REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université de Mohamed El-Bachir El-Ibrahimi - Bordj Bou Arreridj

Faculté *des Sciences et de la technologie*

Département d'Electronique

Mémoire

FILIERE : **Télécommunication**

Spécialité : système de télécommunication

Par

- **Tebbi manel**
- **Akmoum Rania**

Intitulé

Algorithmes d'amélioration d'image basés sur L'apprentissage en profondeur

Présenté le : ……

Devant le Jury composé de :

Nom & Prénom	*Grade*	*Qualité*	*Etablissement*
Dr .sid ahmed soumia	*…*	*Co encadreur*	*Univ-BBA*
pr.MESSALI Zoubeida	*…*	*Encadreur*	*Univ-BBA*

Année Universitaire 2022/2023

Remerciements

En préambule à ce mémoire nous remerciant ALLAH qui nous aidé et nous a donné la patience et le courage durant ces langues années d'étude.

Je voudrai exprimer mes remerciements à mes encadrantes Prof MESSALI Zoubeida et dr .sid ahmed soumia d'avoir accepté de diriger ce travail.

Un grand remerciement aux membres du jury: Monsieur ….D'avoir accepté la présidence du jury, je veux profiter l'occasion pour vous remercier infiniment et vous dire merci vivement pour vos conseils durant mes premières années.

Et enfin, un grand merci à tous ceux et celles qui m'ont aidé de près ou de loin à l'élaboration de ce travail.

Dédicaces

A mes très chers parents, source de vie, d'amour et d'affection.

À ma belle-famille tous ceux qui, ont contribué de près ou de loin à la
Réalisation de ce modeste travail.

À mon cher mari, pour la patience et le soutien dont il a fait preuve pendant toute la durée de ce travail et à qui je voudrais exprimer mes affections et mes gratitudes. Merci infiniment.

A mes chers frères, source d'espoir et de motivation.

A tous mes amis, tout particulièrement Ghofrane.

A Rania, chère amis avant d'être binôme.

A vous cher lecture.

Manel Tabbi

Je dédie ce travail

A ma famille, elle qui m'a doté d'une éducation digne, son amour a fait de moi ce que je suis aujourd'hui :

Particulièrement à **mon père**, pour le gout à l'effort qu'il a suscité en moi, de par sa rigueur.

A toi **maman** ceci est ma profonde gratitude pour ton éternel amour, que ce rapport soit le meilleur cadeau que je puisse t'offrir.

A vous **mes frères** et **sœurs** qui m'avez toujours soutenu et encouragé durant ces années d'études.

A mon amie **Ghofrane** qui m'est toujours encouragé, et à qui je souhaite plus de succès.

A **Manel** chère amis avant d'être binôme

A mes amis universitaires avec qui j'ai passé le meilleur de mon temps
oumaima, khawla, sihem et **ghozlane**
A tous ceux que j'aime

Rania Akmoum

Table de matière

Listes des tableaux

Listes des figures

Liste des abréviations

AWGN	Additive White Gaussian Noise
SR	Super Résolution
PSNR	Peak Signal to Noise Ration
SSIM	Structural SIMilarity Index
EQM	Erreur Quadratique Moyenne
DL	Deep Learning
ANN	Artificial Neurol Network
CNN	Convolution Neural Network
ReLU	REctified Linear Units
Dncnn	Denoising CNN
MNIST	Mixed national institute of standars and technology

Introduction Générale

Introduction Générale

L'algorithme d'amélioration d'image basé sur l'apprentissage en profondeur utilise des réseaux de neurones convolutifs pour améliorer la qualité des images.
L'algorithme fonctionne en apprenant à partir d'un grand nombre d'images d'entraînement de haute qualité et en utilisant ces connaissances pour améliorer la qualité d'une image donnée.
Le processus commence par la collecte d'un grand nombre d'images de haute qualité pour l'entraînement du réseau de neurones. Le réseau de neurones est ensuite entraîné à partir de ces images en utilisant une technique d'apprentissage supervisé, où l'algorithme est informé de la qualité de l'image d'entrée et de la qualité souhaitée de l'image de sortie.
Une fois l'entraînement terminé, l'algorithme est capable d'améliorer la qualité d'une image en utilisant les connaissances acquises lors de l'entraînement.

L'image d'entrée est d'abord divisée en petites zones appelées patchs.
Chaque patch est ensuite analysé par le réseau de neurones, qui utilise les connaissances acquises lors de l'entraînement pour améliorer la qualité de chaque patch.
Les patchs améliorés sont ensuite fusionnés pour former l'image de sortie améliorée. L'algorithme peut également être conçu pour améliorer des caractéristiques spécifiques de l'image, telles que la netteté ou la luminosité.

Dans le Chapitre, nous allons introduire définition d'amélioration d'image et formulation problématique de débruitage et les type de bruit ….etc

Dans chapitre 2 : nous présentons Les architecture de cnn et DNCNn

Dans le Chapitre 3, nous détaillons l'implémentation des algorithmes considérés d'images ainsi que les architectures des techniques et algorithmes utilisés et les tests que nous avons effectués avec des discussions des résultats. Plus précisément nous allons implémenter une seul algorithmes de débruitage images qui elle DnCNN
Calcul des critères usuels, à savoir le PSNR, SSIM, L'originalité de notre travail réside dans l'implémentation d'algorithmes de débruitage d'images à base de DNCNN

.

Organisation du Manuscrit

Ce manuscrit est constitué de la présente introduction et trois chapitres plus la conclusion générale, qui englobe les remarques pertinentes tirées de cette étude.

Dans le premier chapitre, nous allons évoquer la problématique du débruitage d'image. Nous exposerons les différents types du bruit et leurs modèles ainsi que les différents filtres conventionnels. Nous présenterons à la fin de ce chapitre, l'application du deep learning en filtrage. Dans chapitre 2, nous présenterons les concepts de base de l'apprentissage profond (deep learning . Nous mettrons l'accent sur l'architecture de réseaux convolutifs (Conlutional Neurol Network CNN). Nous détaillerons à la fin, le débruiteur à base de l'architecture CNN le DnCNN que nous allons implémenter dans notre étude de simulation. Dans le Chapitre 3, nous détaillerons l'implémentation du débruiteur DnCNN sous Pytorch. Puis nous exposerons la construction du réseau DnCNN sous Keras. Nous testerons le débruiteur sur un grand nombre d'images tirées des ensembles de données utilisées en deep learning. Nous Calculerons les critères usuels, à savoir le PSNR, SSIM pour démontrer l'efficacité du débruiteur DnCNN ; pour montrer la convergence du débruiteur, l'erreur quadratique moyenne en fonction de nombre d'epochs sera tracée.

Chapitre 1

Algorithmes de Débruitage d'Images

Résumé

Dans ce Chapitre, nous allons détailler les différentes techniques de débruitage d'images ; le bruit étant additif gaussien. L'amélioration de la qualité d'images en se basant sur l'apprentissage profond est également présentée.

Les métriques utilisées pour juger les performances des techniques, seront présentés à la fin de ce chapitre.

SOMMAIRE

1.1 Introduction

Comme les images peuvent contenir une grande quantité de données et d'informations, la technologie de traitement d'images est applicable à de nombreux domaines, tels que le traitement médical, les transports, l'armée, l'espace et l'aérospatiale, et l'ingénierie des communications Cependant, lors de la collecte, du stockage et du traitement des images, un bruit d'image peut être introduit, ce qui interfère avec les informations fournies par l'image et réduit la netteté de l'image. En général, les bruits d'image sont diverses tels que le bruit sel et poivre, le bruit gamma et le bruit gaussien [1]. Pour résoudre le problème, des techniques de débruitage d'image et de reconstruction de super-résolution d'image (SR) [2] ont été étudiées pour restaurer des images de haute qualité et de haute résolution.

Il est nécessaire d'appliquer une technique de débruitage efficace pour compenser une telle corruption de données. Le débruitage d'images reste encore un défi pour les chercheurs parce que le bruit introduit la suppression des artefacts et des causes de flou de l'image.

L'objectif principal de l'amélioration de l'image est de traiter une image donnée de manière à ce que le résultat soit plus approprié que l'image originale pour une application spécifique.

L'amélioration n'augmente pas le contenu d'information inhérent aux données, mais elle augmente la plage dynamique des caractéristiques choisies afin qu'elles puissent être détectées facilement. La plus grande difficulté dans l'amélioration de l'image est de quantifier le critère d'amélioration et, par conséquent, un grand nombre de techniques d'amélioration de l'image sont empiriques et nécessitent des procédures interactives pour obtenir des résultats satisfaisants [3]

Dans ce chapitre, nous présenterons deux classes distinctes de techniques de débruitage, à savoir : les méthodes conventionnelles de débruitage (filtrage, à base d'ondlette) et les débruiteurs à base de depp learning.et les deux critères d'évaluation, les plus utilisés : le rapport maximum signal sur bruit (peak signal to noise ration PSNR) et l'indice de similarité (similarity structural index SSIM).

1.2 Modélisation

En pratique, les images peuvent être dégradées par plusieurs types et formes de bruit, Principales sources de bruit dans les images numériques qui se produisent lors de l'acquisition

(Numérisation) et transmission.

Les performances d'un capteur d'imagerie sont affectées par divers facteurs tels que les conditions environnementales lors de l'acquisition, et le capteur d'image eux-mêmes. Par exemple, lors de la capture d'une image avec un appareil photo,
La lumière et la température du capteur sont les principaux facteurs affectant la quantité de bruit image capturée

Les images peuvent également être corrompues pendant le transfert, principalement en raison d'Interférence dans le canal utilisé pour la transmission, par exemple des images La transmission via le réseau sans fil peuvent être affectée par l'éclairage ou autres perturbations atmosphériques. Le bruit est donc un effet indésirable dans l'image, Il doit donc être réduit ou éliminé sans provoquer de changements significatifs Détails et caractéristiques de l'image. Par conséquent, il est important de savoir La nature des images polluantes par le bruit. En fait, le type de bruit le plus connu est le bruit blanc gaussien

$$\boldsymbol{Y = X + n} \text{ ……..(1.1)}$$

Tel que :

Y: Image bruitée (observée)

X: Image reconstruite débruitée qu'on veut obtenir

n: Bruit additif gaussien

Que nous avons considérés le modèle de bruit additif gaussien

1.3 Types de bruit

1.3.1 Bruit gaussien blanc additif

Le bruit gaussien blanc additif est un modèle de bruit de base utilisé dans la théorie de l'information pour imiter l'effet de nombreux processus aléatoires qui se produisent dans la nature. Les modificateurs désignent des caractéristiques spécifiques : Additif parce qu'il est ajouté à tout bruit qui pourrait être intrinsèque au système d'information. Le blanc fait référence à l'idée qu'il a une puissance uniforme sur toute la bande de fréquences pour le système d'information. C'est une analogie avec la couleur blanche qui a des émissions uniformes à toutes les fréquences du spectre visible. Gaussien parce qu'il a une distribution normale dans le domaine temporel avec une valeur moyenne du domaine temporel de zéro. Le bruit à large bande provient de nombreuses sources naturelles, telles que les vibrations

thermiques des atomes dans les conducteurs, le bruit de tir, le rayonnement du corps noir de la terre et d'autres objets chauds, et de sources célestes telles que le Soleil. La modélisation du bruit par bruit blanc gaussien additif (additive white gaussian noise)L'AWGN est souvent utilisé comme modèle de canal dans lequel la seule altération de la communication est un ajout linéaire de large bande ou de bruit blanc avec une densité spectrale constante et une distribution gaussienne de l'amplitude.[4]

$$\boldsymbol{G(X,Y) = \frac{1}{2\pi\sigma^2}\exp\left(\frac{-X^2+Y^2}{2\sigma^2}\right)} \ \ldots\ldots\ldots \text{(1.2)}$$

Où $\boldsymbol{\sigma}$: représente l'écart type, X et Y : les coordonnées de l'origine de l'image

(a) Image original. **(b)** Image bruité, bruit blanc gaussien

Figure 1.1 Bruit additif Blanc gaussien

1.3.2 Bruit impulsionnel :

Est un bruit poivre et sel aléatoire que subit une image numérique, faisant passer l'intensité de certains pixels (répartis d'une manière aléatoire dans l'image) à la valeur minimum ou maximum de la plage dynamique du pixel, respectivement 0 et 255 dans le cas d'une image numérique codée en 8-bits]. Cela se traduit dans le cas d'une photo numérique par l'apparition de pixels noirs et blancs comme le montre la Figure 1.2 d'où l'appellation bruit poivre et sel du bruit. [5]

