République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Mémoire de Master

Présenté à l'Université de Tébessa
Faculté des Sciences Exactes et Sciences de la Nature et de la Vie

Département de : **Mathématiques et Informatique**
Spécialité : **Informatique**
Option : **Systèmes et Multimédias**

Par :
DJEDDI Hana

Un système de compression fractale d'images basé sur les réseaux de neurones profonds

JURY

Encadreur	M.C.A	**MENASSEL Rafik**	Université de Tébessa
Président	M.C.A	**LAIMECHE Lakhdar**	Université de Tébessa
Examinateur	M.A.A	**AOUIN Mohammed**	Université de Tébessa

2019/2020

بسم هللا الرحمن الرحيم

À MES CHERS PARENTS

Aucune dédicace ne saurait exprimer mon respect, mon amour éternel et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être. Je vous remercie pour tout le soutien et l'amour que vous me portez depuis mon enfance et j'espère que votre bénédiction m'accompagne toujours. Que ce modeste travail soit l'exaucement de vos vœux tant formulés, le fruit de vos innombrables sacrifices, bien que je ne vous en acquitterai jamais assez.

*Mon père **LAAYACHI** demandé à dieu de bénir et nous rassemble au paradis.*

*Ma mère **HALIMA**. Que Dieu, le Très Haut, lui accorder santé, bonheur et longue vie et faire en sorte que jamais je ne tu déçois.*

À MES CHERS ET ADORABLES FRERES ET SŒURS QUE J'AIME PROFONDEMENT

SOUMAYA, TAREK, IMENE, YAZID, ISSAM, NESRINE.

À ma grand-mère

À MES AMIS DE TOUJOURS

En souvenir de notre sincère et profonde amitié et des moments agréables que nous avons passés ensemble. Veuillez trouver dans ce travail l'expression de mon respect le plus profond et mon affection la plus sincère.

DJEDDI HANA

Remerciements

***C**'est avec un grand plaisir que je réserve ces quelques lignes en signe de gratitude et de profonde reconnaissance à tous ceux qui de près ou de loin, ont contribué à la réalisation et l'aboutissement de ce travail*

***T**out d'abord, je tiens à remercier **Allah** tout puissant, de me permettre de mener à bien ce mémoire, et de m'orienter au chemin du savoir.*

***E**nsuite, je remercie sincèrement Mr. **MENASSEL Rafik**, Maitre de conférences au sein du département des mathématiques et informatique de la faculté des sciences exactes et sciences de la nature et de la vie à l'université de Tébessa, pour son encadrement, son assistance, son soutien, sa disponibilité et ses précieux conseils.*

*Mes vifs remerciements s'adressent à Mr. **LAIMECHE Lakhdar, et** Mr. **AOUIN Mohammed**, pour l'honneur qu'ils m'ont accordé en acceptants de juger mon travail.*

***E**nfin, j'adresse mes chaleureux remerciements à mes enseignants pour la qualité de l'enseignement qu'ils ont bien voulu me prodiguer durant mes études afin de me fournir une formation efficients.*

***M**erci à toutes et tous.*

Résumé

La compression d'images a pour but de réduire la taille d'une image afin de faciliter son stockage aussi bien que son transfert. Ainsi on distingue deux grandes familles de méthodes compression, à savoir celles qui provoquent des pertes d'information causant une image reconstruite non fidèle à l'originale mais de taille très réduite. Les autres méthodes ne provoquent pas de perte d'information mais présentent des taux de compression réduits.

L'optimisation des taux et les distorsions a longtemps été considérée comme un problème insoluble pour les images, ce qui a poussé les chercheurs à avancer largement dans ce sens en utilisant différentes techniques.

D'autre part, les réseaux de neurones profonds utilisent différentes couches d'unité de traitement non linéaire pour l'extraction et la transformation des caractéristiques ; chaque couche prend en entrée la sortie de la précédente ; les algorithmes peuvent être supervisés ou non supervisés, et leurs applications comprennent la reconnaissance de modèles et la classification statistique.

L'utilisation de ce type d'algorithmes a largement explosé dans ces dernières années dans différents domaines de l'informatique, et l'un de ces domaines est bel et bien le traitement des images numériques en générale et la compression en particulier.

Mots-clés : *Compression d'images, Apprentissage Automatique, Compression Fractale, Apprentissage Profond, Qualité de compression.*

Abstract

The purpose of image compression is to reduce the size of an image in order to facilitate its storage as well as its transfer.

Thus, we can distinguish between two main families of compression methods, those which cause the loss of information causing a reconstructed image which is not faithful to the original one, but of a very small size. The other method doesn't cause the loss of information but it reduces the compression rates.

The Optimization of rates and distortions has been, for a long time, considered as an intractable problem for images, which has prompted researchers to move widely in this direction using different techniques.

On the other side, deep neural networks use different layers of nonlinear processing units for the extraction and the transformation of characteristics; each layer takes as input the output of the previous one; algorithms can be supervised or unsupervised, and their applications include pattern recognition and statistical classification.

The use of this type of algorithms has exploded in recent years in various fields of computer science, and one of these fields is indeed the processing of digital images in general and compression in particular.

Keywords: Image Compression, Machine Learning, Fractal image compression, Deep Learning, Compression Quality.

ملخص

الغرض من ضغط الصورة هو تقليل حجم الصورة من أجل تسهيل تخزينها وكذلك تقليلها وبالتالي، يمكننا التمييز بين عائلتين رئيسيتين من طرق الضغط، تلك التي تسبب فقدان المعلومات التي تسبب صورة معاد بناؤها والتي ليست مخلصة لطريقة الضغط الأصلية، ولكن صغيرة الحجم جدا. الطريقة الأخرى ال تسبب فقدان المعلومات لكنها تقلل معدلات الضغط.

كان االستخدام المثل للمعدلات والتشوهات يعد منذ فترة طويلة مشكلة مستعصية على الحل بالنسبة للصور، المر الذي دفع الباحثين إلى التحرك على نطاق واسع في هذا التجاه باستخدام تقنيات مختلفة.

وعلى الجانب الآخر، تستخدم الشبكات العصبية العميقة طبقات مختلفة من وحدات المعالجة غير الخطية الستخراج وتحويل الخصائص؛ تأخذ كل طبقة كإدخال لمخرجات الطبقة السابقة؛ يمكن الإشراف على الخوارزميات أو عدم الإشراف عليها، وتشمل تطبيقاتها التعرف على النماط والتصنيف الإحصائي.

استخدام هذا النوع من الخوارزميات في السنوات الأخيرة في مختلف مجاالت علوم الكمبيوتر، وأحد هذه المجاالت هو في الواقع معالجة الصور الرقمية بشكل عام والضغط بشكل خاص

الكلمات المفتاحية: ضغط الصور ، التعلم الآلي ، ا ضغط الصور الكسوري ، التعلم العميق ، جودة الضغط.

Table des matières

Liste des figures

Liste des tables

Introduction générale

" "La réussite est la place qu'on occupe dans les journaux.

La réussite est l'insolence d'un jour."

Dave BARRY

La taille des images s'accroît comparativement à leur qualité, leur stockage et transmission qui conçoivent donc les enjeux primordiaux dans l'univers digital. La compression s'exige comme une étape inévitable pour améliorer l'usage de ces grands volumes d'informations dans les réseaux informatiques. L'objectif capital de la compression d'images est de diminuer la quantité d'information nécessaire à une représentation visuelle fidèle à l'image originale. En générale on différencie les méthodes de compression selon la perte d'informations. Les méthodes réversibles, utilisent uniquement le principe de la réduction de la redondance et n'engendrent pas de perte. Les méthodes irréversibles, définissent une représentation approximative de l'information.

Toutes ces méthodes présentent des avantages et des inconvénients ; le but et par contre de trouver un compromis entre les différents critères à vérifier après un algorithme de compression (temps de calcul ; taille de l'image ; dégradation totale…etc.), donc de concevoir un système de compression qui permet d'avoir un meilleur rapport (qualité / autres critères).

Dans ce Mémoire de Master, nous allons, et pour la première fois, mener étude plus détaillée sur un le developpement d'une nouvelle méthode de compression fractale d'image basée sur l'integration d'un algorithme d'apprentissage profond

Notre travail se concentre sur la réduction du nombre de correspondances entre les blocs de domaines et les blocs de plage, et la conservation des autres parties de l'ancienne méthode de compression, car une bon lecteure de partitionnement peu nous mener à une meilleure vitesse de compression.

Ce mémoire de Master est constitué en trois chapitres comme suit :

Une **introduction générale** présentant le contexte de l'étude qui renferme deux grandes axes à savoir la compression d'image et l'apprentissage automatique, on passe ensuite à l'objectif de notre recherche qui est l'élaboration d'un nouveau système de compression fractale basé sur l'utilisation d'un algorithme d'apprentissage profond.

Un premier chapitre. ***Généralités sur la compression d'images et l'apprentissage automatique***.

Ce chapitre est consacré à la présentation des principaux concepts de la compression et l'apprentissage automatique. Où on commencera les différentes méthodes de compression d'image des deux famille (réversible et irréversible), tout en mettant l'accent sur la compression fractale d'images. Cela va nous permettre de chercher les points faibles de quelques algorithmes (compression fractale) afin qu'on puisse introduire le mécanisme d'amélioration adéquat.

