

---

# Décomposition d'images superposées

---

AUTEUR

Ragot Cyrian

2025-06-09

---

**Contexte** Ce travail a été produit dans le cadre des travaux pratiques du cours de Traitements de Données Audio-Visuelles tenu à l'INP-ENSEEIHT en filière Sciences du Numérique spécialité Images et Multimédia.

**Introduction** L'objectif de ce papier est de tenter de restaurer une image qui a été doublement exposée. Une image superposée est produite à partir de deux images, souvent par double exposition sur un capteur photographique. C'est d'ailleurs une technique artistique récurrente en photographie. Cependant, on peut vouloir retrouver les images d'origines qui ont été superposées sur une photographie, par exemple, il est possible de réaliser une double exposition sur un capteur d'un vieil appareil photo par inadvertance. On peut citer d'autres applications telles que la séparation d'un reflet, d'une ombre ou d'un watermark d'une image.

Vous pourrez retrouver les notebooks et scripts python sur le repository GitHub que j'ai créé pour ce projet, voir [1].

L'objectif principal est de tenter de restaurer la photographie qui a été fournie en cours, voir figure 1, mais aussi et surtout de découvrir les techniques de séparation d'images par apprentissage profond. On peut faire un parallèle avec le TP 12 sur la séparation de sources audio.



FIGURE 1 – Photographie à décomposer

La photographie de la figure 1 semble être une erreur d'exposition, en effet, avec un appareil argentique qui n'est pas doté d'un mécanisme automatique pour faire avancer le film de la pellicule, l'utilisateur peut oublier d'avancer le film lui-même.

---

**Création de la base de données** Pour créer un jeu de données pour notre modèle, nous allons artificiellement créer des doubles expositions avec des photographies normales. Sur une pellicule, le fait de doublement exposer la photographie correspond à une simple addition de la lumière reçue lors des deux prises, on modélise alors la double exposition de deux images normales comme une superposition de ces images en appliquant une transparence de 0.5 sur chacune. Il nous suffit donc d'avoir beaucoup d'images dont nous ferons des superpositions pour obtenir un bon jeu de données. L'avantage pour ce problème est qu'on n'aura pas besoin d'annoter les images car nous avons directement la vérité terrain, à savoir les images avant superposition.

Pour choisir les images que l'on va superposer pour notre base de données, il est raisonnable de penser qu'on obtiendra de bons résultats avec des photographies qui contiennent des objets similaires à ceux présents dans la figure 1, c'est-à-dire des personnes dans des lieux/situations différentes. Les images de la base de données PISC (People in Social Context) [2] [3] semblent adaptées à notre problème, quelques exemples sont donnés dans la figure 2.



FIGURE 2 – Images provenant de la base de données PISC

---

Un ”petit” script python `create_dataset.py` me permet de prendre des images normales aléatoirement parmi celles de PISC, de les cropper de manière aléatoire à une taille voulue, de créer les superpositions entre images croppées et de les séparer en un set d’entraînement, un set de validation et un set de test. Le script permet entre autre de choisir combien de crop par image on souhaite effectuer et combien de superpositions une seule image va subir avec d’autres images afin de faire de l’augmentation de données. Dans toute la suite j’ai choisi de mettre ces valeurs à 1 car je n’ai même pas la puissance de calcul pour utiliser l’entièreté des images de PISC donc l’augmentation de données n’est pas envisageable. Pour la suite, les images ont été croppées à une taille 128x128, qui est petite mais nécessite moins de ressources.

**Premier modèle** Ma première approche a été de créer un simple modèle de type CNN en enchaînant des couches de convolution et de max pooling pour réduire la dimension puis des couches de convolution inverses de manière à revenir à la dimension des images, ou presque. Une entrée dans le réseau est une image superposée de dimension 128x128x3 (RGB) mais la sortie du réseau correspond à deux images donc est de taille 128x128x6. Ces deux images de sortie sont comparées à la vérité terrain, c’est-à-dire aux deux images initiales qui ont servies à créer l’image d’entrée superposée, en minimisant la cross-entropie binaire comme fonction objectif. En faite, sans trop le savoir, j’avais re-imaginé l’architecture des U-Net. Ce premier modèle a été écrit dans le notebook `image_dec.ipynb`.

Le premier entraînement a duré 3 :30 min, en fixant la taille des batch à 8 et avec 50 epochs et en utilisant un set d’entraînement de 7000 images et un set de validation de 1500 images. On peut analyser la courbe de la loss durant l’apprentissage sur la figure 3. Cette courbe montre un entraînement très mauvais, la loss est descendue en quasiment une seule epoch puis est restée constante. Les exemples sur des donnée de test de la figure 4 montrent que les images prédites sont en tous points similaires à l’image superposée : on a rien appris.