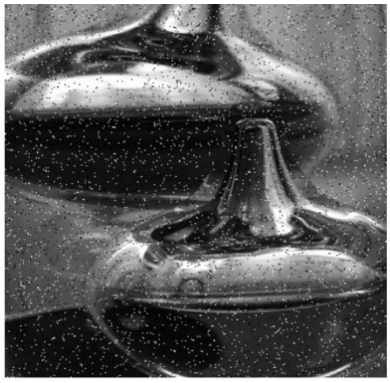

Figure 1.2. Image numérique contaminée par le bruit poivre et sel

1.3.3 Bruit poisson :

Le bruit de Poisson provient de la nature discrète des événements. Par exemple, l'ensemble de données pourrait être une image microscopique d'une intensité très limitée (comme un échantillon fluorescent). Dans ce cas, la probabilité que l'on mesure un nombre spécifique de photons est [6]:

$$p(K) = \frac{\lambda^{k} exp(-\lambda)}{k!} \quad \text{..... (1.3)}$$

ƛ Étant la valeur attendue et **k** celui mesuré

Notre bruit de Poisson sélectionne au hasard les valeurs de cette distribution

Dans notre travail, nous nous intéressons au bruit blanc gaussien additif

1.4 Méthode conventionnelle de débruitage

Il existe plusieurs méthodes conventionnelles de débruitage pour améliorer la qualité d'une image

1.4.1 Filtrage

1.4.1.1 Filtres linéaires

Sont des méthodes de traitement d'image qui permettent de réduire le bruit en remplaçant chaque pixel de l'image par une combinaison linéaire pondérée de ses voisins. Ces filtres sont souvent utilisés pour améliorer la qualité d'une image en réduisant le bruit de fond ou en lissant les contours. [7]

Dans le traitement d'images, différents types de filtres peuvent être utilisés pour réduire le bruit et améliorer la qualité de l'image. Le filtre moyenneur consiste à remplacer chaque pixel par la moyenne de ses voisins, ce qui peut réduire le bruit de fond mais entraîner une perte de détails et un flou d'image. En revanche, le filtre gaussien utilise une pondération gaussienne pour donner plus d'importance aux pixels les plus proches du pixel central, préservant ainsi les contours de l'image tout en réduisant le bruit. Enfin, le filtre médian non linéaire fonctionne en remplaçant chaque pixel par la valeur médiane de ses voisins, ce qui est particulièrement efficace pour supprimer le bruit impulsif qui peut se présenter sous forme de pixels isolés très différents de leurs voisins [8]

1.4.1.2Filtres non-linéaires : Cette méthode consiste à utiliser des filtres non-linéaires tels que le filtre de la médiane ou le filtre adaptatif pour supprimer le bruit dans une image. Ces filtres sont plus efficaces que les filtres linéaires pour supprimer le bruit impulsif ou le bruit de fond [9]

1.4.2 Méthodes de débruitage à base d'ondelettes (wavelets)

Consiste à appliquer une transformée en ondelettes aux images bruitées pour obtenir une représentation dans un espace où les coefficients de haute fréquence représentent les détails (bruits) et les coefficients de basse fréquence représentent les caractéristiques globales de l'image.

Ensuite, on seuille les coefficients de haute fréquence en dessous d'un certain seuil pour les annuler et ne garder que les coefficients de basse fréquence et les détails significatifs. Enfin, on applique une transformée inverse en ondelettes pour obtenir l'image débruitée.

Il existe plusieurs techniques de seuillage des coefficients de haute fréquence, telles que le seuillage dur (où on annule les coefficients en dessous d'un certain seuil) et le seuillage doux (où on atténue les coefficients en dessous d'un certain seuil).

Le choix de la méthode de seuillage et du seuil dépend du type de bruit présent dans le signal, ainsi que des caractéristiques du signal lui-même. Le débruitage par ondelettes est une méthode très efficace pour supprimer les bruits de haute fréquence tout en préservant les caractéristiques globales du signal [10]

1.5 Méthodes à base de Deep Learning (Denoisers)

Les techniques d'apprentissage profond de débruitage d'image sont des méthodes qui utilisent des réseaux de neurones profonds pour supprimer les perturbations ou le bruit des images numériques.

1.6.1 Auto-encodeurs :

Les auto-encodeurs sont des réseaux de neurones qui apprennent à encoder une image en une représentation latente, puis à la décoder à nouveau en une image débruitée.

En entraînant un auto-encodeur sur des images bruyantes et non bruyantes, le modèle peut apprendre à débruiter automatiquement des images.

1.6.2 Réseaux de neurones convolutifs :

Les réseaux de neurones convolutifs sont couramment utilisés pour le traitement d'images. Ils sont particulièrement efficaces pour la suppression de bruit en raison de leur capacité à capturer des motifs locaux. Les réseaux de neurones convolutifs apprennent à filtrer le bruit des images en appliquant des filtres adaptatifs sur l'image d'entrée [11]

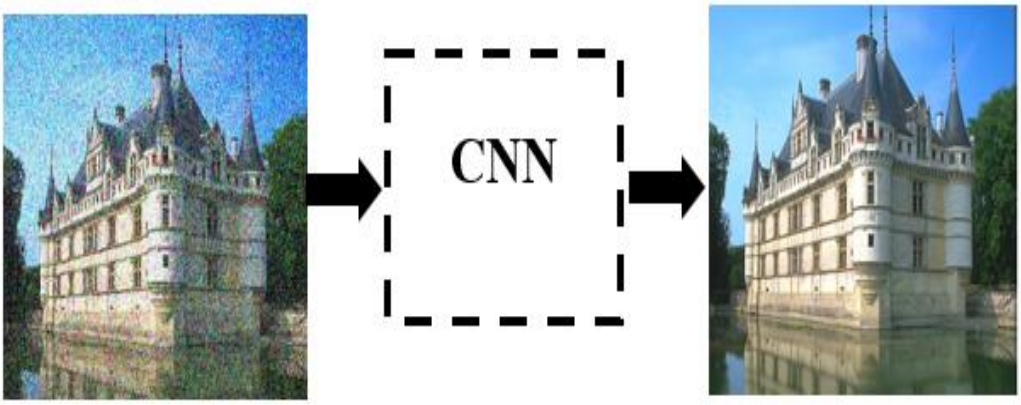

Image Bruité

Image débruitée

1.6 Critères d'évaluation

1.6.1 PSNR : (Peak Signal to Noise Ratio) est une mesure de distorsion utilisée en image numérique, tout particulièrement en débruitage d'image. Elle permet de quantifier la performance de la méthode de débruitage en mesurant la qualité de reconstruction de l'image débruitée par rapport à l'image originale

Le PSNR d'une image $\boldsymbol{I_0}$ $\boldsymbol{de\ taille}$ $[\boldsymbol{m \times n}]$, est défini par la formule suivante :

$$\textbf{PSNR}=\boldsymbol{10log_{10}(\frac{d^2}{EQM})} \quad \textbf{(1.4)}$$

Où $\boldsymbol{d}$est la dynamique de l'image (la valeur maximum possible pour un pixel), dans le cas standard d'une image codée sur 8-bits,

EQM L'erreur quadratique moyenne :

$$\boldsymbol{EQM = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(I_0(i,j) - I_r(i,j))^2} \quad \textbf{(1.5)}$$

$\boldsymbol{I_r}$ L'image reconstruite

$\boldsymbol{m \times n}$ La taille d'image

i, j Représentent le pixel ligne et colonne de l'image position

Si le PSNR est utile pour mesurer la proximité de l'image débruitée par rapport à l'original au niveau de l'image, il ne prend pas en compte la qualité visuelle de reconstruction et ne peut pas être considéré comme une mesure objective de la qualité visuelle d'une image. D'où la nécessité d'un autre critère, le SSIM.

1.6.2 SSIM : Structural SIMilarity Index ou SSIM est une mesure de similarité entre deux images numériques

La formule SSIM est basée sur trois mesures comparant des échantillons de x et y ;

Luminance (l)

Contraste (c)

Fonction de comparaison individuelle sont :

$$\boldsymbol{i(x,y) = \frac{\mu_x \mu_y + C_1}{\mu_x{}^2 + \mu_y{}^2 + C_1}} \text{..... (1.6)}$$

$$\boldsymbol{C(x,y) = \frac{2\sigma_x \sigma_y + C_2}{\sigma_x{}^2 \sigma_y{}^2 + C_2}} \text{..... (1.7)}$$

$$\boldsymbol{S(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x \sigma_y + C_3}} \text{...../..... (1.8)}$$

$\boldsymbol{\mu_x, \mu_y}$ Les moyennes des images originales et débruitée, $\boldsymbol{\sigma_x, \sigma_y}$ les écarts type et $\boldsymbol{C_1, C_2}$ les coefficients de corrélation.

Avec, en plus des définitions ci-dessus :

$$\boldsymbol{C_3 = C_2/2}$$

Le SSIM est alors une combinaison pondérée de ces mesures comparatives :

$$\boldsymbol{SSIM(x,y) = i(x,y)^{\alpha}.C(x,y)^{\beta}.s(x,y)^{\gamma}} \text{..... (1.9)}$$

Conclusion

L'obtention d'information à partir de mesures corrompues par un bruit reste un problème ouvert, que ce soit en traitement du signal ou en traitement d'image.

Nous avons exposé, dans ce chapitre, les méthodes classiques de débruitage ainsi que les méthodes basées sur l'apprentissage profond.

Celles-ci consistent à réduire le bruit dans les zones qui ne présentent pas d'objets intéressants et à accentuer la perception des structures d'intérêt.

L'inconvénient des méthodes classiques est l'atténuation dès les contours de l'image. Cette limitation est à la base de proposer d'autres procédures de prétraitement. Nous avons également détaillé les deux critères d'évaluation, à savoir le PSNR et le SSIM. Ces deux critères seront calculés dans notre étude de simulation.

Le Chapitre suivant, détaillera les concepts de base de l'apprentissage profond et quelques architectures des réseaux neuronaux.

Chapitre 2

Principes d'Apprentissage Profond

Résumé

Dans ce Chapitre, nous allons détaillé le principe de l'apprentissage automatique et l'apprentissage profond . L'a**rchitecture d'un réseau neuronale profond sera présentée ainsi que les types de réseaux de neurones. Nous métrerons l'accent sur le réseau de neurones Convolutifs (Convolutional neurol network CNN). Nous aborderons l'application des réseaux CNN en débruitage d'images, à travers le débruiteur CNN (denoiser CNN DnCNN network).**

SOMMAIRE

2.1 Introduction

L'apprentissage profond (deep learning DL) est un type d'apprentissage automatique (Machine Learning ML) et d'intelligence artificielle (IA) qui imite la façon dont les humains acquièrent certains types de connaissances. L'apprentissage profond est un élément important de la science des données, qui comprend les statistiques et la modélisation prédictive. Il est extrêmement bénéfique pour les scientifiques de données qui sont chargés de recueillir, d'analyser et d'interpréter de grandes quantités de données; l'apprentissage profond rend ce processus plus rapide et plus facile.

Le deep learning (DL) ou apprentissage profond, regroupe actuellement les méthodes les plus efficaces et les plus performantes appliquées dans la communauté de l'apprentissage automatique [1]. La Figure 1.1 illustre le lien entre, l'intelligence artificielle, l'apprentissage automatique et l'apprentissage profond.

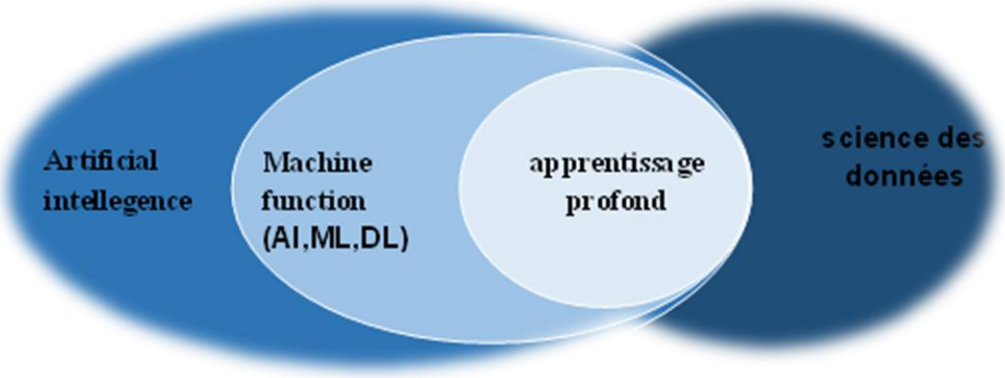

Figure 2.1 Lien entre AI, ML & DL

Dans ce qui suit, nous détaillerons d'abord l'apprentissage automatique (Machine learning).

L'architecture d'une réseau neuronal profond avec ses différentes couches, sera ensuite présentée. Nous nous intéressons essentiellement au réseau neuronal convolutif (convolutional neurol network CNN), qui sera appliquer pour le débruitage des images affectées par un bruit gaussien additif.