Dans la deuxième partie de ce chapitre nous aborderons le domaine d'apprentissage . Une taxinomie des Métaheuristiques d'optimisation est dressée à la fin de ce chapitre.

Le troisième chapitre. Métaheuristiques d'optimisation pour la compression d'image : travaux similaires.

Le deuxième chapitre. ***Approche proposée***

Dans ce chapitre nous nous intéressons, en premier lieu, aux travaux abordés dans le contexte de l'intégration des algorithmes d'apprentissage profonds dans la compression d'image. Le rest de chapitre sera consacré à expliquer notre approche proposée pour mettre en évidence une nouvelle méthode de compression fractale basée sur .

Le troisème chapitre. ***Résultats et discussion***

Ce dernier chapitre exposera les différents résultats obtenus après l'expérimentation faites sur des images standards.

Nous terminerons ce mémoire de Mastere par une conclusion générale et quelques perspectives ouvrant la porte pour d'autres travaux de recherche.

Chapitre 1

Généralités sur la compression d'images et l'apprentissage automatique

"L'informatique : alliance d'une science inexacte

Et d'une activité humaine faillible."

Luc Fayard

Introduction

Avec l'avancement de l'ère de l'information, la nécessité de stocker des informations de masse et les liens de communication rapides se développent. Stocker des images dans moins de mémoire entraîne une réduction directe en coûts de stockage et des transmissions de données plus rapides.

Les images sont stockées sur des ordinateurs sous forme de collections de bits représentant des pixels ou des points formant les éléments de l'image. Puisque l'œil humain peut traiter de grandes quantités d'informations (environ 8 millions de bits), de nombreux pixels sont nécessaires pour stocker des images de qualité modérée.

La plupart des données contiennent une certaine redondance qui peut parfois être supprimée pour stockage et remplacé pour la récupération, mais cette redondance ne conduit pas à une compression élevée ratios.

Les méthodes standard de compression d'image sont de plusieurs types. Actuellement les méthodes les plus populaires reposent sur l'élimination des composantes haute fréquence du signal en stockant uniquement les composants basse fréquence (algorithme discret de transformation en cosinus). Cette méthode est utilisée avec les formats JPEG (images fixes), MPEG (images vidéo animées) et H.261. Algorithmes de compression (téléphonie vidéo sur lignes RNIS).

D'autre part, les outils d'optimisation de la compression se multiplient et se diversifient. Les algorithmes d'apprentissage et d'apprentissage profond se présentent à

la tête des techniques introduites dans le domaine de la compression d'images pour plusieurs raisons : réduire la taille et le temps de compression, réduire le bruit et offrir de meilleur qualité pour les images reconstruites.

Avant de parler sur ces techniques, il est nécessaire de se familiariser avec les deux axes de recherche impliqués dans notre contexte d'étude, à savoir la compression d'images et l'apprentissage automatique.

I. Partie 1 : la compression fractale d'image

1.1 Définition de l'image

L'image est une représentation d'une personne ou d'un objet par la peinture, Dessin, photographie, films, etc. C'est également un ensemble organisé d'informations qui, après avoir été affiché à l'écran, a un sens pour l'œil humain.

Elle peut être décrite comme une fonction I (x, y) avec une luminosité analogique continue, définie dans un champ spécifique, de sorte que x et y sont les coordonnées spatiales d'un point dans l'image et I est une fonction de l'intensité lumineuse et de la couleur. Sous cet aspect, l'image n'est pas utilisable par l'appareil, ce qui nécessite sa numérisation. [1]

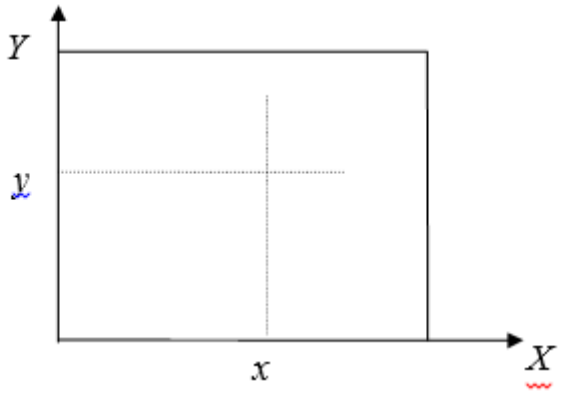

Figure 1.1. Un point de l'image de coordonnées (*x*, *y*) [1]

1.2 Image numérique

Une image numérique est une représentation d'une image réelle comme un ensemble de nombres qui peuvent être stockés et manipulés par un ordinateur numérique. Afin de traduire l'image en nombres, elle est divisée en petites zones appelées pixels (éléments d'image). Pour chaque pixel, le dispositif d'imagerie enregistre

un nombre, ou un petit ensemble de nombres, qui décrivent certaines propriétés de ce pixel, telles que sa luminosité (l'intensité de la lumière) ou sa couleur. Les nombres sont disposés dans un tableau de lignes et de colonnes qui correspondent aux positions verticales et horizontales des pixels de l'image. [2]

1.2.1 Les attributs de l'image

- Pixel.
- Dimension.
- Bruit.
- Résolution.
- Histogramme.
- Contours et textures.
- Luminance.
- Contraste.
- Le poids de l'image.

1.2.2 Les différents types d'image

- L'image en niveaux de gris.
- Mode monochrome.
- L'image en couleurs

1.2.3 Les formats d'image

- **Les images bitmap :**
    - Le format GIF (Graphic Interchange Format).
    - Le format JPG ou JPEG (Joint Photographique Experts Group).
    - Le format PCX (Picture Exchange Image Bitmap Zsoft.
- **Les images vectorielles :**
    - Le format EPS (Postscript / Encapsulated Postscript).
    - Le format CGM (Computer Graphics Metafile).

1.2.4 Qualité de l'image numérique

La résolution d'une image numérique est l'un des nombreux facteurs qui déterminent la qualité d'une photo numérique. Il existe quatre principaux facteurs qui fonctionnent ensemble pour créer une qualité photo numérique [3] :

• La qualité de l'appareil d'enregistrement (optique et capteur de la caméra, capteur du scanner). [3]

• La taille (en pixels) de l'image numérique. [3]

• Le format numérique dans lequel il est stocké (compression sans perte vs compression avec perte). [3]
• La compétence technique et "l'œil" du photographe. [3]

1.3 Pour Quoi compresser ?

La manipulation des images pose cependant des problèmes beaucoup plus complexes que celles du texte. En effet, l'image est un objet à deux dimensions, censé de représenter un espace à trois dimensions, ce qui pose deux problèmes majeurs [1] :

• Le volume des données à traiter est beaucoup plus important (Les problèmes de stockage, de traitement et de transmission qui apparaissent rapidement avec la taille des images). [1]
• La structure de ces données est nettement plus complexe, se heurtant à certaines limitations (grâce au traitement d'image, ces contraintes sont résolues ou contournées). [1]

Pour en savoir plus, nous présentons quelques exemples, qui donnent une idée des besoins qu'implique l'utilisation de la technologie de la numérisation, en compression comme solution aux problèmes de stockage et de transmission de l'image [1]:

• Une image couleur représentée dans l'espace *Rouge-Vert-Bleu*, de taille 512*x*512 pixels dont chacune des composantes est codée sur 8 bits par pixel représente 786 *K* octets.
• Une image de 800*x*600 pixels en 16 millions de couleurs (24 bits par pixels), correspondant à un fond d'écran, occupe 1 millions 400 000 octets. [1]
• La transmission d'une séquence d'images couleur au format *QCIF* échantillonnée à 30 *Hertz* représente un débit de 9.12 *M* bits par seconde, et de 36.40 *M* bits par second
• au format *CIF* (Le format de *QCIF* [Quart de *CIF*] est défini par 167 pixels sur 144 lignes pour la luminance et 88*x*72 pixels pour la chrominance. [1] Le format *CIF* [Common intermediate format] est défini par 352 pixels sur 288 lignes pou la luminance et 176*x*144 pixels pour la chrominance). [1]
• Un film négatif 24*x*36 *mm* numérisé à 12□*m* par point retourne une image de taille 3000*x*2000 pixels par couleurs, 8 *bpp*, 3 couleurs ce qui représente 18 *M* octets. [1]

- La transmission d'une séquence vidéo 512x512, 8 *bpp*, 3 couleurs sur une ligne téléphonique avec modem à 9600 bauds nécessite 11 minutes par image. [1]
- Une image *LANDSAT* : 6000*6000 pixels par bande spectrale, 8 bits par pixel, et 6 bandes, soit 1.7*109 bits. [1]
- Une image de télévision basse résolution contient trois composantes ou couleurs, 512x512 pixels par couleur et 8 bits par pixel, soit un total de 6x106 bits. [1]

✓ La compression et le codage consiste en la réduction de la taille physique d'un bloc d'information (en réduisant le nombre de bits par pixel à stocker ou à transmettre), en exploitant la redondance informationnelle dans l'image. [1]

✓ La compression et le codage consiste en la réduction de la taille physique d'un bloc d'information (en réduisant le nombre de bits par pixel à stocker ou à transmettre), en exploitant la redondance informationnelle dans l'image. [1]

