nos deux modèles (abstraction hed -> hed peinture), pour gagner davantage en performance.

Chapitre 3

Conclusion et Perspectives

L'ensemble de ces expériences nous montrent que dans le cadre d'un transfert "image-image", il semble impossible de générer directement une peinture crédible depuis une abstraction. Cet état de faite est principalement lié à l'absence de datasets conséquents. En revanche nous avons obtenu de meilleurs résultats en séparant les étapes d'apprentissages, malgré des datasets réduits.

Dans tous les cas, le rendu final donnera des résultats acceptables que si l'entrée appartient au même thème que la sortie, dans notre cas un paysage. Nous n'avons à jour pas trouvé d'exemples permettant de prouver le contraire. Nous avons aussi remarqué que plus y a de variations et de disparités dans le dataset, moins la prédiction est convaincante.

Théoriquement, nos résultats pourraient être améliorés en augmentant le nombre d'éléments dans le dataset ainsi que le nombre d'epochs, ce qui nécessite des ressources de calculs plus importantes.

Le passage d'un paysage abstrait à la peinture semble donc possible à condition de trouver une méthode de génération d'abstraction automatique pour ainsi augmenter la taille du dataset. Générer ces abstractions veut dire ne garder que les informations minimum et utiles à la compréhensions de l'image. C'est pourquoi la méthode de l'instance segmentation et/ou de l'utilisation d'un algorithme d'approximation d'image auraient mérité d'être approfondis.

Bibliographie

- [Can86] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6) :679–698, 1986.
- [EHA12] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects ? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4) :44 :1–44 :10, 2012.
- [GEB15] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015.
- [hin18] hindupuravinash. The gan zoo. <https://github.com/hindupuravinash/the-gan-zoo/>, 2018.
- [IZZE16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.
- [LLM⁺19] Mengtian Li, Zhe Lin, Radomír Mech, Ersin Yumer, and Deva Ramanan. Photo-sketching : Inferring contour drawings from images. *CoRR*, abs/1901.00542, 2019.
- [PLWZ19] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. *CoRR*, abs/1903.07291, 2019.
- [SBHH16] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database : Learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2016.
- [SPS⁺18] Jifei Song, Kaiyue Pang, Yi-Zhe Song, Tao Xiang, and Timothy M. Hospedales. Learning to sketch with shortcut cycle consistency. *CoRR*, abs/1805.00247, 2018.
- [Wik20] Wikipedia. Réseaux antagonistes génératifs — Wikipedia, the free encyclopedia. <http://fr.wikipedia.org/w/index.php?title=R%C3%A9seaux%20antagonistes%20g%C3%A9n%C3%A9ratifs&oldid=9000000>

- %C3\\%A9ratifs&oldid=170457375, 2020. [Online ; accessed 17-May-2020].
- [XT15] Saining Xie and Zhuowen Tu. Holistically-nested edge detection, 2015.
- [ych18] ychen93. The gan zoo. <https://github.com/ychen93/Chinese-Painting-Dataset>, 2018.
- [ZPIE17] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.