2.2 Apprentissage Machine [13]

L'apprentissage machine est une technique de programmation informatique qui utilise des probabilités statistiques pour donner aux ordinateurs la capacité d'apprendre par eux-mêmes sans programmation explicite. L'objectif de base de l'apprentissage machine est d'apprendre à apprendre aux ordinateurs et par la suite, à agir et réagir comme le font les humains, en améliorant leur mode d'apprentissage et leurs connaissances de façon autonome sur la durée. L'objectif ultime serait que les ordinateurs agissent et réagissent sans être explicitement programmés pour ces actions et réactions. L'apprentissage machine utilise des programmes de développement qui s'ajustent chaque fois qu'ils sont exposés à différents types de données en entrée.

Un bon exemple de l'apprentissage machine est la voiture autonome. Une voiture autonome est équipée de plusieurs caméras, plusieurs radars et d'un capteur lidar.

2.3 Architecture des réseaux neuronaux artificiels (Artificial Neuron Network ANN)

Le cerveau humain est la source d'inspiration de l'architecture des réseaux neuronaux. Les cellules du cerveau humain, appelées neurones, forment un réseau complexe et hautement interconnecté et s'envoient des signaux électriques les unes aux autres pour aider les humains à traiter les informations. De même, un réseau neuronal artificiel est constitué de neurones artificiels qui travaillent ensemble pour résoudre un problème. Les neurones artificiels sont des modules logiciels, appelés nœuds, et les réseaux neuronaux artificiels sont des programmes logiciels ou des algorithmes qui, à la base, utilisent des systèmes informatiques pour résoudre des calculs mathématiques [14].

Les réseaux neuronaux sont des structures complexes constituées de neurones artificiels qui peuvent prendre plusieurs entrées pour produire une seule sortie. Il s'agit de la tâche principale d'un réseau neuronal – transformer les intrants en extrants significatifs. Habituellement, un réseau neuronal se compose d'une couche d'entrée et de sortie avec une ou plusieurs couches cachées à l'intérieur. Il est également connu comme Artificial Neural Network ou ANN. Dans un réseau neuronal, tous les neurones s'influencent mutuellement, et par conséquent, ils sont tous connectés. Le réseau peut reconnaître et observer tous les aspects de l'ensemble de données à portée de la main et la façon dont les différentes parties des données peuvent ou non être liées les unes aux autres. C'est ainsi que les réseaux neuronaux sont capables de trouver des schémas extrêmement complexes dans de vastes volumes de

données. La Figure 2.1 illustre l'architecture d'un réseau de neurone avec une seule couche cachée.

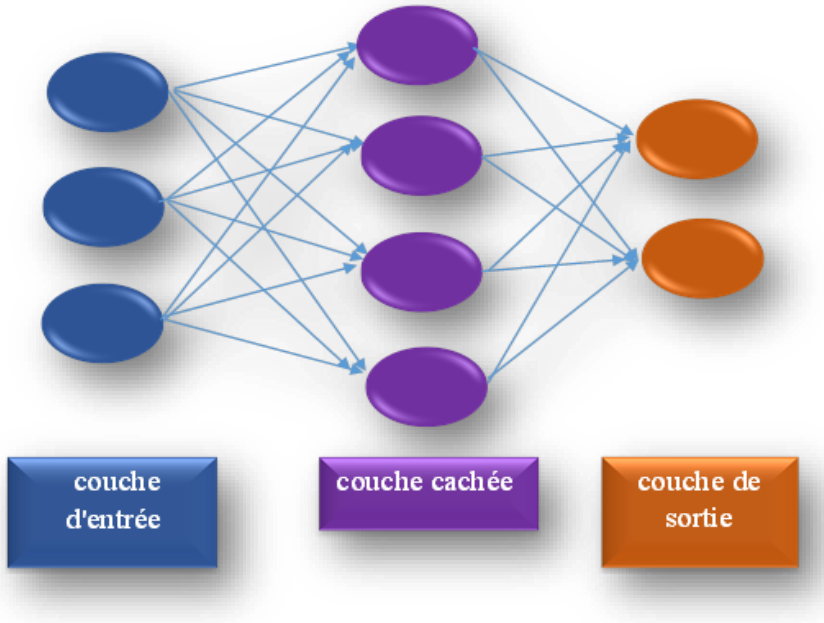

Figure 2.2 l'architecture des réseaux neuronaux

Dans un réseau neuronal, le flux d'informations se produit de deux façons :

Réseaux directs (Feedforward): Dans ce modèle, les signaux ne se déplacent que dans une direction, vers la couche de sortie. Les réseaux Feedforward ont une couche d'entrée et une couche de sortie unique avec zéro ou plusieurs couches cachées. Ils sont largement utilisés dans la reconnaissance de forme.

Réseaux de rétroaction : Dans ce modèle, les réseaux récurrents ou interactifs utilisent leur état interne (mémoire) pour traiter la séquence d'entrées. En eux, les signaux peuvent voyager dans les deux directions à travers les boucles (couches cachées) dans le réseau. Ils sont généralement utilisés dans les séries chronologiques et les tâches séquentielles.

L'ANN est constitué de plusieurs neurones. Le neurone artificiel est une modélisation simplifiée du neurone biologique. Son premier modèle date des années quarante et il a été présenté par Mc Culloch et Pitts [15]. Le neurone formel est une modélisation mathématique qui reprend les principes de fonctionnement du neurone biologique, en particulier la sommation des entrées. La Figure 2.2 représente le modèle mathématique d'un neurone artificiel.

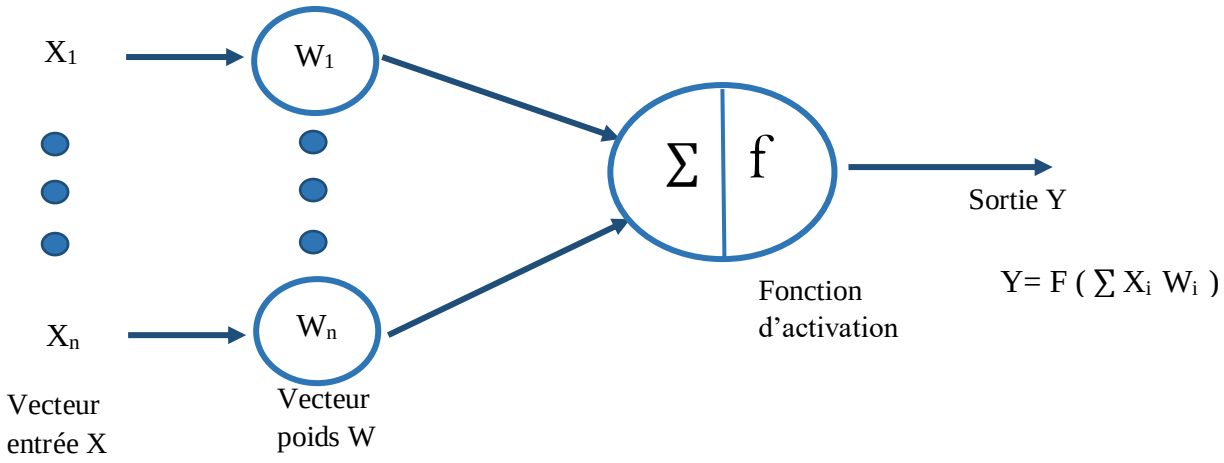

Figure 2. 3 Modélisation d'un neurone formel

2.4 Apprentissage Profond (Deep learning DL) [16]

Le deep learning ou apprentissage profond passe par le déploiement d'un réseau de neurones artificiel préalablement entraîné. Il s'agit d'une pratique d'intelligence artificielle (IA) issue de l'apprentissage automatique ou machine learning.

Des architectures d'apprentissage profond telles que les réseaux de neurones profonds, l'apprentissage par renforcement profond, les réseaux de neurones récurrents, les réseaux de neurones convolutifs et les transformateurs ont été appliqués pour résoudre différents problématiques tels que la reconnaissance vocale, traitement du langage naturel, traduction automatique, bioinformatique, conception de médicaments, analyse d'images médicales, climatologie, inspection du matériel et programmes de jeux de société, lorsqu'ils ont produit des résultats comparables et, dans certains cas, supérieurs à ceux d'experts humains.

Les réseaux neuronaux artificiels (ANN) ont été inspirés par le traitement de l'information et les nœuds de communication distribués dans les systèmes biologiques.

L'adjectif "profond" dans l'apprentissage profond se réfère à l'utilisation de plusieurs couches dans le réseau. Les premiers travaux [17] ont montré qu'un perceptron linéaire ne peut pas être un classificateur universel, mais qu'un réseau avec une fonction d'activation non polynomiale avec une couche cachée de largeur illimitée peut l'être. L'apprentissage profond est une variation moderne qui concerne un nombre illimité de couches de taille limitée, qui permet une application pratique et une mise en œuvre optimisée, tout en conservant l'universalité théorique dans des conditions douces. Dans l'apprentissage profond, les couches

sont également hétérogènes et à s'écartent largement des modèles connexes biologiquement informés, pour des raisons d'efficacité, de formation et de compréhension.

2.4.2 Propagation directe :

C'est une technique utilisée pour trouver la sortie réelle des réseaux neuronaux. Dans cette étape, l'entrée est alimentée au réseau dans une direction vers l'avant. Il nous aide à trouver la sortie réelle de chaque neurone. La propagation vers l'avant fait référence au stockage et au calcul des données d'entrée qui sont transmises vers l'avant par le réseau pour générer une sortie.[18]

2.4.3 Fonction de perte (Loss function) :

Une fonction de perte, ou Loss function, est une fonction qui évalue l'écart entre les prédictions réalisées par le réseau de neurones et les valeurs réelles des observations utilisées pendant l'apprentissage. Plus le résultat de cette fonction est minimisé, plus le réseau de neurones est performant. Sa minimisation, c'est-à-dire réduire au minimum l'écart entre la valeur prédite et la valeur réelle pour une observation donnée, se fait en ajustant les différents poids du réseau de neurones.[19]

2.4.4 Former le réseau neuronal :

Tous les neurones d'une couche donnée génèrent une sortie, mais ils n'ont pas le même poids pour la prochaine couche de neurones. Cela signifie que si un neurone sur une couche observe un modèle donné, cela pourrait signifier moins pour l'image globale et sera partiellement ou complètement coupé. C'est ce que nous appelons la pondération : un gros poids signifie que l'entrée est importante et bien sûr un petit poids signifie que nous devrions l'ignorer. Chaque connexion neurerale entre les neurones aura un poids associé.

Les poids seront ajustés au cours de l'entraînement pour correspondre aux objectifs que nous avons fixés (exemple reconnaître qu'un chien est un chien et qu'un chat est un chat). En termes simples : Former un réseau neuronal signifie trouver les poids appropriés des connexions neuronales grâce à une boucle de rétroaction gradient appelée Gradient Backward propagation.[20]

2.5 Réseaux de neurones Convolutifs CNN : Principe d'architecture CNN

Un réseau de neurones convolutifs (convolution neuronaux network (CNN)) est un type de réseau neuronal artificiel utilisé dans la reconnaissance et le traitement d'images et spécifiquement conçu pour traiter les données de pixels (donc un traitement ponctuels).Les convolutions neuronaux network sont de puissants systèmes de traitement d'images, d'intelligence artificielle (IA) qui utilisent un apprentissage approfondi pour effectuer des tâches à la fois génératives et descriptives–qui inclut l'amélioration d'images (enhancement) et débruitage (la problématique que nous traitons dans notre étude), la reconnaissance d'images et de vidéos, ainsi que des systèmes de recommandation et le traitement du langage naturel[21].

2.5.1 Architecture

Les réseaux de neurones convolués sont à ce jour les modèles les plus performants pour classer des images. Désignés par l'acronyme CNN, de l'anglais Convolutional Neural Network, ils comportent deux parties bien distinctes. En entrée, une image est fournie sous la forme d'une matrice de pixels. Elle a deux dimensions pour une image aux niveaux de gris. La couleur est représentée par une troisième dimension, de profondeur 3 pour représenter les couleurs fondamentales [Rouge, Vert, Bleu]. La première partie d'un CNN est la partie convolutive à proprement parler. Elle fonctionne comme un extracteur de caractéristiques des images. Une image est passée à travers–une succession de filtres, ou noyaux de convolution (kernel), créant de nouvelles images appelées cartes de convolutions (feature maps). Certains filtres intermédiaires réduisent la résolution de l'image par une opération de maximum local. Enfin, les cartes de convolutions sont mises à plat et ce code CNN en sortie de la partie convolutive et ensuite branché en entrée d'une deuxième partie, constituée de couches entièrement connectées. Le rôle de cette partie est de combiner les caractéristiques du code CNN pour classer l'image. La sortie est une dernière couche comportant un neurone par catégorie. Les valeurs numériques obtenues sont généralement normalisées entre 0 et 1, de somme 1 pour produire une distribution de probabilité sur les catégories..

Il existe quatre types de couches pour un réseau de neurones convolutif : la couche de convolution, la couche de pooling, la couche de correction ReLU et la couche fully-connected.[22]

Dans ce qui suit, le détail de chaque couche.