1.4 Mesures de performance de la compression d'image

- **Rapport et taux de compression**

Le rapport de compression est l'une des caractéristiques les plus importantes de toutes les méthodes de compression, il représente le rapport entre le nombre de bits de la forme canonique au nombre de bits après codage [1] :

$$\boldsymbol{Rapport = CR = \frac{nombre\ de\ bits\ Io}{nombre\ de\ bits\ Ic}}$$

Par conséquent le taux de compression est un pourcentage de l'espace obtenu après la compression par rapport à l'espace total requis les données avant la compression. [1]

Il est défini par :

$$\boldsymbol{Taux = \left(1 - \frac{1}{CR}\right) * 100}$$

- **Entropie**

Dans une image, L'entropie est une grandeur qui caractérise la qualité de l'information que contient cette dernière. Par exemple une image dont tous les pixels ont la même valeur contient très peu d'informations car elle est extrêmement redondante, son

entropie est faible. En revanche une image dont tous les pixels ont une valeur aléatoire contient beaucoup d'information, son entropie est forte. [1]

En pratique, l'entropie d'une image numérique est inversement liée à la probabilité d'apparition des niveaux de gris dans l'image. Par définition l'entropie d'ordre zéro H 0 est donnée par :

$$H_0 = -\sum_{k=0}^{2^R-1} p(k) * ln\, p(k)\ ppb$$

Avec : $P(K)$ est la probabilité d'apparition des niveaux de gris dans l'image, K est la valeur de gris et R est le nombre de bits par pixels. [1]

L'entropie H0 d'une image originale fournit le débit minimal qu'il est possible d'atteindre par compression, pixel par pixel sans dégrader l'image, est par la même, un taux de compression sans perte maximal. [1]

On peut cependant par détérioration de l'image dépassé l'entropie comme indiquée sur la courbe E (Quantité d'information, Détérioration) ci-dessous :

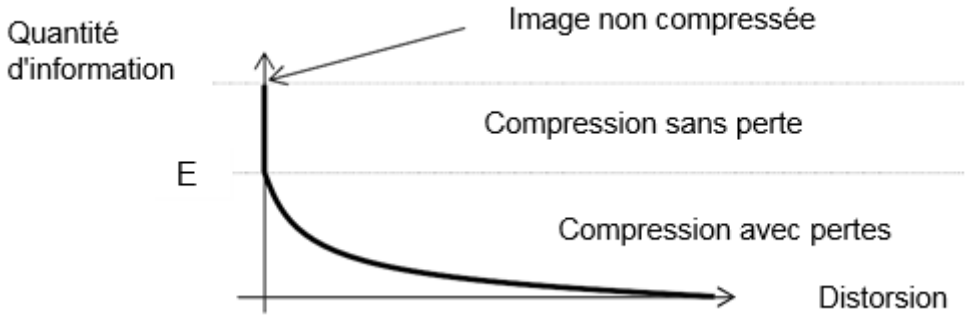

Figure 1.2. la détérioration de l'image en fonction de l'entropie. [1]

- **Mesures de distorsion**

Les exemples les plus importants pour les mesures mathématiques de la distorsion sont l'erreur quadratique moyenne, le Rapport signal / bruit de crête et erreur maximale. [4] :

$$MSE = \frac{1}{M * N} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} [I(m,n) - I'(m,n)]$$

Le rapport signal sur bruit (*Signal to Noise Ratio*) est défini par :

$$\boldsymbol{SNR = 10log_{10}\frac{\sum_m \sum_n [x(m,n)^2]}{MSE} DB}$$

Le signal à bruit de crête pour une image dont le maximum est 2^R -1 dénoté par *PSNR* (*Peak Signal to Noise Ratio*) est déterminé par la formule :

$$\boldsymbol{PNSR = 10\log_{10}\frac{(2^R - 1)}{MSE}\ décibels\ DB}$$

En compression d'image le *PSNR* d'une image de taille $8x(512)^2$ bits ($(512)^2$ indique une image de taille 512 par 512 pixels, chaque pixel est codé sur 8 bits) est défini plus souvent par

$$\boldsymbol{PNSR = 10\log 10\frac{(255)^2}{MSE}}$$

Les mesures de distorsion sont très utiles pour déterminer la performance d'une méthode par rapport à d'autres méthodes. [4]

1.5 L'intérêt de compression d'image

La compression et le codage consiste à la réduction de la taille physique d'un bloc d'information (réduction du nombre de bits par pixel à stocker ou à transmettre), en exploitant la redondance informationnelle dans l'image. Trois sortes de redondances sont exploitées dans la compression d'images [1] :

- La redondance spatiale entre pixels ou blocs voisins dans l'image. [1]
- La redondance temporelle entre images successives dans une séquence. [1]
- La redondance spectrale entre plans de couleur ou bandes spectrales. [1]

Les principaux critères d'évaluation de toute méthode de compression sont :

- la mesure de qualité : la qualité d'un système se mesure par la qualité de reconstruction de l'image. [1]
- La rapidité du codeur et décodeur. [1]
- La réduction des débits : le taux de compression dépend de l'application. [1]

1.6 Model générale pour l'analyse des méthodes de compression

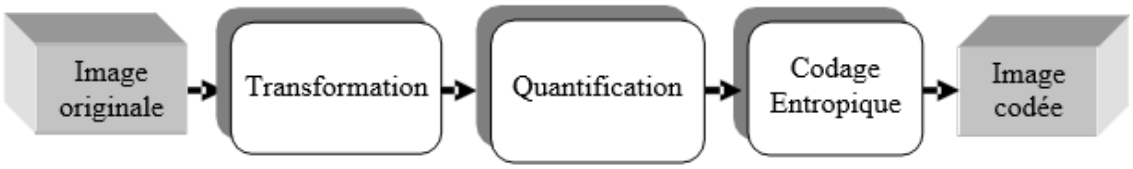

Figure 1.3. Schéma général de la compression [5]

- **Transformation ou décorrélation**

La dépendance existante entre chacun des pixels et ses voisins (la luminosité varie très peu d'un pixel à un pixel voisin) traduisent une corrélation très forte sur l'image. La décorrélation consiste à transformer les pixels initiaux en un ensemble de coefficients moins corrélés pour réduire le volume d'information, c'est une opération réversible. [5]

- **Quantification**

La quantification des coefficients a pour but de réduire le nombre de bits nécessaires pour leurs représentations. Elle représente une étape clé de la compression. Elle approxime chaque valeur d'un signal par un multiple entier d'une quantité q, appelée quantum élémentaire ou pas de quantification. Elle peut être scalaire ou vectorielle. [5]

- **Le codage entropique**

Le codage entropique effectue un codage sans perte sur les valeurs quantifiées. Cette dernière étape est nécessaire dans les méthodes sans perte, mais elle est souvent présente aussi dans les algorithmes irréversibles, puisque les valeurs transformées et quantifiées contiennent davantage de redondances. [5]

1.7 Méthodes de compression

- **Les méthodes réversibles ou sans pertes**
  - Le codage par répétition ou "Run Length Coding" (*RLC*)
  - Codage de *SHANNON-FANO*
  - Le codage de *HUFFMAN*
  - Le codage arithmétique
  - Codage par dictionnaire adaptatif (*LZW*) (*Lempel-Ziv-Welch*) ou *LZ77*

- **Les méthodes de compression avec pertes ou irréversible**
  - Quantification
  - Codage par transformée
  - Le codage en sous-bandes
  - La compression par ondelettes
  - ***La compression fractale***

1.8 La compression fractale d'images ?

Imaginez un type spécial de photocopieuse qui réduit l'image à copier par moitié et le reproduit trois fois sur la copie. Qu'est-ce qui se passe quand on nourrit la sortie de cette machine en entrée? La figure 1.13 montre plusieurs itérations de ce processus sur plusieurs images d'entrée. On peut constater que toutes les copies semblent converger vers la même finale image. Dans la mesure où le copieur réduit l'image d'entrée, toute image initiale placée sur la machine à copier sera réduite à un point car nous la faisons fonctionner à plusieurs reprises ; en fait, il est seulement la position et l'orientation des copies qui détermine ce que l'image finale ressemble à. [6]

La transformation de l'image d'entrée détermine le résultat final lors de l'exécution de la copie. Machine dans une boucle de rétroaction. Cependant, nous devons limiter ces transformations, avec le limitation que les transformations doivent être contractives (voir encadré contractif), c'est-à-dire une transformation donnée appliquée à deux points quelconques dans l'image d'entrée doit les rapprocher de la copie. Cette condition technique est assez logique, puisque si des points de la copie étaient répartis l'image finale devrait être de taille infinie. Sauf pour cette condition, la transformation peut avoir n'importe quelle forme. En pratique, choix des transformations de la forme [6]

$$w_i \begin{vmatrix} x \\ y \end{vmatrix} = \begin{vmatrix} a_i & b_i \\ c_i & d_i \end{vmatrix} \begin{vmatrix} x \\ y \end{vmatrix} + \begin{vmatrix} e_i \\ f_i \end{vmatrix}$$

où x, y - coordonnées;

a, b, c, d, e, f - coeficients,

est suffisant pour générer des transformations intéressantes appelées transformations affines du avion. Chacun peut incliner, étirer, faire pivoter, redimensionner et traduire une image d'entrée. [6]