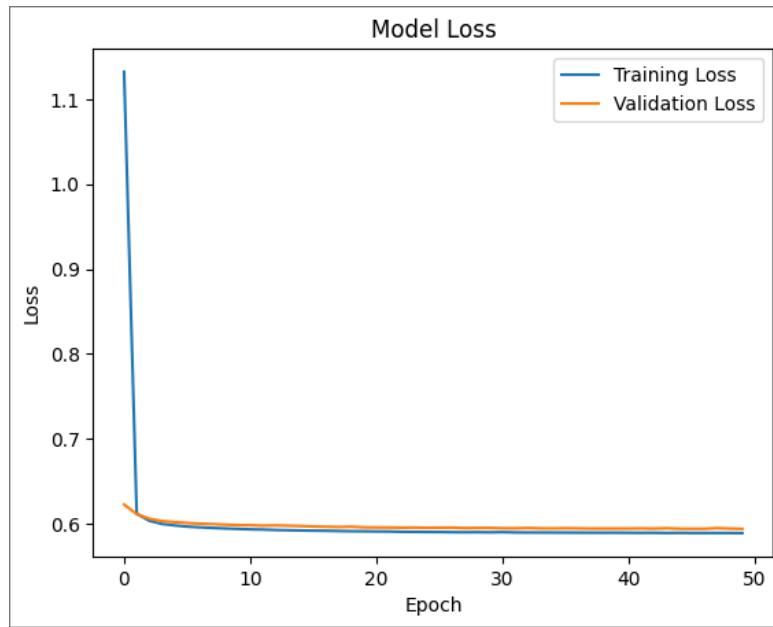


FIGURE 3 – Fonction de coût pendant l’entraînement

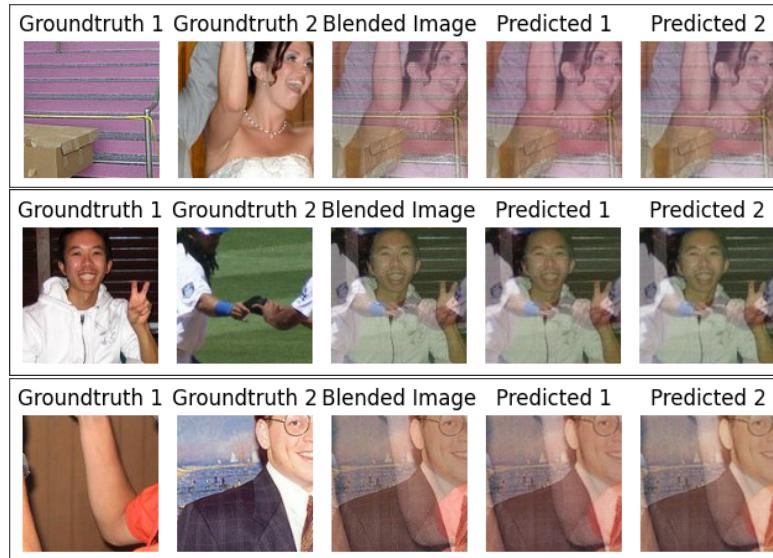


FIGURE 4 – Résultats de décomposition d’images superposées par apprentissage avec le premier modèle

D’une certaine façon, on a appris au modèle à copier-coller une image. Un point intéressant est qu’on pourrait trouver une application de ce modèle à la compression d’images. En effet, la par-

---

tie de réduction de dimension du réseau, encore appelée encodeur, permet de réduire l'information nécessaire pour stocker l'image et la seconde partie du réseau, appelée décodeur permet de revenir à l'image d'origine avec une ressemblance très forte.

**Quelques références et lectures** Pour tenter d'améliorer mes résultats, j'ai pu lire plusieurs papiers qui traitent de la décomposition d'images. En particulier, des architectures de génération d'image tels que les cGAN (conditional Generative Adversarial Network) comme pix2pix présenté dans [4] ou CycleGan dans [5] utilisent aussi des U-Net et sont performants pour diverses tâches de conversion d'image à image. Les mêmes auteurs ont aussi publié [6], une extension de leurs premiers papiers qui traite spécifiquement de la décomposition d'images superposées. Enfin, j'ai aussi découvert un autre framework de décomposition d'images superposées utilisant de l'apprentissage non-supervisé dans [7]. Ces papiers semblent être l'état de l'art en matière de décomposition d'images superposées, je n'ai pas pu trouver d'autres recherches à ce sujet.