Couche de Convolution :

La couche de convolution est la composante clé des réseaux de neurones convolutifs, et constitue toujours au moins leur première couche. Son but est de repérer la présence d'un ensemble de features dans les images reçues en entrée. Pour cela, on réalise un filtrage par convolution : le principe est de faire "glisser" une fenêtre représentant la feature sur l'image, et de calculer le produit de convolution entre la feature et chaque portion de l'image balayée. Une feature est alors vue comme un filtre : les deux termes sont équivalents dans ce contexte.[23]

***Couche de Pooling* :**

Ce type de couche est souvent placé entre deux couches de convolution : elle reçoit en entrée plusieurs cartes de caractéristiques (feature maps), et applique à chacune d'entre elles l'opération de pooling. L'opération de pooling consiste à réduire la taille des images, tout en préservant leurs caractéristiques importantes. La couche de pooling permet de réduire le nombre de paramètres et de calculs dans le réseau. On améliore ainsi l'efficacité du réseau et on évite le sur-apprentissage.[24]

***Couche de Correction Relu* :**

ReLU (Rectified Linear Units) désigne la fonction réelle d'activation non-linéaire qui est considéré comme seuillage. Elle est illsutrée sur la Figure 2.3

$$ReLU(x) = max(0, x)$$

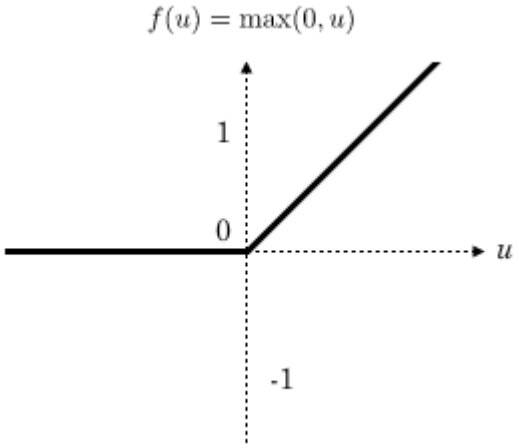

Figure 2.4 Allure de la fonction ReLU

La couche de correction ReLU remplace donc toutes les valeurs négatives reçues en entrées par des zéros. Elle joue le rôle de fonction d'activation.[25]

Couche de Entièrement Connecté (Full Connected Layer) :

Ce type de couche reçoit un vecteur en entrée et produit un nouveau vecteur en sortie. Pour cela, elle applique une combinaison linéaire puis éventuellement une fonction d'activation aux valeurs reçues en entrée. La dernière permet de classifier l'image en entrée du réseau : elle renvoie un vecteur de taille N, où N est le nombre de classes dans notre problème de classification d'images. Chaque élément du vecteur indique la probabilité pour l'image en entrée d'appartenir à une classe. La Figure 2.4 montre les différents blocs du CNN.[26]

2.6 Bases de Données dans l'Apprentissage Profond (Datasets) :[27]

Les ensembles de base de données (Datasets) dans le deep learning sont considérés big data. Ceux sont des bibliothèques permettant d'accéder et de partager facilement des ensembles de données pour les tâches Audio, Computer Vision et Natural Language Processing (NLP). Les bases de données servent en l'apprentissage du réseau grâce au grand nombre d'images test.

2.7 Application des réseaux CNN en débruitage d'images : Réseaux DnCNN

Les réseaux de neurones convolutifs ont montré un grand succès dans le traitement de diverses tâches de vision de bas niveau. Le débruitage des images en est un des problèmes traités par les CNN.

Le but du débruitage d'image est de récupérer l'image propre x de l'image bruitée $y = x + n$, L'hypothèse est que n est le bruit gaussien blanc additif (additive white gaussian noise AWGN). En général, les méthodes de débruitage de l'image peuvent être regroupées en deux grandes catégories - les méthodes basées sur des modèles et l'apprentissage discriminatoire. Les méthodes basées sur des modèles comme BM3D et WNNM sont flexibles dans la gestion des problèmes de débruitage avec différents niveaux de bruit, mais leur exécution prend du temps, et elles nécessitent la modélisation d'antécédents complexes. Pour surmonter ces inconvénients, des méthodes de débruitage basées sur le DL ont été développées [31].

Dans notre travail, nous nous sommes basés sur le débruiteur proposé dans [ref], pour traiter des images affectées par un bruit gaussien additif. Le réseau de neurones convolutionnels de débruitage proposé s'appui essentiellement sur le réseau CNN, d'où son nom débruiteur CNN (denoiser CNN : DnCNN). Plutôt que de dépasser directement l'image propre x, le modèle est formé pour prédire l'image résiduelle $x' = R(y)$, c'est-à-dire la

différence entre l'observation bruitée y et l'image propre latente x. La technique de normalisation améliore et stabilise davantage les performances de formation du DnCNN.

2.8 Architecture du réseau DnCNN :

La Figure 2.6 illustre l'architecture du réseau DnCNN, l'entrée étant une image bruitée.

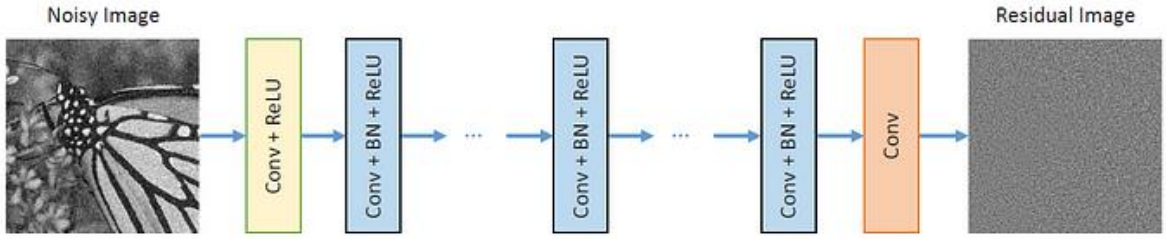

Figure 2.5 Dénoiser DnCNN

L'image d'entrée représente les observations bruitées. L'image va passer par des filtres de la couche de convolution (Conv), suivie par l'application de la fonction d'activation RelU(rectfied linear Unit). La taille des filtres convolutionnels est fixée à $[3 \times 3]$ et toutes les couches de regroupement sont supprimées. Par conséquent, le champ réceptif de DnCNN avec une profondeur de d devrait être $(2d + 1) \times (2d + 1)$.

La formulation de l'apprentissage résiduel est adoptée pour former une cartographie résiduelle, en d'autres termes, l'image propre est éstimée par la différence des observations et du résiduel estimé :

$$x = y - R(y) \quad \text{.....(1.1)}$$

Ainsi, $R(y)$ est appris.

Pour Récapituler, il y a 3 types de couches :

-(i) Conv+ReLU: Pour le premier calcul, 64 filtres de convolution (Conv) de taille $[3 \times 3]$ sont utilisés pour générer 64 cartes d'images (feature maps). Un paramètre c représente le nombre de canaux de couleurs, i.e $c = 1$ pour l'image niveaux de gris et $c = 3$ pour l'image couleur RVB.

-(ii) Conv+BN+ReLU: pour les couches 2 à $(D - 1)$, 64 filtres de convolution de taille $[3 \times 3]$, 64 sont utilisés, et la normalisation (BiNarisation) est ajoutée entre la convolution (Conv) et ReLU.

-(iii) Conv : pour la dernière couche, ces filtres de taille $[3 \times 3]$ 64 sont utilisés pour reconstruire la sortie.

La stratégie de remplissage zéro simple (zero padding) est utilisée avant la convolution pour n'entraîner aucun artefact de bord.

En incorporant la convolution avec ReLU, DnCNN peut progressivement séparer la structure de l'image de l'observation bruitée à travers les couches cachées.

DnCNN est formé de bout en bout (end-to-end).[32]

2.9 Conclusion :

Dans ce chapitre, nous avons présenté les notions importantes liées à l'apprentissage profond. Nous avons présenté une des applications de l'apprentissage profond qui est le déruitage des images. Nous avons mis l'accent sur le denoiser DnCNN qui se base essentiellement sur la structure de CNN. Ce denoiser va être implémenté dans notre travail de simulation. Le chapitre suivant sera entièrement dédié à la présentation de nos résultats de simulation ainsi que les étapes d'implémentation du denoiser DnCNN.[33]

Chapitre 3

Implémentation du Débruiteur à Base du réseau à Convolution (Denoiser Convolutional Neurol Network: DnCNN)

Résumé

Dans ce chapitre, nous détaillerons notre implémentation du denoiser DnCNN : d'abord sous Pytorch, puis nous allons construire le réseau DnCNN sous Keras, en définissant tous les paramètres du réseau. Les codes sources sont exécutés sous colab, language python. Les bases de données utilisées dans notre étude de simulation, sont celles utilisées dans la littérature de débruitage, pour établir une comparaison objective. Deux critères seront calculés pour évaluer les résultats du débruitage : le PSNR et le SSIM.

SOMMAIRE

3.1 Introduction :

Dans ce chapitre, nous allons implémenter le denoiser DnCNN sous pytorch et la construction du réseau DnCNN sous keras, en utilisant deux ensembles de données (considérées images originales références) **Set12** et **MNIST**. Le langage de programmation utilisé est Python en exploitant surtout les bibliothèques : Numpy, Torch, Tensorflow, Keras et Matplotlib. Les données bruitées sont générées avant d'appliquer le denoiser DnCNN. Nous considérons que le bruit est additif gaussien avec différentes sigma qui représente le niveau du bruit dans l'image. Pour établir une comparaison quantitative, nous calculerons les deux critères les plus utilisés en débruitage : le rapport signal max sur bruit (Peak Signal to Noise Ratio PSNR) et l'indice de similarité de structure (Similarity structure index SSIM), en plus de la qualité d'image débruitée.

3.2 Matériels Utilisés :

Notre implémentation se base sur les deux frameworks :

(a) Pytorch

Qui se base la programmation Python. Ces principales caractéristiques : tenseurs dynamiques, visualisation facile des graphiques de calcul et des opérations effectuées par le modèle et accélération GPU, ce qui permet d'exécuter des calculs intensifs sur les cartes graphiques, accélérant ainsi le processus d'apprentissage des modèles.

(b) Keras

Nous utilisons Keras pour la construction du réseau débruiteur DnCNN, en déterminant tous les paramètres du réseau. Nous nous sommes basés sur des couches préconstruites du réseau débruiteur. Keras est également basé sur Python et fonctionne sur plusieurs frameworks de calcul sous-jacents, dont TensorFlow. Notre choix de Keras est pour tirer profit de sa Modularité, c'est-à-dire que Keras permet de construire des modèles en les assemblant à partir de blocs de construction appelés couches (layers). Il propose une large gamme de couches préconstruites, ainsi que la possibilité de créer ses propres couches personnalisées. Keras est souvent utilisé avec des bibliothèques telles que TensorFlow pour des tâches plus larges d'apprentissage en profondeur (DL).

3.3 Jeu de données ÷

Dans nos expériences de simulation, nous utilisons les bases suivantes :

- **MNIST** :

Dans notre travail, cette base est utlisée pour tester DnCNN sous Keras. Nous générons également les images buitées. Rappelons que la base de données MNIST (Changed National Organization of Benchmarks and Innovation database) est une énorme base de données de chiffres écrits manuellement qui est normalement utilisée pour préparer différents systèmes de traitement d'images. La base de données est en outre généralement utilisée pour préparer et tester dans le domaine de l'apprentissage automatique. La base de données du MNIST contient 60 000 images de préparation et 10 000 images d'essai. La Figure 3.1 illustre quelques images de cette base.

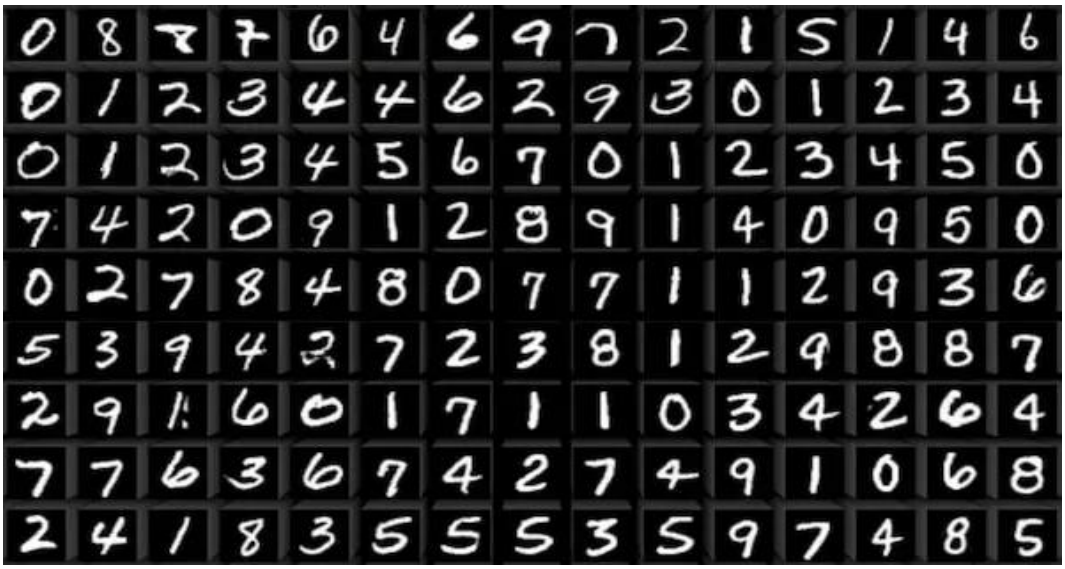

Figure 3.1 Images de la base MNIST

- **Set 12**

C'est une collection de 12 images en niveaux de gris de différentes scènes qui sont largement utilisées pour l'évaluation des méthodes de débruitage d'image.