Une caractéristique commune de ces transformations qui s'exécutent en mode de boucle de retour est que pour une image initiale, chaque image est formée à partir d'une copie transformée (et réduite) d'elle-même, et il doit donc avoir des détails à to[3]utes les échelles. C'est-à-dire que les images sont des fractales. [6]

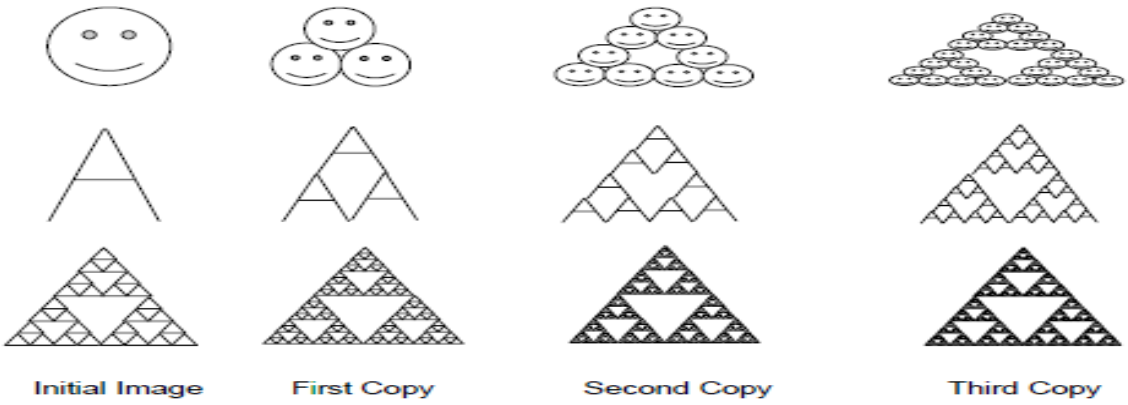

Figure 1.4. Les trois premiers exemplaires de l'image initiale. [6]

Dans, il a été suggéré que peut-être stocker des images en tant que collections de transformations pourrait conduire à la compression d'image. Son argument était le suivant : l'image de la figure 1 ressemble compliqué mais il est généré à partir de seulement 4 transformations affines. Chaque transformation wi est définie par 6 nombres ai, bi, ci, di, ei et fi, qui ne nécessitent pas beaucoup de mémoire à stocker sur un ordinateur (4 transformations x 6 transformations de chiffres x 32 bits / nombre = 768 bits). Stocker l'image en tant que collection de pixels a cependant demandé beaucoup plus de mémoire. Donc, si nous souhaitons stocker une image d'une fougère, nous pouvons le faire en stockant le nombres qui définissent les transformations affines et génèrent simplement la fougère chaque fois que nous. Supposons maintenant que nous ayons reçu une image quelconque, disons un visage. Si un petit nombre de transformations affines pourrait générer ce visage, alors il pourrait aussi être stocké compactement. L'astuce consiste à trouver ces chiffres. [6]

1.8.1 Transformations Contractive

Une transformation w est dite contractive si pour deux points quelconques P1, P2, la distance

$$d(w(P1),w(P2)) < sd(P1, P2)$$

pour certains s <1, où d = distance. Cette formule dit l'application d'une map contractive toujours rapproche les points. [6]

1.8.2 Le théorème de contraction mapping en point fixe

Ce théorème dit quelque chose qui est intuitivement évident: si une transformation est contractive puis lorsqu'il est appliqué à plusieurs reprises à partir de tout point initial, nous convergeons vers un unique fixe point. Si X est un espace métrique complet et que W: X-> X est contractif, alors W a une valeur unique point fixe W. [6]

Ce théorème simple nous dit comment nous pouvons nous attendre à un ensemble de transformations définir une image. On peut dire que le schéma de compression d'image décrit plus tard est fractal à plusieurs égards Le schéma va coder une image en tant que collection de transformations très similaires à la métaphore de la photocopieuse. Tout comme la fougère a des détails à toutes les échelles, l'image reconstruit à partir des transforme. L'image décodée n'a pas de taille naturelle, elle peut être décodée à n'importe quelle taille. Les détails supplémentaires nécessaires au décodage à des tailles supérieures sont générés automatiquement par le codage transforme. On peut se demander si ce détail est "réel"; nous pourrions décoder une image d'une personne augmentant la taille à chaque itération, et éventuellement voir cellules de la peau ou peut-être des atomes. La réponse est, bien sûr, non. Le détail n'est pas du tout lié à le détail présent lors de la numérisation de l'image ; c'est juste le produit de le codage transforme ce qui à l'origine ne codait que les entités à grande échelle. Cependant, dans certains cas, les détails sont réalistes à faible grossissement. Le taux de compression pour le schéma de fractal est difficile à mesurer car l'image peut être décodée à n'importe quelle échelle. [6]

1.8.3 Encodage des images

Le théorème précédent nous dit que la transformation W aura un point fixe unique dans l'espace de toutes les images. C'est-à-dire que quel que soit l'image (ou l'ensemble) avec laquelle nous commençons, nous pouvons répéter appliquons W it et nous convergerons vers une image fixe. [6]

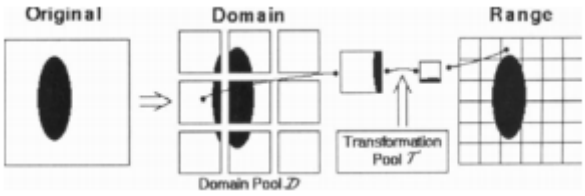

Figure 1.5. Le processus d'encodage. [6]

Supposons qu'on nous donne une image f que nous souhaitons encoder. Cela signifie que nous voulons trouver une collection de transformations w1, w2,… wN et veux que f soit le point fixe de la carte W. Autrement dit, nous voulons partitionner f en morceaux auxquels nous appliquons les transformations wi et récupérer l'image originale f. L'image typique d'un visage, ne contient pas le type d'auto-similitude. L'image contient d'autres types d'auto-similarité. Ces pièces transformées font ne s'emboîtent pas, en général, pour former une copie exacte de l'image originale, et nous devons donc autoriser certaines erreurs dans notre représentation d'une image sous la forme d'un ensemble de transformations. [6]

1.8.4 Les méthodes de compression fractale

Il existe plusieurs méthodes de compression fractale, et nous avons choisis de présentes trois :

- La compression Jacquin.
- La compression par subdivisions successives de triangles.
- La compression de Delaunay.

Ces trois algorithmes sont fondés sur une même idée : Définir une zone dans l'image (un carre ou un triangle suivant la m´méthode utilisée) [Source], lui appliquer des transformations fondamentales (réduction, normalisation de la moyenne, rotation etc...)

et ré-implanter le résultat dans l'image dans une zone plus réduite [Destination]. La zone de destination doit être la plus 'proche' possible du résultat. [7]

La répétition de ce modèle dans toute l'image, et ceci de manière itératif, implique une certaine convergence de l'image reconstruite par l'assemblage de transformations/réductions locales, vers l'image d'origine. [7]

✓ Méthode Jacquin

L'algorithme de compression par la méthode Jacquin correspond parfaitement `à l'algorithme générique présente dans la première partie. La m´méthode Jacquin utilise des régions carrées pour segmenter l'image. Cet algorithme de compression/décompression est le plus simple des trois algorithmes présentés dans ce document, car la manipulation de régions carrées prédéfinies (tableaux 2D de tableaux 2D) est beaucoup plus simple à implémenter que les régions triangulaires dynamiques (liste chainée de structures) [7]

Voici le fonctionnement de la compression fractale par la méthode de Jacquin. [7]

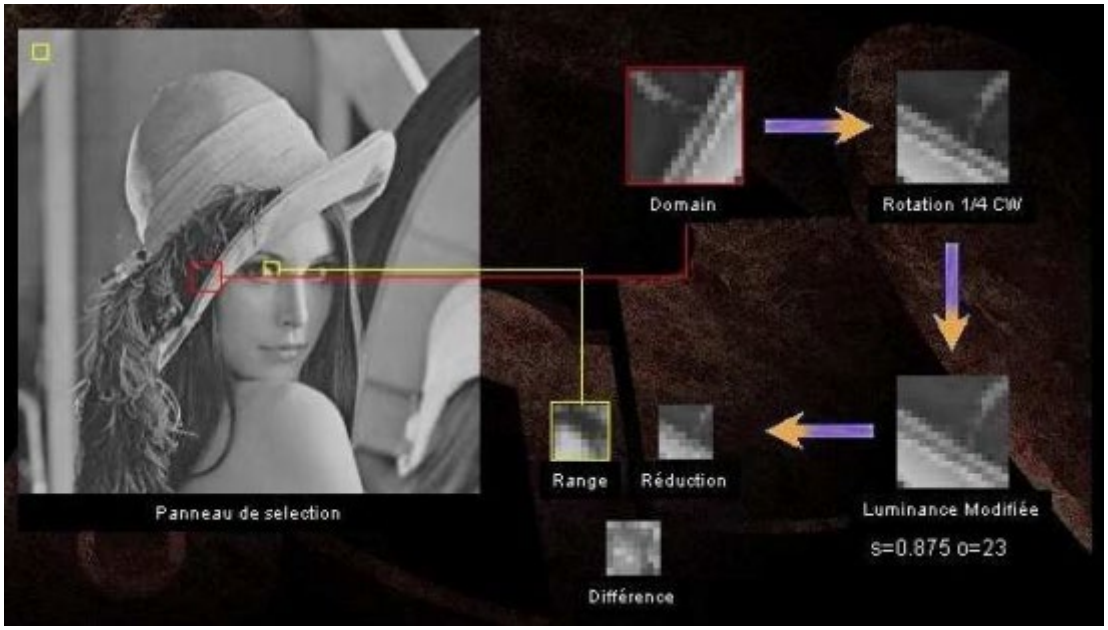

Figure 1.6. Méthode de compression Jacquin. [7]

On commence par générer sur l'image un pavage de carrés appelés Sources et un pavage de carrés Destinations. [étape 0].