Un point essentiel que j'ai pu apprendre avec [6], est le choix de la fonction de coût. En effet, il est précisé dans les paragraphes 4.2 et 3.1 qu'une fonction de coût classique comme la cross-entropie binaire ne peut pas fonctionner, le modèle aura tendance à moyenner les deux sorties car on ne peut pas spécifier dans l'entraînement l'ordre des deux images prédictes. Ceci explique donc mon problème du premier modèle. Finalement, [6] propose une fonction de coût nommée "crossroad 11 loss function" que j'ai implémentée pour la suite. J'ai aussi remarqué qu'ils utilisaient une taille de batch de 2 et plus d'epochs donc j'ai aussi modifié ces paramètres.

**Second modèle** Comme discuté précédemment, j'ai pu améliorer mon modèle dans un notebook nommé `image_dec_v2.ipynb`. Le nombre d'epochs a été fixé à 100, la taille de batch à 2 et j'ai utilisé un plus grand nombre de données, soit 11900 images pour le set d'entraînement et 2550 pour le set de validation. L'entraînement a duré 23 minutes. Quelques résultats sont présentés sur la figure 6 et la variation de la fonction de coût est donnée dans la figure 5.

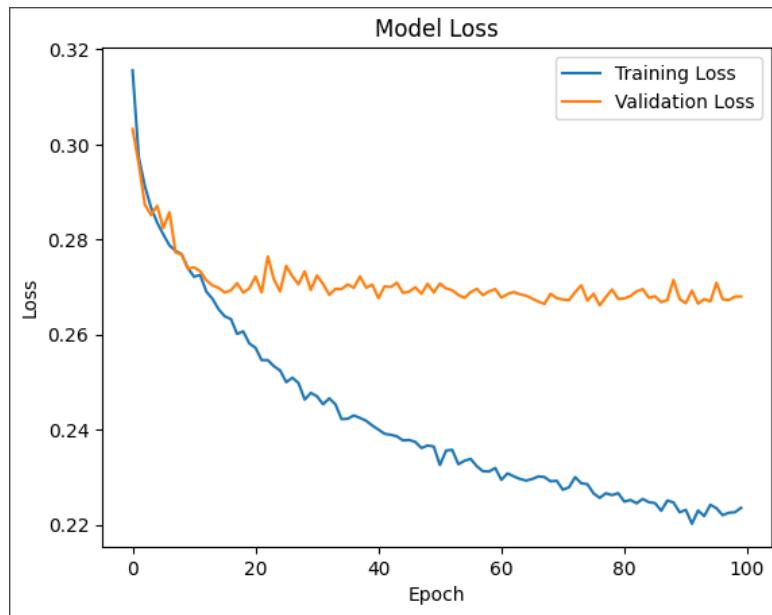


FIGURE 5 – Fonction de coût pendant l’entraînement

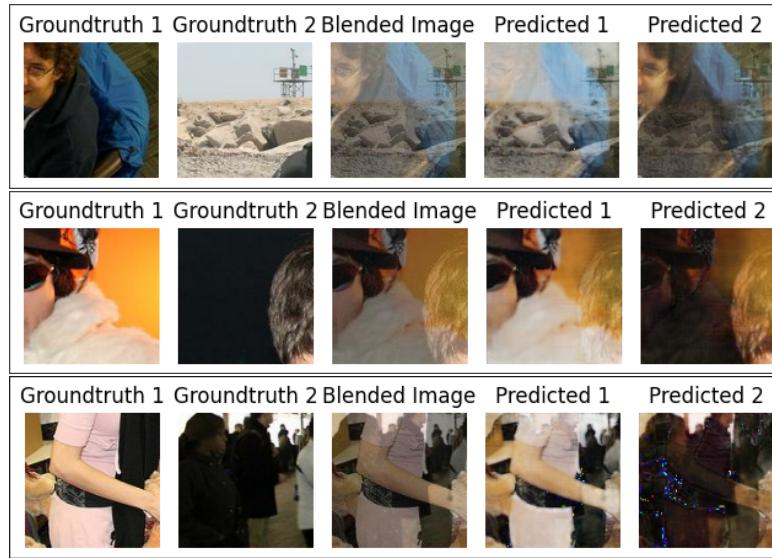


FIGURE 6 – Résultats de décomposition d’images superposées par apprentissage avec le deuxième modèle

Les résultats montrent maintenant des changements dans les images prédictives, on avance ! Mais on est toujours loin de réussir à faire de la bonne décomposition d’images superposées, les résultats

---

présentés sont les quelques meilleurs que j'ai pu obtenir mais dans l'ensemble, les résultats ne sont pas satisfaisants du tout. La courbe de la fonction coût est mieux que dans le premier entraînement, on a une descente progressive cette fois-ci. Cependant, on remarque clairement que notre modèle sous-apprend par l'écart entre la fonction coût sur l'entraînement et celle sur la validation. On en déduit que notre modèle n'est pas assez complexe et sûrement aussi qu'on n'a pas utilisé suffisamment de données pour l'entraîner.