Nous avons choisi cette base pour comparer nos résultats avec ceux de la littérature. Le tableau 3.1 résume les différentes images de l'ensemble de données Set 12.

Tableau 3.1 Images de la base de données Set 12

Nom d'image	Base d'image	format	Taille	Type
01	**Set12**	**PNG**	**256*256**	**NB**
02	**Set12**	**PNG**	**256*256**	**NB**
03	**Set12**	**PNG**	**256*256**	**NB**
04	**Set12**	**PNG**	**256*256**	**NB**
05	**Set12**	**PNG**	**256*256**	**NB**
06	**Set12**	**PNG**	**256*256**	**NB**
07	**Set12**	**PNG**	**256*256**	**NB**
08	**Set12**	**PNG**	**512*512**	**NB**
09	**Set12**	**PNG**	**512*512**	**NB**
10	**Set12**	**PNG**	**512*512**	**NB**
11	**Set12**	**PNG**	**512*512**	**NB**
12	**Set12**	**PNG**	**512*512**	**NB**

Figure 3.2 Base de données Set 12

3.4 Etapes d'implémentation du débruiteur DnCNN :

L'organigramme de la Figure 3.3 résume les bibliothèques que nous avons utilisées dans nos simulations. Nous implémentons DnCNN sous Pytorch, puis nous construisons DnCNN séparément sous Keras. L'objectif est de vérifier les résultats de simulation par deux frameworks.

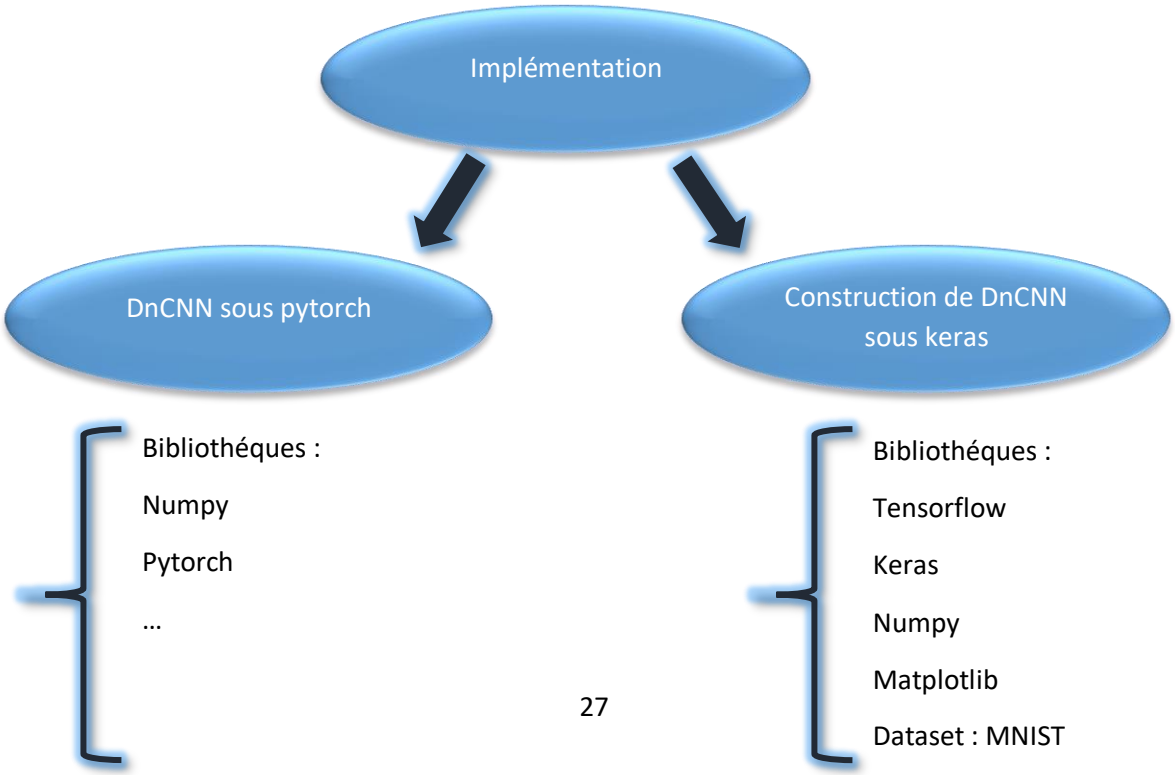

Figure 3.3 Différentes bibliothèques utilisées

- ***DnCNN par Pytorch***

La Figure 3.4 résume l'organigramme de nos simulations du DnCNN, avec l'exemple de l'image buttrerfly ainsi que les valeurs PSNR et SSIM obtenus. Rappelons que plus PSNR et grand et SSIM proche de 1, meilleur est le débruitage. Les images des données sont les images références sans bruit. Nous générons d'abord les versions bruitées des images originales, en additionnant le bruit gaussien. Nous avons fait varier σ sur l'intervalle$[25 - 50]$.

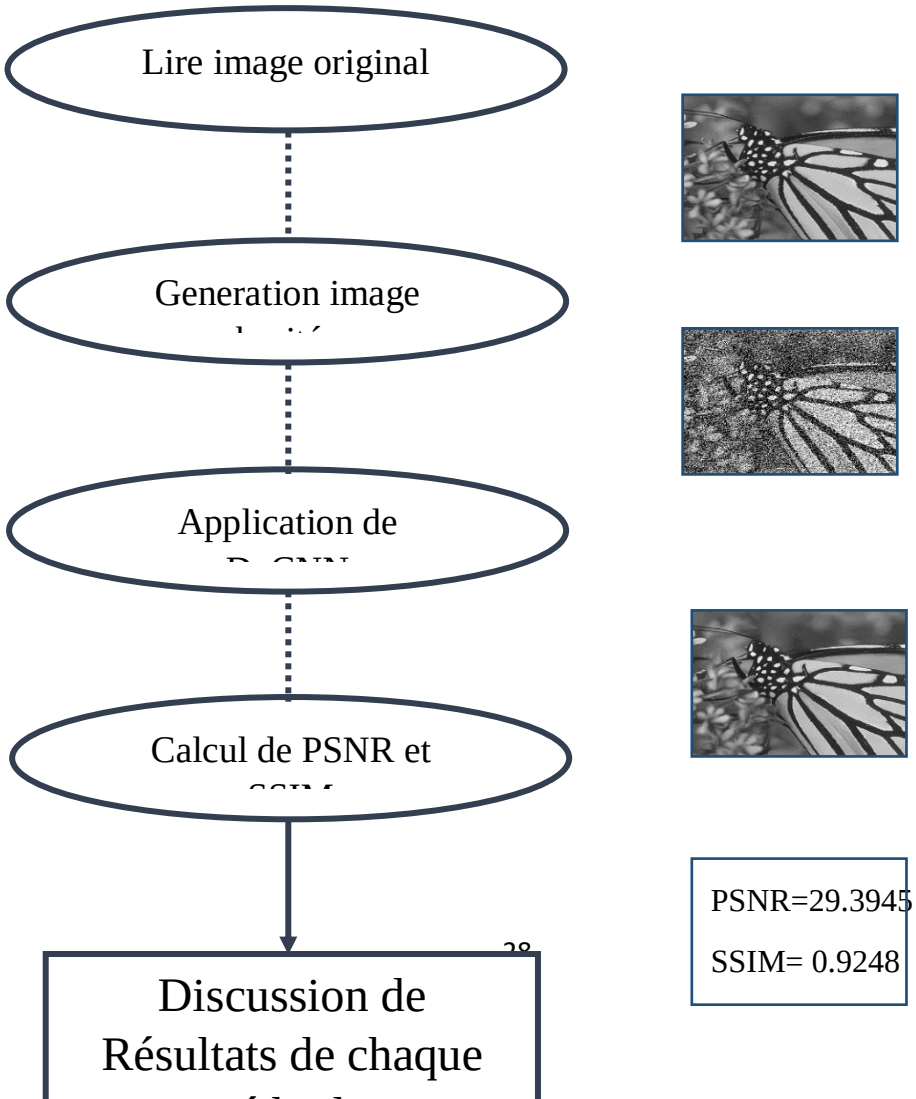

Figure 3.4 Organigramme des étapes d'implémentation de DnCNN

DnCNN sous Keras (l'autoencoder DnCNN)

Les données que nous avons utilisées dans nos études de simulation dans cette partie ont été extraites des bases de données MNIST et Set12.

L'ensemble de données MNIST contient 70 000 exemples d'entraînement de taille 28 × 28 pixels. Puisqu'il est important de bien gérer l'ensemble de données, nous avons divisé une partie des données en un pour la formation (60 000 exemples) et un autre pour les tests (10 000 exemples),

La Figure 3.5 montre l'architecture de l'Auto-encodeur utilisé : encodeur-décodeur. L'architecture se base sur le principe de CNN, *i.e* une succession de couches de convolution 2D avec des filtres de taille $[3x3]$ et de modules nonlinéaires ReLU et des couches maxpooling pour réduire la taille des images. La sortie du réseau est l'image résultante débruitée.

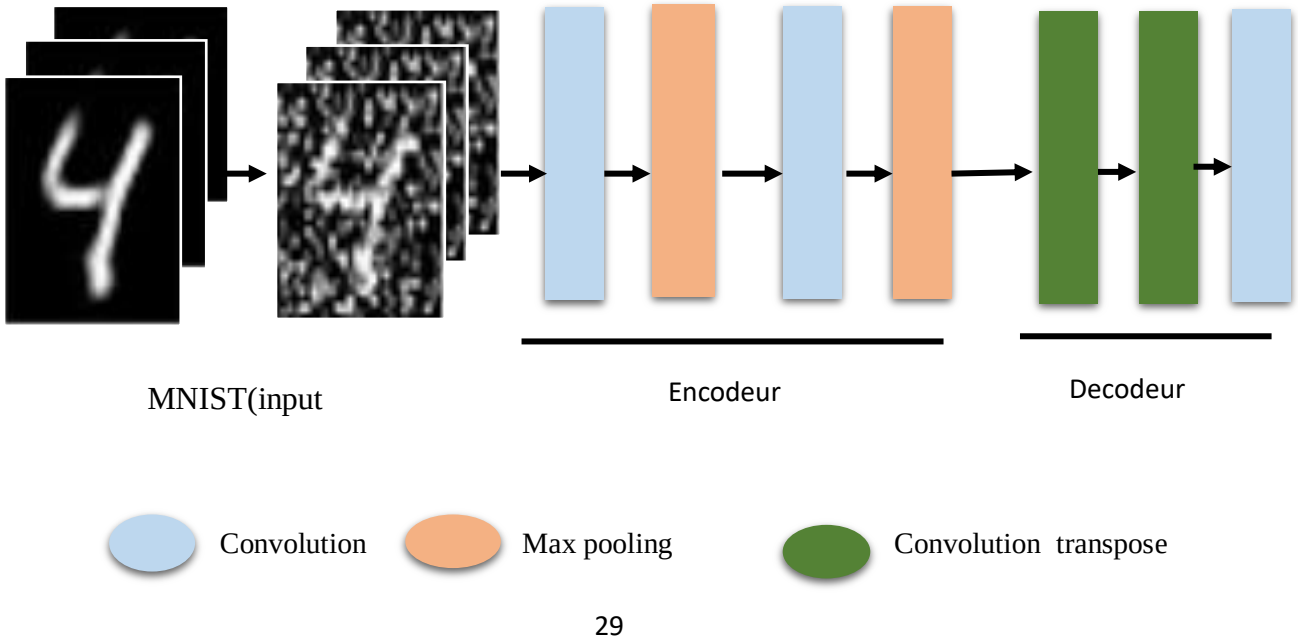

Figure 3.5 Architecture adoptée pour l'implémentation de l'auto-encodeur DnCNN.