Puis, on cherche quelle est la meilleure combinaison Source/Destination pour chaque figure Source. Pour cela,

- ✓ On applique différentes transformations (flip et/ou rotation, réduction de l'intensité) sur le carré Source courant (comme le montre la figure précédente) [étape 1].
- ✓ On réduit sa taille pour le rendre de la même hauteur que la figure destination [étape 2].
- ✓ Et on calcule l'erreur entre les deux entités [étape 3].

L'erreur est ´évaluée grâce `a l´écart type des pixels des 2 régions. Si cette erreur est minimale, on sauvegarde le couple Source/Destination ainsi que les modifications apportées. [7]

Pour recréer l'image à partir du fichier final, il suffit de recréer les 2 partitions source et destination, et d'appliquer plusieurs fois les transformations des couples sauvés dans le fichier. Le gène fractal de l'algorithme de Jacquin garantit alors la convergence de l'image vers l'image de départ. [7]

- **Partitionnements**

Le partitionnement est l'opération qui consiste à segmenter une image en régions. Dans la compression par la méthode Jacquin, nous avons besoin de 2 partitionnements : Source et Destination. Comme nous l'avons vu précédemment, la méthode Jacquin utilise des figures carrées. Voici les pavages réalisés lors de la compression et décompression Jacquin [7] :

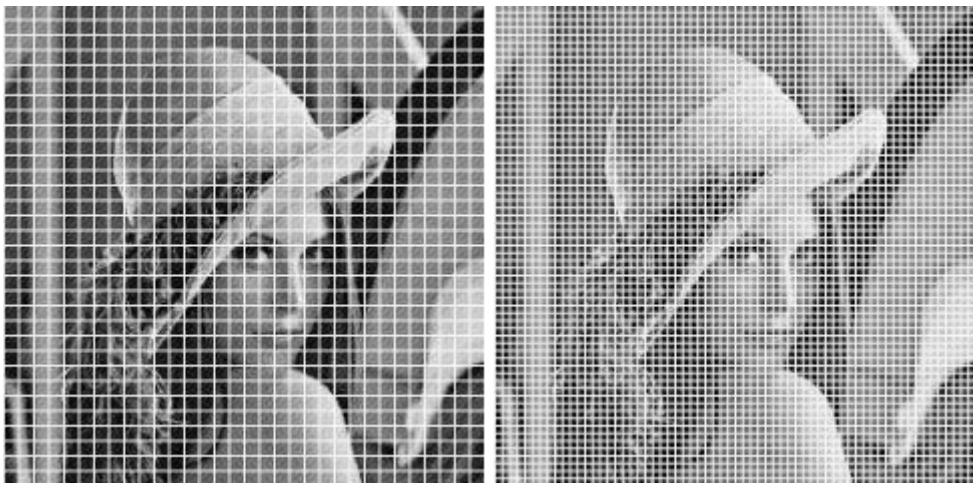

Figure 1.7. Pavages Source et Destination, Méthode Jacquin. [7]

Un point essentiel dans les partitionnements Source et Destination est que le pavage destination doit être plus petit que le pavage source. En effet, dans le cas contraire, nous serions amenés à faire un agrandissement (et non une réduction) lors de la transposition des figures sources vers les figures destinations. Une fractale possède un motif se répètent à l'infini, en se rétrécissant. Aussi, nous perdons cette propriété si le partitionnement destination est plus grand que le partitionnement source, l'image ne pourra alors pas converger. [7]

✓ Méthode par subdivisions de triangles

La méthode par subdivisions de triangles est quasiment identique à la méthode Jacquin. Elles possèdent cependant deux différences majeures. D'une part, nous travaillons maintenant sur des triangles. Et d'autre part, un module de subdivision des triangles a été ajouté. [7]

Pour savoir si un point (ou un pixel) appartient ou non à un triangle, nous avons choisi de modéliser un triangle à l'aide de 3 droites d1, d2 et d3, comme le montre la figure suivante [7] :

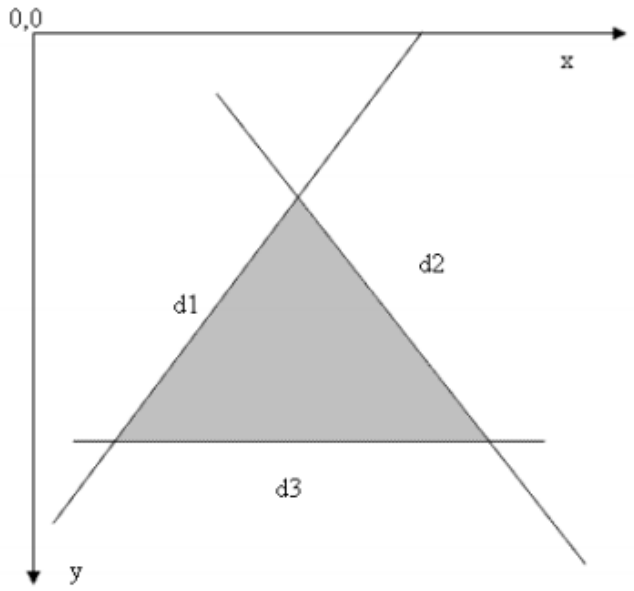

Figure 1.8. Appartenance d'un point à un triangle. [7]

Pour connaître l'appartenance d'un point à un triangle, il suffit de vérifier les équations définies ci-dessous :

$$P.x \geq d1 \quad P.y \geq d1$$

$$P.x \leq d2 \quad P.y \leq d2$$

$$P.x \leq d3 \quad P.y \leq d3$$

Si un point P (de coordonnées (P.x,P.y)) appartient au triangle formé par les segments d1, d2 et d3, alors il vérifie ces inéquations. [7]

Une fois que nous avons conçu le module de gestion d'un pixel dans un triangle, nous avons simplement repris toutes les fonctions utilisées par la compression Jacquin, et nous les avons transformées pour qu'elles manipulent non plus des carrés, mais des triangles. [7]

- **La subdivision**

Comme nous l'avons vu précédemment, la compression fractale par subdivisions successives de triangles, découpe l'image à traiter en un pavage de triangles. Pour améliorer l'efficacité de l'algorithme de compression, une division des triangles ayant un nombre important de détails est ajoutée. Pour connaître le niveau de détails d'une zone de l'image (couverte par un triangle considéré), nous calculons simplement l'écart type des pixels de celui-ci. Plus cette valeur sera importante, et plus le triangle est loin d'être uniforme. L'opération de subdivision sera alors lancée. [7]

Le triangle sera divisé en 6 plus petits triangles comme le montre la figure suivante :

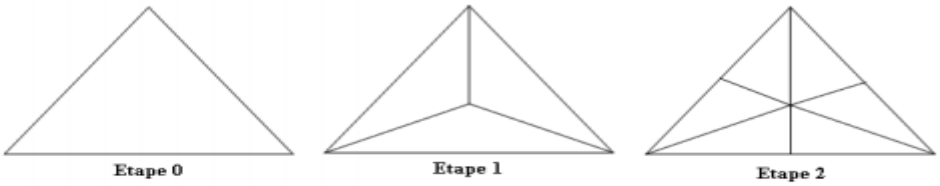

Figure 1.9. Subdivision d'un triangle. [7]

A l'étape 1, nous division une première fois le triangle en 3 triangles, qui seront formés d'une part, avec les 3 points du triangle parent, et d'autre part avec son barycentre. L'étape 2 va ensuite scinder les 3 nouveaux fils en 2, en utilisant la médiane (partant de l'ancien barycentre) de chacun. [7]

Cette subdivision améliore considérablement le rendu de l'image compressée. La perte de qualité est diminuée, mais le temps de compression impose une profondeur de division restreinte. De plus, la taille du fichier de sortie augmente car il est nécessaire de stocker les différentes subdivisions. Un simple booléen suffit (1 : triangle divisé, 0 : triangle non subdivisé). [7]

- **Transformation d'un triangle**

Le processus discrétisé d'une réduction, qu'elle soit avec un carré (méthode de Jacquin) ou avec un triangle, reste identique. Il s'agit simplement de caractériser une matrice de transformation des coordonnées d'une figure de départ, vers les coordonnées d'une figure d'arrivée (de même topologie dans notre étude). Prenons l'exemple de deux triangles :