On peut quand même tenter d'appliquer notre modèle à l'image cible. Cependant notre modèle prend des images 128x128 en entrée donc il a fallu redimensionner l'image et le résultat n'est pas génial sur la figure 7. Nous présentons aussi un résultat obtenu en prédisant la décomposition sur plusieurs patchs de l'image cible de tailles 128x128 sur la figure 8. Il faudrait ensuite raccorder les images pour améliorer ce dernier résultat, par exemple avec des méthodes variationnelles.

Le résultat final est approximatif, notre modèle semble plus doué pour séparer des composantes sombres de composantes claires dans les images mais rien de bien incroyable. De plus, dans les zones très sombres, le modèle fait apparaître des artefacts étranges dont la cause m'est inconnue.



FIGURE 7 – Résultat de la décomposition de la photographie avec le deuxième modèle et redimensionnement



FIGURE 8 – Résultat de la décomposition de la photographie avec le deuxième modèle et patchs

**Autres tentatives** J'ai pu tenté d'utiliser les implémentations de [6] et de [7] que j'ai dû modifier en partie pour les faire correspondre à mon problème. Ainsi, j'ai pu lancer ces modèles sur les images de ma base de données et obtenir les résultats de figure 9 et figure 10. Le résultat de la figure 9 commence à faire apparaître des zones plus ou moins bien décomposées, certains objets précis se retrouve mieux dans l'une ou l'autre des deux images décomposées.



FIGURE 9 – Résultat de la décomposition de la photographie avec [6]



FIGURE 10 – Résultat de la décomposition de la photographie avec [7]

**Conclusion** Le problème est très mal posé et malgré pas mal de recherche, il ne semble pas avoir de solution satisfaisante pour le moment. Cependant, les résultats obtenus laissent un espoir de voir un jour une méthode générale pour la décomposition d'images superposées. En particulier, [6] permet déjà d'obtenir de très bon résultats pour des tâches de décomposition un peu plus précises et avec des images superposées plus simples.

## Table des figures

1	Photographie à décomposer . . . . .	1
2	Images provenant de la base de données PISC . . . . .	2
3	Fonction de coût pendant l'entraînement . . . . .	4
4	Résultats de décomposition d'images superposées par apprentissage avec le premier modèle . . . . .	4
5	Fonction de coût pendant l'entraînement . . . . .	6
6	Résultats de décomposition d'images superposées par apprentissage avec le deuxième modèle . . . . .	6
7	Résultat de la décomposition de la photographie avec le deuxième modèle et redimensionnement . . . . .	8
8	Résultat de la décomposition de la photographie avec le deuxième modèle et patchs .	9
9	Résultat de la décomposition de la photographie avec [6] . . . . .	10
10	Résultat de la décomposition de la photographie avec [7] . . . . .	11

## Références

- [1] C. Ragot, “Image decomposition.” <https://github.com/cyrianR/image-decomposition>, 2025. GitHub repository. Accessed : 2025-06-08.
- [2] J. Li, Y. Wong, Q. Zhao, and M. S. Kankanhalli, “Dual-glance model for deciphering social relationships,” *arXiv preprint arXiv :1708.00634*, 2017. Accepted at IEEE International Conference on Computer Vision (ICCV), 2017.
- [3] J. Li, Y. Wong, Q. Zhao, and M. S. Kankanhalli, “People in social context (pisc) dataset.” <https://zenodo.org/records/1059155>, 2017. Version v5, Published July 19, 2017. A dataset for social relationship recognition, people detection, and occupation recognition.
- [4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *arXiv preprint arXiv :1611.07004*, 2017. Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
- [5] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” *arXiv preprint arXiv :1703.10593*, 2017. Extended version of the ICCV 2017 paper.
- [6] Z. Zou, S. Lei, T. Shi, Z. Shi, and J. Ye, “Deep adversarial decomposition : A unified framework for separating superimposed images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12806–12816, 2020.
- [7] Y. Gandelsman, A. Shocher, and M. Irani, “”double-dip” : Unsupervised image decomposition via coupled deep-image-priors,” 6 2019.