Nos données (c'est-à-dire les unités d'entrée) passent par une couche convolutive, puis une couche de sous-échantillonnage avec une mise en commun maximale (maxpooling), puis un deuxième passage à travers la couche convolutive, puis une deuxième couche de mise en commun pour un deuxième sous-échantillonnage. Cela constitue la partie codage. Pour le décodage, il est composé de deux couches de convolution transposée, c'est-à-dire qu'on a fait une opération de suréchantillonnage (Upsampling) pour augmenter la résolution de l'image après la sortie de la dernière couche de convolution. Nous avons conçu cet algorithme à partir de Keras Tensorflow, qui dispose d'une d'interface de programmation d'application (Application Programming Interface API). Cette interface simplifie et facilite la conception de nos réseaux de neurones, qui pourraient être très complexes sans cette interface de programmation. Notre réseau se compose des éléments suivants :

- Des données d'entrée avec le bruit gaussien ajouté. Les images de taille 28 × 28 × 1

puisque les images sont niveaux de gris alors le canal vaut 1. La puissance du bruit est fixée à travers σ, avec différentes valeurs ($\sigma \in [25 - 50]$).

- La sortie de la première couche devient l'entrée de la couche suivante convolutive de

L'encodeur, donc nous avons déterminé 32 filtres de noyau $\mathbf{3} \times \mathbf{3}$, en ajoutant un padding = 1 pour avoir en sortie de la couche des caractéristiques (feature maps) de même taille, la sortie de cette couche après convolution donne 32 maps de taille 28×28 ce qui donne 28 × 28 × 32.

- Au passage vers la couche de Pooling un sous-échantillonnage de noyau de taille 2 ×2 est effectué avec un Stride = 2 pour avoir en sortie (selon l'équation 1.12) une taille réduite qui vaut 14 × 14 × 32.
- Un deuxième filtrage par la couche de convolution qui suit donne à sa sortie la même taille de 14 × 14 × 32.
- Après un second sous-échantillonnage, de la même façon que le 1er, la taille devient encore plus réduite en sortie de cette couche 7 × 7 × 32, par la suite elle devient l'entrée du décodeur.
- À l'entrée du décodeur, il y'a la couche de convolution transposée pour un suréchantillonnage de taille 7 × 7 × 32 vers une taille plus grande 14 × 14 × 32 pour stride = 2.

- La deuxième couche convolutive transposée fait la même chose, c'est-à-dire de 14 ×14 × 32 vers 28 × 28 × 32 à sa sortie.
- Le filtrage dans la dernière couche permet de restituer une image de même taille que celle de l'entrée. Nous avons donc résumé cette description par le schéma illustré sur la Figure 3.6.

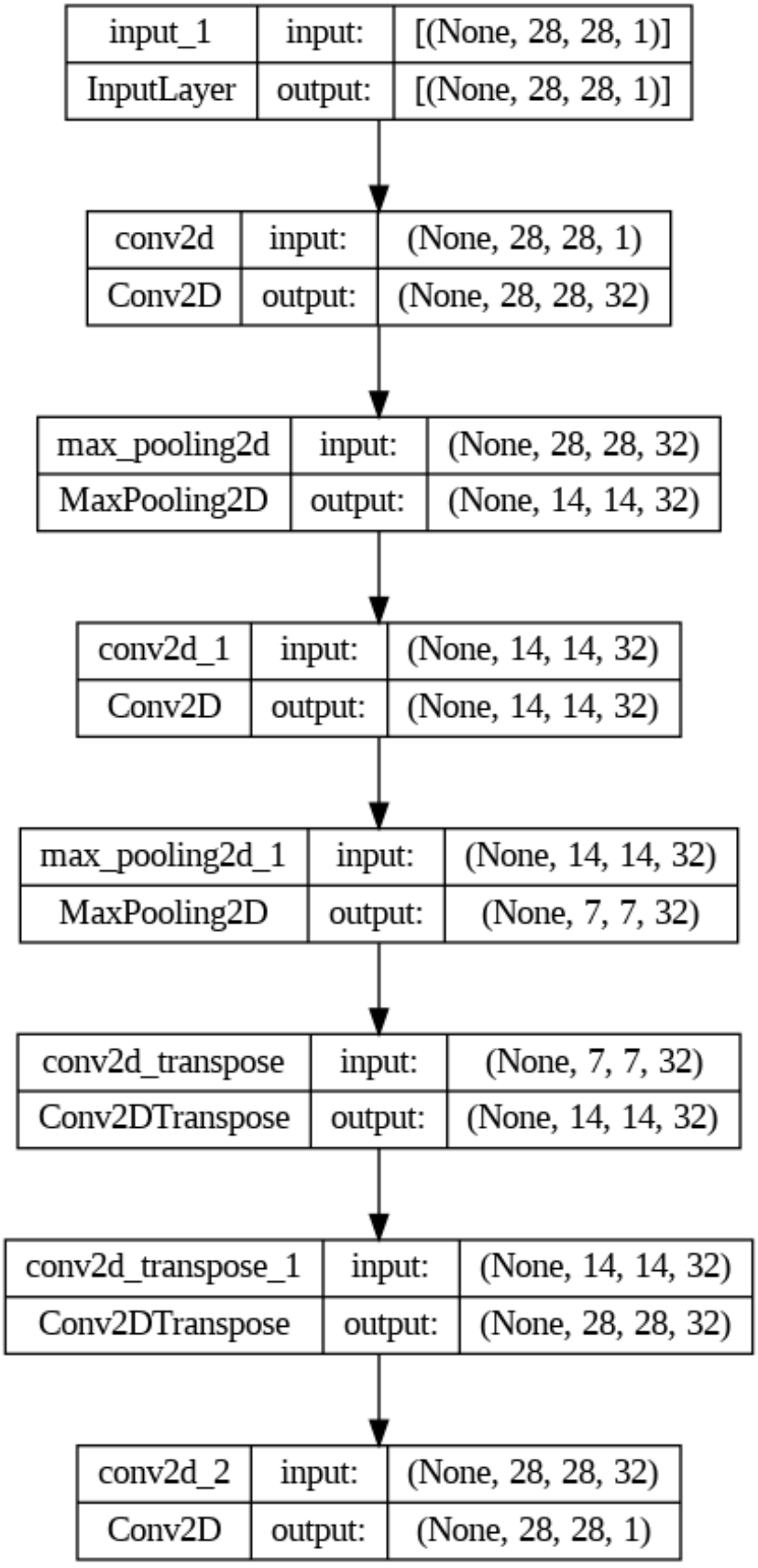

Figure 3.6 Structure Auto-encodeur du denoiser DnCNN

3.5 Résultats et Discussions

Dans les deux implémentations du denoiser DnCNN, nous générons d'abord les images bruitées de la base Set12. La Figure 3.8 illustre les images bruitées générée $\sigma = 25$. Vu la limitation des pages, nous avons pris deux images de l'ensemble Set1é, House et butterfly.

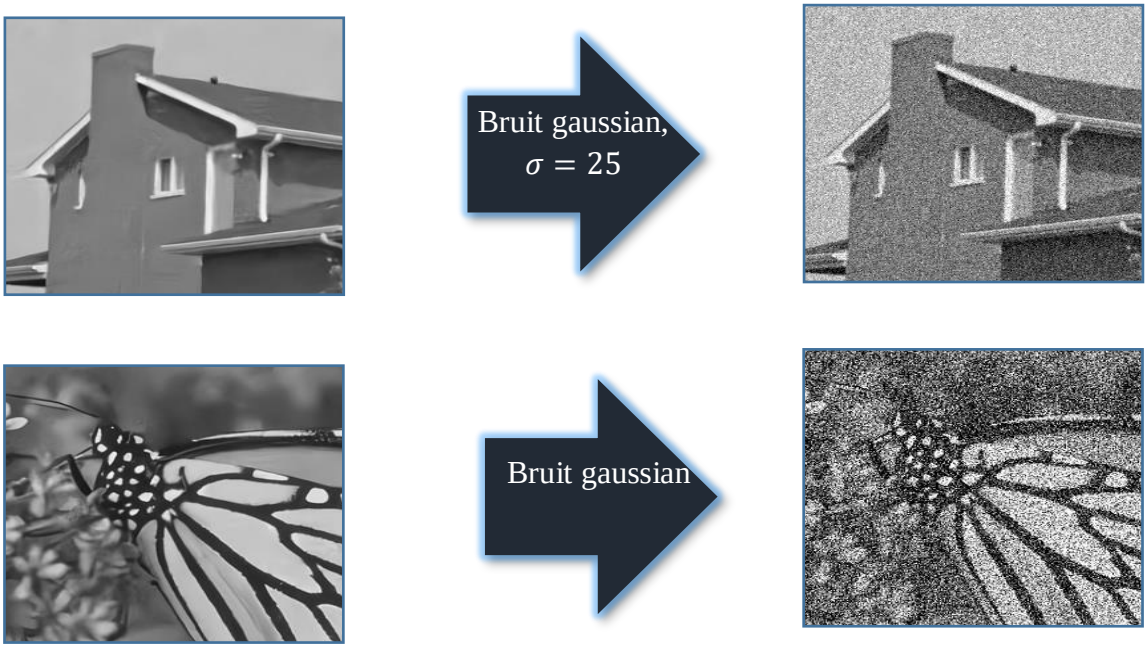

Figure 3.7 Génération des images bruitées

Résultats Obtenus par DnCNN sous Pytorch

La Figure 3.9 illustre les résultats du débruitage de toutes les images de Set12, pour différentes valeurs de . Il est clair que l'amélioration des images obtenues est considérable, le bruit est très réduit. Le Tableaux 3.2 des valeurs de PSNR et SSIM traduisent en concordance avec cette bonne qualité. Nous avons fait varié le niveau du bruit dans les images en augmentant σ, tel que $\sigma \in [25 - 50]$. Nous avons calculé PSNR (dB) et SSIM pour chaque σ et pour les douze images de Set12 ainsi que les valeurs moyennes. Les valeurs PSNR et SSIM dans les Tableaux 3.2-3.7 montrent la diminution de PSNR et SSIM, ce qui est très logique.

Tableau 3.2 PSNR et SSIM de DnCNN sous Pytorch

σ	Nom d'image	PSNR (dB)	SSIM
σ =25	01	30.0910	0.9134
	02	30.0857	0.9072
	03	30.0578	0.9343
	04	30.2646	0.9317
	05	29.3945	0.9248
	06	30.4557	0.9558
	07	32.4228	0.9421
	08	29.0877	0.9318
	09	30.2178	0.9081
	10	29.4448	0.9271
	11	30.8104	0.9398
	12	33.1382	0.9439

Figure 3.8 Images débruitées par DnCNN sous Pytorch (σ=25)

Tableau 3.3: Expérience de débruitage des images pour σ = 30

σ	**Nom d'image**	**PSNR (dB)**	**SSIM**
σ =30	01	26.6499	**0.8008**
	02	26.6601	**0.7891**
	03	26.6699	**0.8265**
	04	27.0635	**0.8139**
	05	26.3864	**0.8469**
	06	27.1027	**0.8637**
	07	27.6103	**0.7960**
	08	26.2505	**0.8181**
	09	26.7158	**0.7898**
	10	26.6765	**0.8253**
	11	27.1114	**0.8241**
	12	28.0363	**0.8047**

Tableau 3.4 Expérience de débruitage des images pour σ = 35

σ	Nom d'image	PSNR (dB)	SSIM
σ =35	01	22.5191	0.6034
	02	22.5440	0.5805
	03	22.6288	0.6363
	04	22.9559	0.6013
	05	22.5207	0.6836
	06	22.8035	0.6866
	07	22.8621	0.5589
	08	22.5595	0.6218
	09	22.5865	0.5883
	10	22.8157	0.6284
	11	22.7647	0.6213
	12	23.0714	0.5680

Tableau 3.5 Expérience de débruitage des images pour σ = 40

σ	Nom d'image	PSNR (dB)	SSIM
σ =40	01	19.9068	0.4688
	02	19.9472	0.4436
	03	19.9981	0.5049
	04	20.3162	0.4701
	05	19.9698	0.5585
	06	20.0777	0.5627
	07	20.1262	0.4156
	08	20.0661	0.4948
	09	19.9699	0.4543
	10	20.2451	0.5007
	11	20.0863	0.4877
	12	20.2402	0.4244

Tableau 3.6 Expérience de débruitage des images pour σ = 45

SIGMA	Nom d'image	PSNR	SSIM
σ =45	01	18.1315	0.3825
	02	18.1750	0.3578
	03	18.2034	0.4183
	04	18.4898	0.3896
	05	18.2392	0.4740
	06	18.2459	0.4817
	07	18.2868	0.3289
	08	18.3498	0.4153
	09	18.1847	0.3696
	10	18.4716	0.4219
	11	18.2710	0.4030
	12	18.3539	0.3385