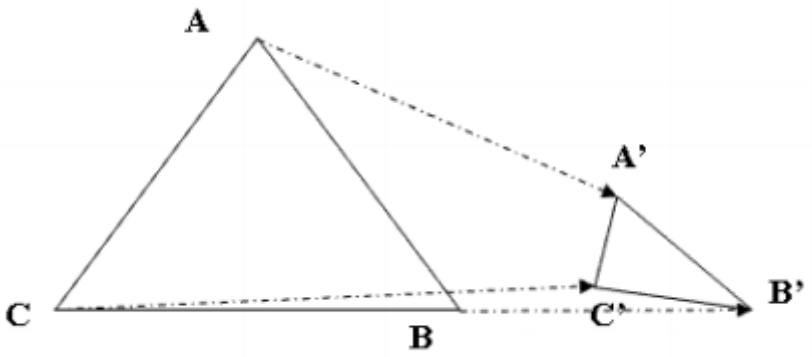

Figure 1.10. Transformation d'un triangle. [7]

La matrice de correspondance des 3 points A,B et C vers les 3 points A', B' et C' est telle que :

$$\begin{vmatrix} Ax & Ay & 1 & 0 & 0 & 0 \\ Bx & By & 1 & 0 & 0 & 0 \\ Cx & Cy & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & Ax & Ay & 1 \\ 0 & 0 & 0 & Bx & By & 1 \\ 0 & 0 & 0 & Cx & Cy & 1 \end{vmatrix} * \begin{vmatrix} a \\ b \\ c \\ d \\ e \\ f \end{vmatrix} = \begin{vmatrix} A'x \\ B'x \\ C'x \\ A'y \\ B'y \\ C'y \end{vmatrix}$$

Matrice de correspondance des points. [7]

La détermination des coefficients a,b,c,d,e et f caractérise donc entièrement la transformation, d'un triangle dans ce cas précis. [7]

Remarque :

Les rotations et les flips peuvent être aisément réalisés grâce à cette matrice de transpositions. En effet, si l'on associe le point A au point B', B au point C' et C au point A', nous obtenons alors une rotation de $\frac{2\pi}{3}$ (environ). Diverses combinaisons sont d`es lors possibles (Flips horizontaux et verticaux, combinaisons linéaires de rotations et de flips etc...). [7]

Voici ce que nous obtenons après avoir effectué une transformation d'un triangle dans l'image lena.jpg :

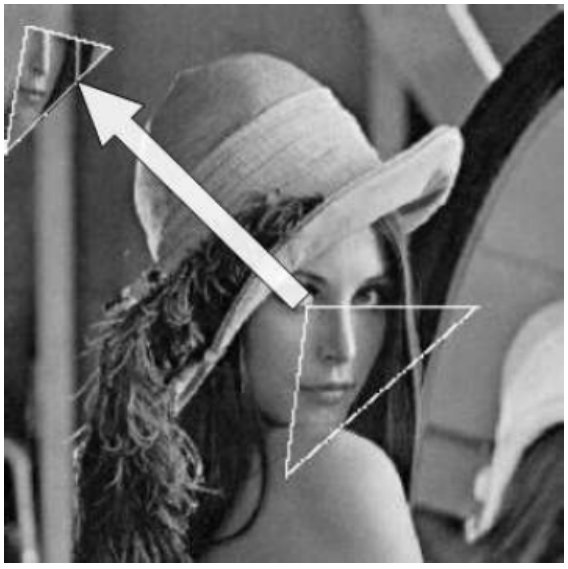

Figure 1.11. Réduction d'un triangle. [7]

Le triangle en haut à gauche est donc le résultat de la transformation, sans aucun flip, ni aucune rotation, du triangle plus grand au milieu de l'image. Le morceau d'image contenu dans le triangle source est donc bien réduit et translaté dans le triangle destination. [7]

- **Partitionnements**

Le partitionnement des figures Sources et Destinations par la méthode de subdivision de triangles sont identiques à ceux de la méthode de Jacquin. Il s'agit ici, de réaliser un pavage de triangles (et non plus de carrés), avec un pas déterminé. Le pavage Destination est bien entendu plus compacte que le pavage Source, pour conserver le caractère fractaléen de la compression. [7]

La figure suivante montre les deux pavages Source et Destination sur l'image lena.jpg. Les triangles destinations ont subi le processus de subdivision une seul fois. [7]

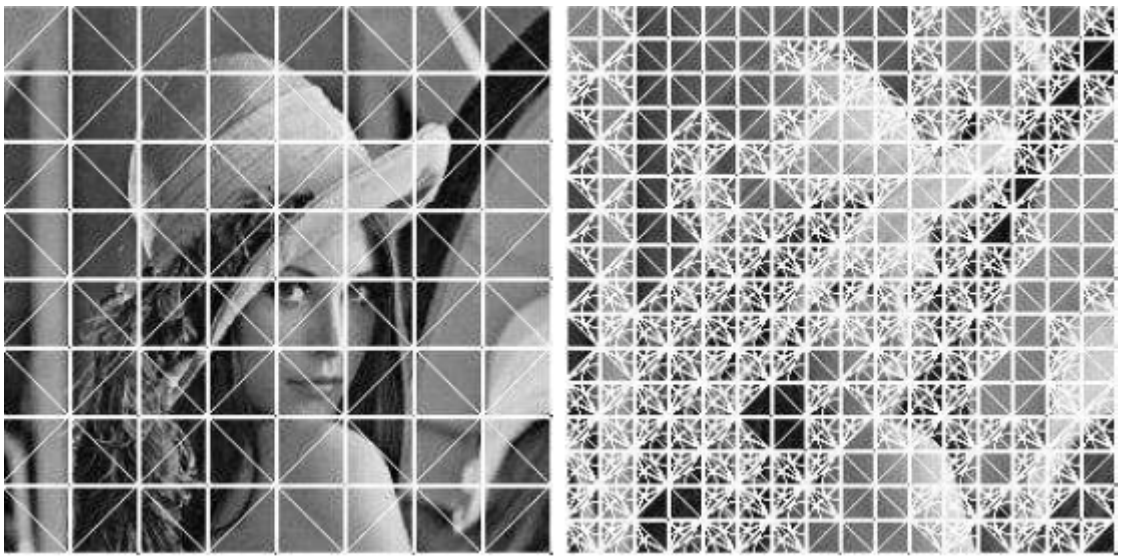

Figure 1.12. Pavages Source et Destination, Méthode par Subdivision de Triangles. [7]

Nous remarquons alors, que les détails de l'image sont encadrés par de plus petits triangles. La qualité est par conséquent augmentée. [7]

Une amélioration possible de la subdivision de triangles est de ne plus définir le nombre de subdivision par une constante, mais de lancer ce processus tant qu'il existe un triangle non_homogène. [7] Cela améliorera certainement la qualité certes, mais cela augmentera aussi le temps de compression, qui est déjà très élevé. [7]

✓ Méthode de Delaunay

La compression de Delaunay, cette méthode manipule tout comme la Subdivision de triangles, des figures triangulaires pour segmenter l'image. Mais au lieu de partitionner suivant un pavage régulier de triangles, la méthode de Delaunay utilise comme son nom

l'indique, un pavage formé par les triangles de Delaunay. Seul, le pavage Destination est formé ainsi, le partitionnement Source étant un simple pavage réguler. [7]

Pour augmenter encore la qualité de restitution des détails lors de la reconstruction de l'image, l'algorithme de compression de Delaunay segmente l'image en triangles de Delaunay. Mais avant de définir les triangles de Delaunay, il nous faut présenter le diagramme de Voronoi. [7]

- **Diagramme de Voronoï**

On désigne par P un ensemble composé de n points de l'espace R^2 appelés aussi sites ou germes. [7]

$$P = \{pi \in R^2, i = 1, .., n\}$$

On appelle polygone de Voronoï associé au site Pi la région Vor(Pi) (chaque région étant l'ensemble de points (x,y) les plus proches `a un point de P) telle que chaque point de P a pour plus proche site Pi. [7]

$$V_{or\,p}(P_i) = \{x \in R^2, d(x, P_i) \leq d(x, P_j), \forall P_j \in P - P_j\}$$

où d représente la distance Euclidienne. [7]

L'union de l'ensemble des polygones de Voronoï est appelé Diagramme de Voronoï. Un sommet de Voronoï est la rencontre de plusieurs arrêtes. [7]

Voici quelques germes déposés au hasard de l'image, ainsi que leur polygone de Voronoï associé. [7]

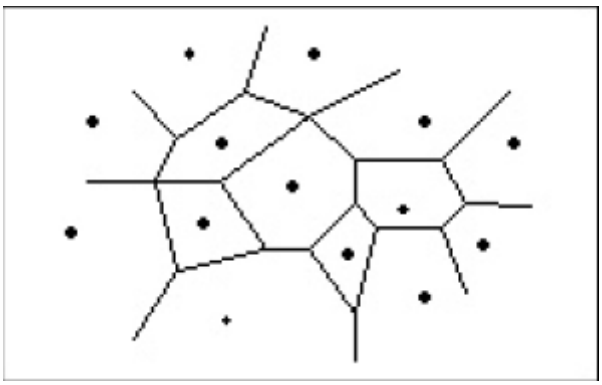

Figure 1.13. Diagramme de Voronoï[7]

Le diagramme de Voronoï possède quelques propriétés remarquables :

- les polygones sont convexes.
- le diagramme couvre toute l'image.
- une arrête de Voronoï sépare tout point de son plus proche voisin.
- chaque sommet des polygones possède exactement 3 arrêtes.
- pour chaque sommet S du diagramme de Voronoï, le cercle passant par les trois points voisins à ce sommet, ne contient aucun autre point de P.

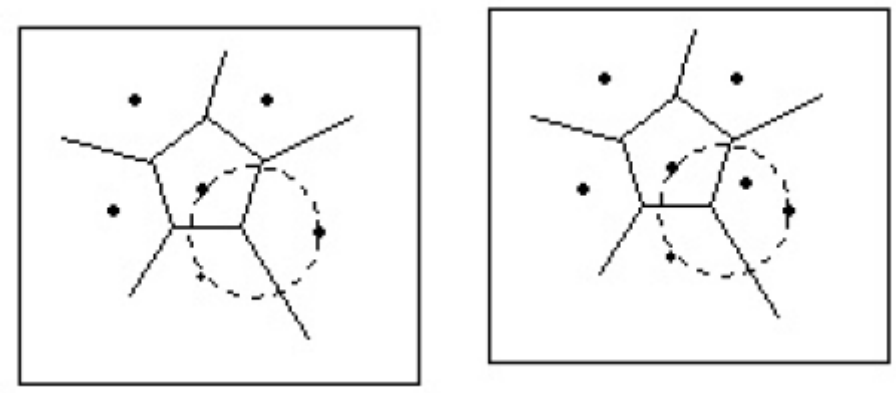

Figure 1.14. Propriétés d'un diagramme de Voronoï

Dans la figure de gauche, le cercle passant par les trois points voisins d'un sommet de Voronoï, ne contient aucun autre germe de Voronoï. A l'inverse, la figure de droite n'est pas un diagramme de Voronoï car un site est présent à l'intérieur du cercle. [7]

A partir du diagramme de Voronoï, nous pouvons construire le dual, appelé alors la triangulation de Delaunay. [7]

- **Triangulation de Delaunay**

La construction des triangles de Delaunay à partir d'un diagramme de Voronoï est assez simple. En effet, l'algorithme de Delaunay recherche premièrement les germes voisins (donc séparés par un arrête de Voronoï), pour chaque germe du Diagramme, et les relie entre eux. Comme le montre la figure suivante :

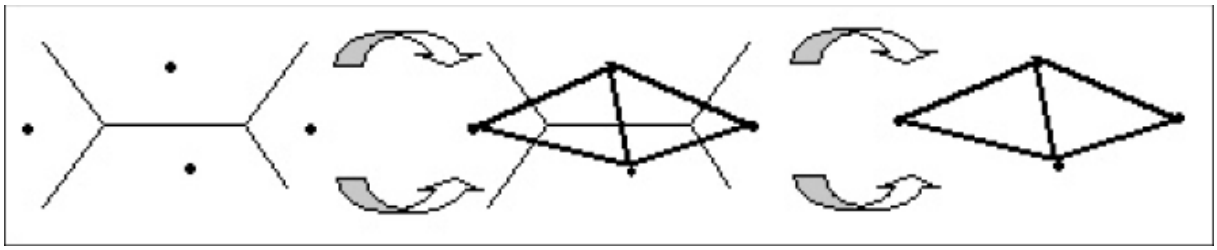

Figure 1.15. Du diagramme de Voronoï aux Triangles de Delaunay. [7]

Notons que seul le couple de points (point à gauche, point à droite), ne possède pas d'arrêté de Voronoï, ils ne sont donc pas reliés (figure de droite). Cet algorithme garantit la génération de triangles, ce pourquoi il est appelé Triangulation de Delaunay. Une triangulation de Delaunay est unique, et hérite des propriétés du Diagramme de Voronoï (les triangles ne se recouvrent pas etc...). [7]

Voici une illustration du Diagramme de Voronoï, et de sa Triangulation de Delaunay réciproque. [7]

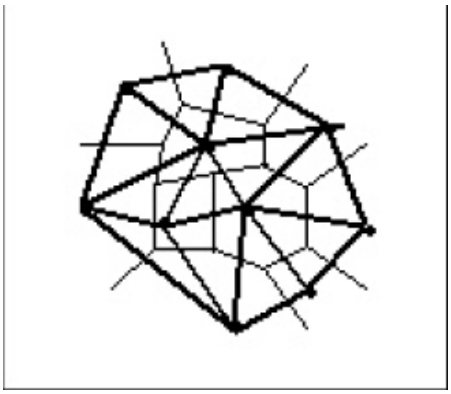

Figure 1.16. Diagramme de Voronoï, Partition de Delaunay. [7]

La Triangulation de Delaunay livre par conséquent une partition de l'image sous la forme de figures triangulaires. Nous pouvons d`es lors entreprendre une compression. [7]

II Partie 2 : Deep Learning

1.9Apprentissage automatique

L'apprentissage automatique est un sous-domaine de l'intelligence artificielle, où le terme fait référence à la capacité des systèmes informatiques à trouver indépendamment des solutions aux problèmes en reconnaissant les modèles dans les bases de données. En d'autres termes : le Machine Learning permet aux systèmes informatiques de

reconnaître des modèles sur la base d'algorithmes et d'ensembles de données existants et de développer des concepts de solutions adéquats. Par conséquent, dans le Machine Learning, la connaissance artificielle est générée sur la base de l'expérience. [8]

- **Apprentissage Supervisé**

Cette approche a pour objectif la conception d'un modèle reliant des données d'apprentissage à un ensemble de valeurs de sortie (un comportement). Un expert est employé pour étiqueter correctement des exemples. L'apprenant doit alors trouver ou approximer la fonction qui permet d'affecter la bonne étiquette à ces exemples. [9]

La **Figure II.1.1.** représente le schéma de l'apprentissage supervisé, le processus se déroule en deux phrases. La première est appelée la phase d'apprentissage durant laquelle le professeur fournit une étiquette à chaque exemple disponible dans l'environnement en les associant à une classe : x1,x2,...,xm,... et xn, xO,... dans le cas présenté ici. Ensuite il classifie les exemples en élaborant un échantillon de données étiquetées noté : Sm =(x1, U1), (x2,U2),…, (xm,Um). Il guide ainsi l'apprenant en lui fournissant des exemples sur ce qu'il doit apprendre à savoir faire. [9]
La seconde étape est nommée phase de test. Elle consiste à tenter de prédire l'étiquette d'un nouvel exemple en utilisant une fonction apprise à partir de Sm notée h. Avec cette hypothèse, l'apprenant peut associer Yn=h(xn) à chaque xn, Yo=h(xo) pour chaque xo, et etc... Plus ces valeurs seront proches de celles que le professeur aurait fournies à sa place, meilleur sera l'apprentissage. L'apprenant apprend donc par lui-même à classer ou à décider d'une action comme le fait le professeur. [9]

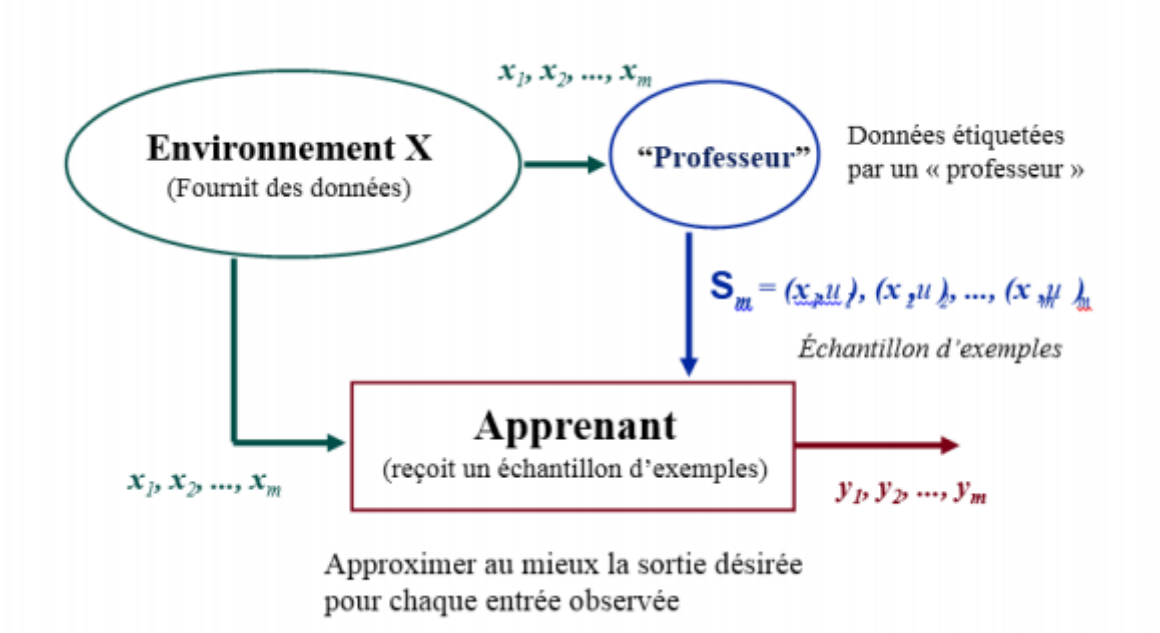

Figure 1.17. Schéma de l'apprentissage supervisé. [9]