Tableau3.7 Expérience de débruitage des images pour σ = 50

σ	Nom d'image	PSNR (dB)	SSIM
σ =50	01	16.7814	0.3220
	02	16.8245	0.2986
	03	16.8555	0.3568
	04	17.0900	0.3342
	05	16.9346	0.4125
	06	16.8696	0.4238
	07	16.9018	0.2711
	08	17.0813	0.3611
	09	16.8250	0.3109
	10	17.1155	0.3671
	11	16.8980	0.3440
	12	16.9526	0.2819

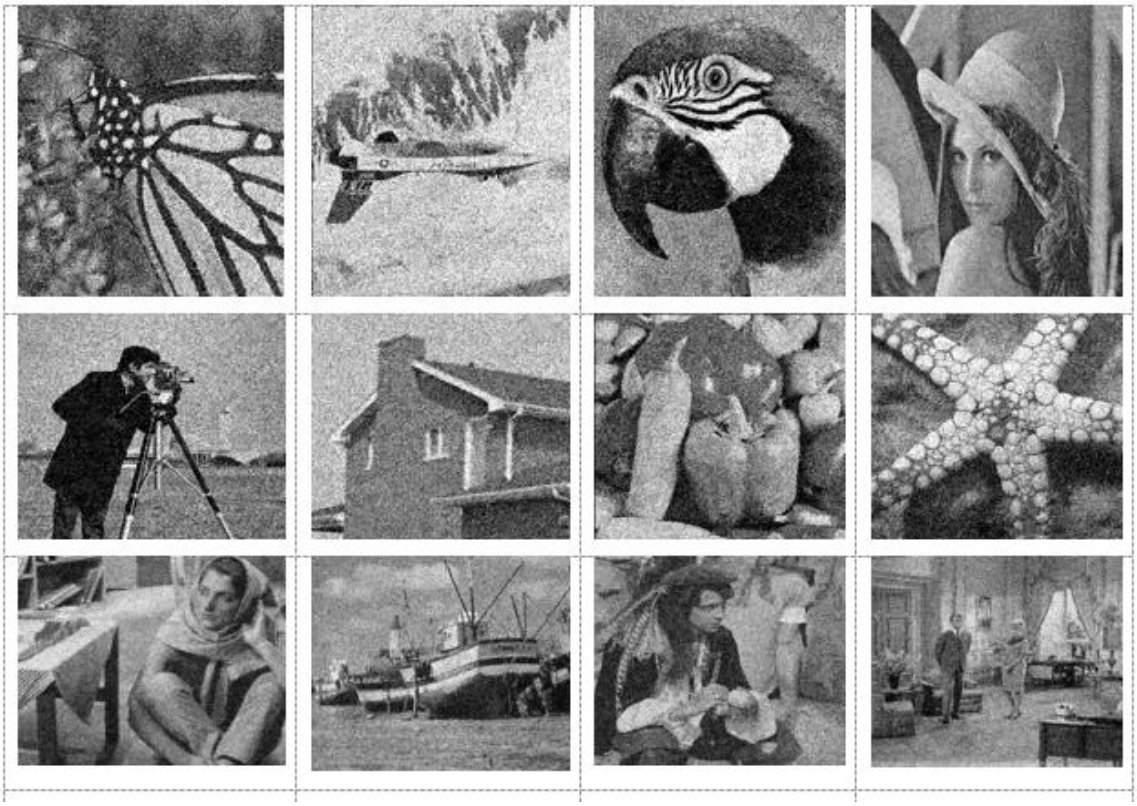

Figure 3.9 :images débruitées par DnCNN sous Pytorch (σ=50)

Tableau 3.8 : Resumée de la moyenne PSNR et SSIM, de DnCNN sous Pytorch Set12

σ	25	30	35	40	45	50
PSNR	30 .4559	26.91	22.72	20.08	18.2836	16.9275
SSIM	0.9300	0.8166	0.6149	0.4822	0.3984	0.3403

Il est clair que la qualité de l'image débruitée est satisfaisante même pour un niveau élevé du bruit.

3.7 Résultats de l'encodeur DnCNN Keras

Par analogie à la première partie, nous générons les images bruitées avec différentes σ, et nous calculons les deux critères PSNR et SSIM pour juger la qualité de la reconstruction. Nous avons calculé PSNR et SSIM pour chaque image et pour chaque σ. Dans un souci d'organisation, nous avons calculé la moyenne de PSNR et SSIM pour l'image ?????., comme c'est montré sur le Tableau 3.9. Les images débruitées obtenues, pour les valeurs de σ=25 et σ=50 sont illustrées sur la Figure 3.10. Les résultats montrent l'efficacité de DnCNN en

débruitage. La valeur de SSIM traduit le compromis entre qualité meilleur et préservations des informations dans l'image traitée.

SIGMA	Nom d'image	PSNR	SSIM
σ =25	01	33.0919	0.9434
	02	30.0935	0.9347
	03	30.2464	0.9316
	04	30.2183	0.9078
	05	30.0846	0.9068
	06	29.0884	0.9317
	07	29.3521	0.9243
	08	32.3997	0.9416
	09	30.4249	0.9557
	10	30.8483	0.9400
	11	30.1028	0.9135
	12	29.4497	0.9266

Figure 3.10 :images débruitées par DnCNN sous keras(σ=25)

SIGMA	Nom d'image	PSNR	SSIM
σ =30	01	28.0690	0.8063
	02	26.7284	0.8291
	03	27.1043	0.8161
	04	26.7855	0.7930
	05	26.7392	0.7927
	06	26.2563	0.8190
	07	26.4248	0.8489
	08	27.7192	0.8007
	09	27.1443	0.8662
	10	27.2002	0.8277
	11	26.7221	0.8043
	12	26.7125	0.8283

SIGMA	Nom d'image	PSNR	SSIM
σ =35	01	23.1340	0.5726
	02	22.7034	0.6406
	03	23.0158	0.6050
	04	22.6421	0.5924
	05	22.6259	0.5858
	06	22.5942	0.6253
	07	22.5812	0.6870
	08	22.9535	0.5649
	09	22.8526	0.6901
	10	22.8392	0.6264
	11	22.5938	0.6083
	12	22.8691	0.6333

SIGMA	Nom d'image	PSNR	SSIM
σ =40	01	20.2908	0.4283
	02	20.0813	0.5097
	03	20.3391	0.4715
	04	20.0233	0.4584
	05	20.0288	0.4490
	06	20.0841	0.4967
	07	20.0389	0.5628
	08	20.1940	0.4201
	09	20.1309	0.5662
	10	20.1416	0.4917
	11	19.9845	0.4740
	12	20.2866	0.5045

SIGMA	Nom d'image	PSNR	SSIM
σ =45	01	18.3601	0.3397
	02	18.2417	0.4209
	03	18.4653	0.3889
	04	18.1940	0.3714
	05	18.2068	0.3604
	06	18.3447	0.4158
	07	18.2767	0.4767
	08	18.2983	0.3302
	09	18.2702	0.4837
	10	18.2820	0.4045
	11	18.1633	0.3854
	12	18.4693	0.4230

SIGMA	Nom d'image	PSNR	SSIM
σ =50	01	16.9158	0.2813
	02	16.8358	0.3568
	03	17.0373	0.3329
	04	16.7856	0.3107
	05	16.7991	0.2988
	06	17.0560	0.3608
	07	16.9278	0.4132
	08	16.8582	0.2702
	09	16.8598	0.4242
	10	16.8681	0.3436
	11	16.7603	0.3227
	12	17.0713	0.3664

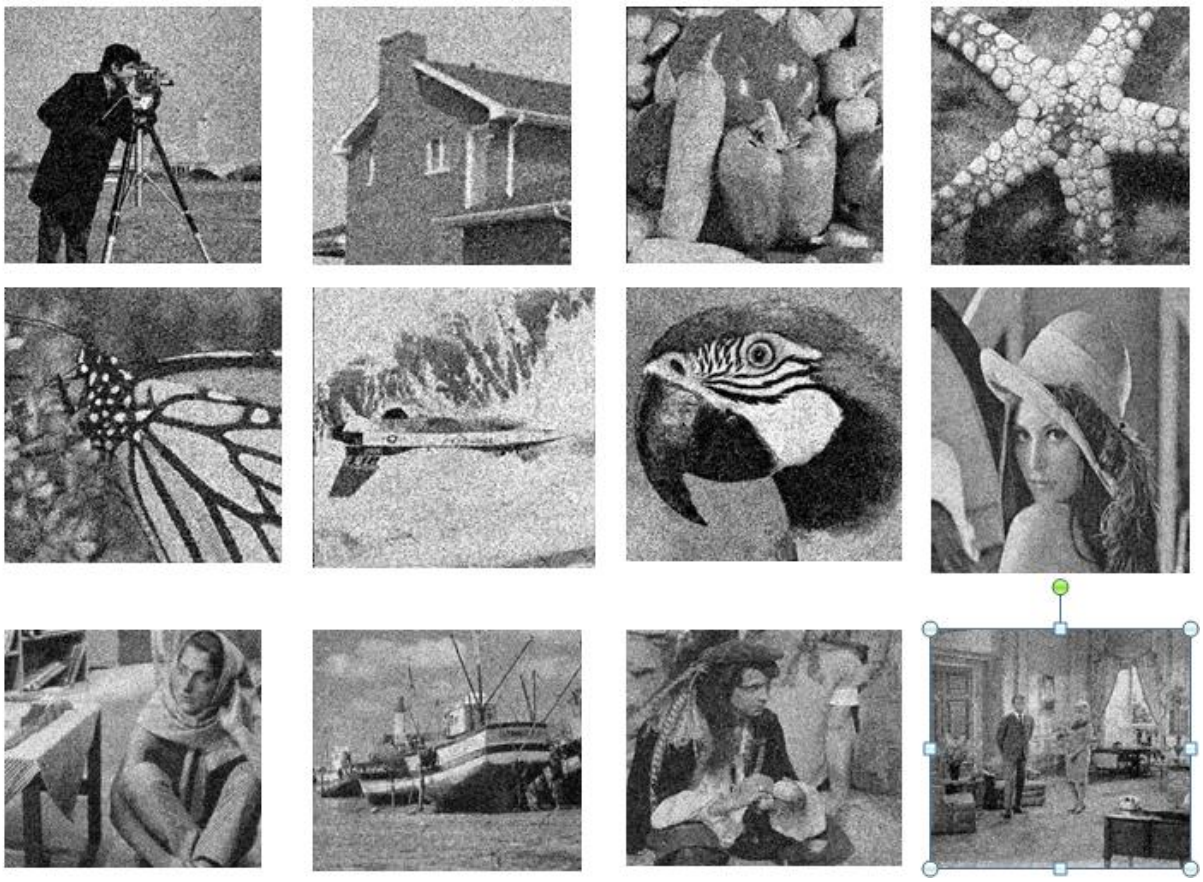

Figure 3.11 :images débruitées par DnCNN sous keras(σ=50)

Les valeurs de PSNR et SSIM sont résumés dans le Tableau 3.

Tableau 3.9 PSNR et SSIM de DnCNN Keras

σ	25	30	35	40	45	50
PSNR	30.4501	26.9672	22.7837	20.1353	18.2977	16.8979
SSIM	0.9298	0.8194	0.6193	0.4861	0.4000	0.3401

Il est clair que les résultats de simulation du DnCNN sou Pytorch et l'encodeur DnCNN sont très closes et permettent une amélioration considérable de la qualité d'image.

Pour mieux illustrer l'efficacité de l'encodeur DnCNN construit sous Keras, nous avons appliqué l'ensemble de données MNIST (très utilisé en deep learning).

Les données d'apprentissage sont montrés sur la Figure 3.12.

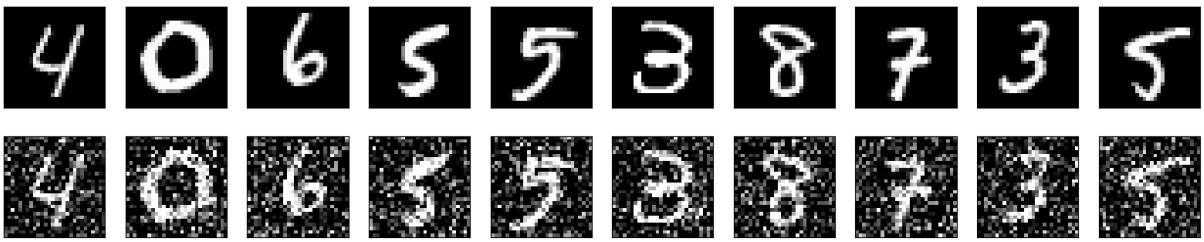

Figure 3.12: Images de Dataset MNIST(première ligne images originales, deuxième ligne images bruitées Les images débruitées sont montrées sur la Figure 3.12.

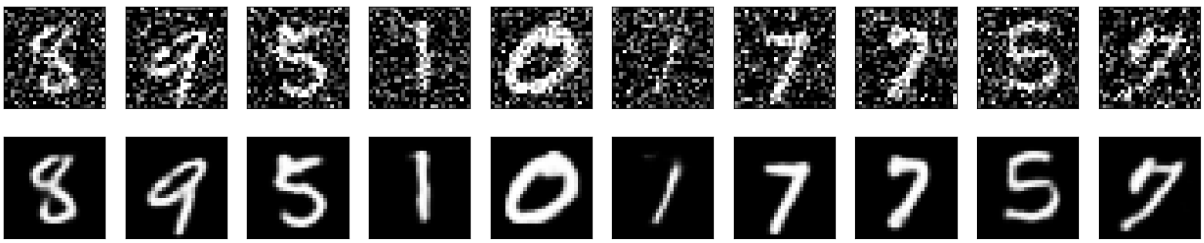

Figure 3.13. Images débruitées par l'encodeur DnCNN,

Pour mieux voir l'évolution de la convergence de l'encodeur DnCNN, nous avons tracé, dans la Figure3.13. la courbe de l'erreur quadratique moyenne (MSE) de l'entrainement et de la validation en fonction du nombre d'epoch. Il est clair que l'MSE diminue en augmentant le nombre d'epoch, puis elle se stabilise à 0.007 pour un nombre d'epoch égale à 100.

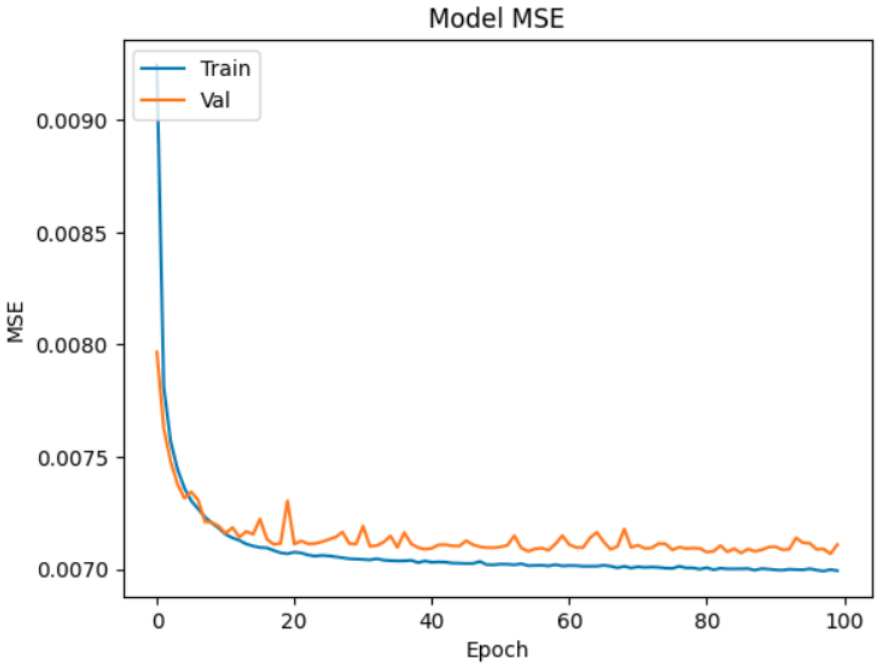

Figure 3.14. MSE vs le nombre d'epoch de l'encodeur DnCNN

Le résultat de prédiction des images bruitées en termes de qualité visuelle, est obtenu selon la correspondance entre chaque image :

3.6 Conclusion

Dans chapitre, nous avons détaillé les étapes de simulation du denoiser DnCNN sous Pytorch . Nous avons également expliqué les étapes de construction du DnCNN sous Keras et ses différentes couches (convolution, maxpooling, ReLU,..). Nous avons rassemblé touts les résultats obtenus au cous de ce travail. Nous avons utlisé principalement deux jeux de données : Set12 et MNIST. La première étape, consistait à générer les données bruitées. Le bruit considéré étant gaussien avec différentes σ. Nous avons calculé les deux critères PSNR et SSIM pour toutes les images traitées, puis nous avons calculé les valeurs moyennes.

Nous avons également tracé l'MSE en fonction du nombre d'epochs, pour montrer la convergence de l'encodeur DnCNN. Les résultats du débruitage de DnCNN sous Pytorch et celui construit sous Keras sont très approximatives et satisdaisantes. Les valeurs de PSNR et SSIM sont en concordance avec la qualité d'image s débruitées.

Pour conclure, DnCNN a permis une amélioration considérable des images bruitées, il dépasse les méthodes de filtrage classiques que nous avons effectué aux travaux pratiques.

Conclusion Générale

Conclusion Générale

En guise de conclusion générale, nous allons tenter d'établir une synthèse globale sur le travail qui a été réalisé dans ce mémoire.

Dans ce document, nous avons étudié le problème du débruitage qui a engendré une importante littérature en prétraitement des images. Nous nous sommes focalisés sur le débruitage à base d'apprentissage profond comme solution au problème.

Dans le premier Chapitre de ce document, nous avons exposé le formalisme général du problème de débruitage. Ainsi, Nous avons exposé les méthodes classiques de filtrage. Nous avons fait un tour d'horizon sur ces méthodes et leur intérêt en traitement d'images. Nous avons mis l'accent sur l'algorithme de débruitage dans le cas des images affectées par un bruit additif gaussien à différentes puissances.

Dans le deuxième Chapitre, Nous avons introduit les concepts de base de l'apprentissage profond (deep learning DL) ainsi que les premières architectures des réseaux DL. Nous avons mis l'accent sur le réseau de neurone convolutif CNN et ses différentes couches. .Nous avons détaillé le débruiteur d'image qui est à base de CNN (DnCNN) à la fin du chapitre 2.

Dans le troisième chapitre, nous avons exposé toutes les expériences de simulation que nous avons mené au cours de cette étude. Nous avons d'abord généré des données bruitées, le bruit étant additif gaussien de différentes puissances. Nous avons ensuite appliqué le DnCNN pour traiter ces images. La première configuration est réalisée sous Pytorch. les images des donnés sont set12 et set5. La 2éme configuration est la construction du réseau débruiteur sous Keras. Dans cette partie, la base MNIST est également utilisé. Nous avons calculé ensuite les critères PSNR et SSIM pour chaque image traitée, puis nous avons calculé leurs valeurs moyenne. La courbe MSE vs le nombre d'epochs est présentée pour le réeau DnCNN construit.

La conclusion générale, résume les principales remarques déduites de cette étude. D'après les résultats obtenus, il y a une amélioration considérable de la qualité des images reconstruites par DnCNN. Les valeurs de PSNR et SSIM confirment cette remarque.

Bibliographie

Bibliographie :

[1] C. Dong, C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in Proceedings of the European Conference on Computer Vision, pp. 184–199, Zurich, Switzerland, September 2014. View at: Google Scholar

[2] X. Qian, L. Peng, and X. Pang, “Image denoising auto-encoders based on residual entropy maximum,” IET Image Processing, vol. 14, no. 6, 2020.

View at: Google Scholar
W. Gong, Z. Shi, and H. Wei, “High resolution remote sensing satellite image uneven cloud removal algorithm,” Advances in Laser and Optoelectronics, vol. 57, no. 6, pp. 294–301, 2020
[3] https://blog.ifaci.com/ameliorer-limage-de-laudit-interne-pourquoi/

[4] https://www.definitions.net/definition/Additive+White+Gaussian+Noise

[5] Rafael C. González et Richard E. Woods, *Digital Image Processing (2nd Edition)*, New Jersey, Prentice Hall, 15 janvier 2002, 793 p.(ISBN 0-201-18075-8), p. 226

[6] : https://docs.juliahub.com/Noise/qByhT/0.3.0/man/poisson/

[7] Digital Image Processing" de Rafael C. Gonzalez et Richard E. Woods (2008)

[8] Image Processing, Analysis, and Machine Vision" de Milan Sonka, Vaclav Hlavac et Roger Boyle (2014)

[9] Image Processing, Analysis, and Machine Vision" de Milan Sonka, Vaclav Hlavac et Roger Boyle (2014)

[10] http://bib.univ-oeb.dz:8080/jspui/bitstream/123456789/8078/1/MII-00050.pdf

[11] Image Denoising Using Deep Learning: A Survey" de Jinshan Pan, et al. (2018)

[12]

[13] https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neural-network

https://culturesciencesphysique.ens-lyon.fr/pdf/IA-apprentissage-Rousseau.pdf

[14] machine learning : https://www.talend.com/fr/resources/what-is-machine-learning/

[15] Le deep learning ou apprentissage https://www.journaldunet.fr/web-tech/guide-de-l-intelligence-artificielle/1501333-deep-learning-definition-et-principes-de-l-apprentissage-profond/

https://en.wikipedia.org/wiki/Deep_learning

[16] https://www.upgrad.com/blog/neural-network-architecture-components-algorithms/

[17] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9686179/

[18] https://www.educative.io/answers/what-is-forward-propagation-in-neural-networks

[18] https://www.projectpro.io/recipes/what-is-forward-propagation

[19] https://www.editions-eni.fr/open/mediabook.aspx?idR=f6e7a7353a3574180124387fa03fdc1c

[20] https://blog.ovhcloud.com/what-does-training-neural-networks-mean/#:~:text=In%20simple%20terms%3A%20Training%20a,propagation%20%E2%80%A6%20and%20that's%20it%20folks

[21] https://developer.nvidia.com/discover/convolutional-neural-network#:~:text=Components%20of%20a%20Convolutional%20Neural,%2C%20height%2C%20and%20depth%20dimensions)

[22] https://actualiteinformatique.fr/intelligence-artificielle/definition-convolutional-neural-network

[23] https://adityarastogi2k12.github.io/Projects/NNLS/

[23] https://openclassrooms.com/fr/courses/4470531-classez-et-segmentez-des-donnees-visuelles/5083336-decouvrez-les-differentes-couches-dun-cnn

[24] https://paperswithcode.com/dataset/set12

[25] https://opendatalab.com/LIVE1

[26] https://huggingface.co/datasets/eugenesiow/Set5

[27] https://sh-tsang.medium.com/review-dncnn-residual-learning-of-deep-cnn-image-denoising-super-resolution-jpeg-deblocking-cbf464b03130

[28] https://medium.com/analytics-vidhya/what-is-unet-157314c87634

[29] https://www.researchgate.net/figure/A-comparison-between-the-results-of-GI-DnCNN-U-Net-and-DRU-Net_fig3_343134918

Résumé :

Aujourd'hui, l'apprentissage automatique s'est avéré être un concurrent sérieux pour la recherche technologique moderne. Dans ce manuscrit, nous proposons d'exploiter un certain domaine de l'apprentissage automatique, à savoir l'apprentissage en profondeur. Avec le Deep Learning, nous étudierons le débruitage d'images au moyen de réseaux de neurones profonds, plus spécifiquement les CNN (Convolutional Neural Networks), selon plusieurs algorithmes récents « ex :DnCNN paythorch »Nous les comparerons avec d'autres algorithmes de débruitage image tels que , « dncnn kerass »Avec cette étude comparative, nous utiliserons les critères d'évaluation en calculant PSNR, SSIM ainsi que le temps d'exécution et la qualité visuelle des images obtenues.

Mots-clés : apprentissage profond, débruitage, réseau de neurones profonds, Dncnn ,CNN, PSNR, SSIM

ملخص

اليوم أثبت التعلم الآلي أنه منافس جاد للبحث التكنولوجي الحديث. في هذه المخطوطة ، نقترح استغلال مجال معين من التعلم الآلي ، وهو التعلم العميق. مع التعلم العميق ، سوف ندرس تقليل ضوضاء الصور عن طريق الشبكات العصبية العميقة ، وبشكل أكثر تحديدا) CNN الشبكات العصبية التلافيفية) ، وفقا للعديد من الخوارزميات الحديثة nCNN D: ex" "paythorchسنقارنها بخوارزميات تقليل ضوضاء الصور الأخرى مثل"dncnn kerass" مع هذه الدراسة المقارنة ، سنستخدم معايير التقييم عن طريق حساب PSNR و SSIM بالإضافة إلى وقت التنفيذ والجودة المرئية للصور التي تم الحصول عليها.

الكلمات المفتاحية :التعلم العميق ، تقليل الضوضاء ، الشبكة العصبية العميقة ، Dncnn ، CNN ، SSIM،PSNR

Abstract

Today, machine learning has proven to be a strong contender for modern technological research. In this manuscript, we propose to exploit a certain area of machine learning, namely deep learning. With Deep Learning, we will study the denoising of images by means of deep neural networks, more specifically CNNs (Convolutional Neural Networks), according to several recent algorithms "ex: DnCNN paythorch" We will compare them with others image denoising algorithms such as , "dncnn kerass" With this comparative study, we will use the evaluation criteria by calculating PSNR, SSIM as well as the execution time and visual quality of the obtained images.