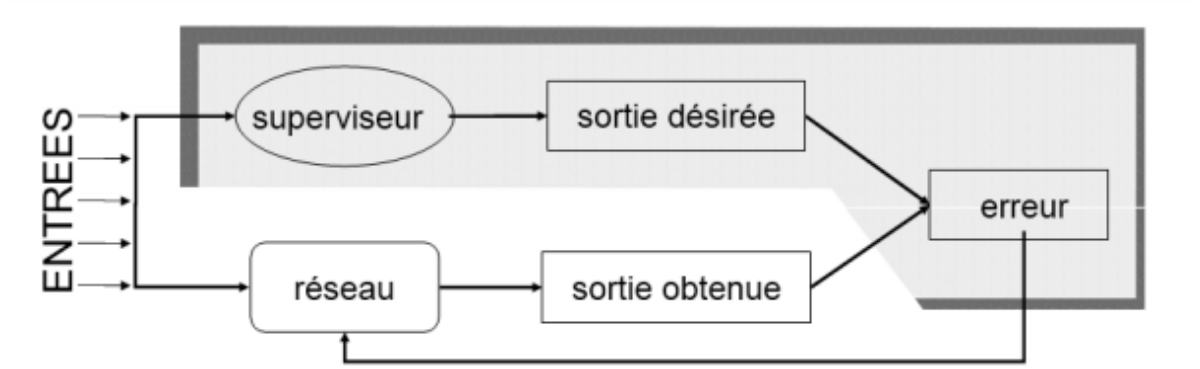

Figure 1.18. Schéma d'un modèle supervisé. [9]

- **Apprentissage Non-Supervisé**

Aucun expert n'est disponible. Il vise à concevoir un modèle structurant l'information. La différence ici est que les comportements (ou catégories ou encore les classes) des données d'apprentissage ne sont pas connus, c'est ce que l'on cherche à trouver. [9]

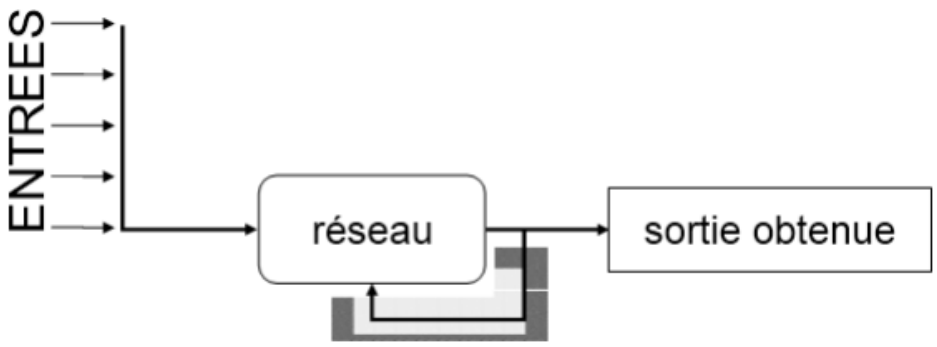

Figure 1.19. Schéma d'un modèle non supervisé. [9]

- **Apprentissage par Renforcement**

L'apprentissage par renforcement est un domaine de l'apprentissage automatique. Il s'agit de prendre des mesures appropriées pour maximiser la récompense dans une situation particulière. Il est utilisé par divers logiciels et machines pour trouver le meilleur comportement possible ou la voie à suivre dans une situation spécifique. L'apprentissage par renforcement diffère de l'apprentissage supervisé d'une manière qui, dans l'apprentissage supervisé, les données de formation ont le corrigé avec lui, de sorte que le modèle est formé avec la bonne réponse elle-même tandis que dans l'apprentissage par renforcement, il n'y a pas de réponse mais l'agent de renforcement décide quoi faire pour effectuer la tâche donnée. En l'absence d'un ensemble de données de formation, il est tenu de tirer des leçons de son expérience. [10]

1.10 Réseaux de neurones (RN)

Un réseau de neurones est une série d'algorithmes qui tentent de reconnaître les relations sous-jacentes dans un ensemble de données grâce à un processus qui imite le fonctionnement du cerveau humain. En ce sens, les réseaux de neurones se réfèrent à des systèmes de neurones, de nature organique ou artificielle. Les réseaux de neurones peuvent s'adapter à l'évolution des entrées ; afin que le réseau génère le meilleur résultat possible sans avoir à repenser les critères de sortie. Le concept de réseaux de neurones, qui a ses racines dans l'intelligence artificielle, gagne rapidement en popularité dans le développement de systèmes commerciaux. [11]

1.11 Apprentissage profond(Deep Learning)

Le Deep Learning est une technique d'apprentissage automatique qui apprend aux ordinateurs à faire ce qui vient naturellement aux humains: apprendre par l'exemple. Le Deep Learning est une technologie clé derrière les voitures sans conducteur, qui leur permet de reconnaître un panneau d'arrêt ou de distinguer un piéton d'un lampadaire. C'est la clé de la commande vocale dans les appareils grand public tels que les téléphones, les tablettes, les téléviseurs et les haut-parleurs mains libres. L'apprentissage en profondeur retient beaucoup l'attention ces derniers temps et pour une bonne raison. Il atteint des résultats qui n'étaient pas possibles auparavant.

Dans l'apprentissage en profondeur, un modèle informatique apprend à effectuer des tâches de classification directement à partir d'images, de texte ou de son. Les modèles d'apprentissage en profondeur peuvent atteindre une précision de pointe, dépassant parfois les performances au niveau humain. Les modèles sont formés à l'aide d'un grand ensemble de données étiquetées et d'architectures de réseaux de neurones qui contiennent de nombreuses couches. [12]

Pourquoi le Deep Learning est important ?

L'apprentissage en profondeur atteint une précision de reconnaissance à des niveaux plus élevés que jamais. Cela permet à l'électronique grand public de répondre aux attentes des utilisateurs, et elle est cruciale pour les applications critiques pour la sécurité comme les voitures sans conducteur. Les progrès récents de l'apprentissage en profondeur se sont améliorés au point où l'apprentissage en profondeur surpasse les humains dans certaines tâches comme la classification d'objets dans les images. [12]

Alors que le Deep Learning a été théorisé pour la première fois dans les années 80, il existe deux raisons principales pour lesquelles il n'est devenu utile que récemment :

- L'apprentissage en profondeur nécessite de grandes quantités de données étiquetées. Par exemple, le développement d'une voiture sans conducteur nécessite des millions d'images et des milliers d'heures de vidéo. [12]
- Le Deep Learning nécessite une puissance de calcul importante. Les GPU hautes performances ont une architecture parallèle efficace pour l'apprentissage en profondeur. Lorsqu'il est combiné avec des clusters ou le cloud computing, cela

permet aux équipes de développement de réduire le temps de formation d'un réseau d'apprentissage en profondeur de quelques semaines à quelques heures ou moins.

Il existe différents algorithmes de Deep Learning. Nous pouvons ainsi citer [12] :

- **Les réseaux de neurones récurrents (RNN ou Recurrent Neural Networks)**

Les humains ne commencent pas leur réflexion à partir de zéro chaque seconde. En lisant cet essai, vous comprenez chaque mot en fonction de votre compréhension des mots précédents. Vous ne jetez pas tout et recommencez à penser à partir de zéro. Vos pensées ont de la persévérance. [13]

Les réseaux de neurones traditionnels ne peuvent pas le faire, et cela semble être une lacune majeure. Par exemple, imaginez que vous souhaitiez classer le type d'événement qui se produit à chaque étape d'un film. On ne sait pas comment un réseau de neurones traditionnel pourrait utiliser son raisonnement sur des événements antérieurs dans le film pour les informer plus tard. [13]

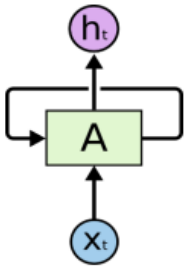

Figure 1.20. Les réseaux neuronaux récurrents ont des boucles. [13]

Les réseaux de neurones récurrents résolvent ce problème. Ce sont des réseaux avec des boucles qui permettent à l'information de persister. [13]

Dans la **figure 1.21.** ci-dessus, un segment de réseau neuronal; "A" regarde une entrée Xt et fournit une valeur ht. Une boucle permet de passer des informations d'une étape du réseau à l'autre. [13]

Ces boucles rendent les réseaux neuronaux récurrents un peu mystérieux. Cependant, si vous pensez un peu plus, il s'avère qu'ils ne sont pas tous différents d'un réseau de neurones normal. Un réseau de neurones récurrent peut être considéré comme des copies

multiples du même réseau, chacune transmettant un message à un successeur comme le montre la **figure 1.II.3.2.** Considérez ce qui se passe si nous déroulons la boucle [13] :

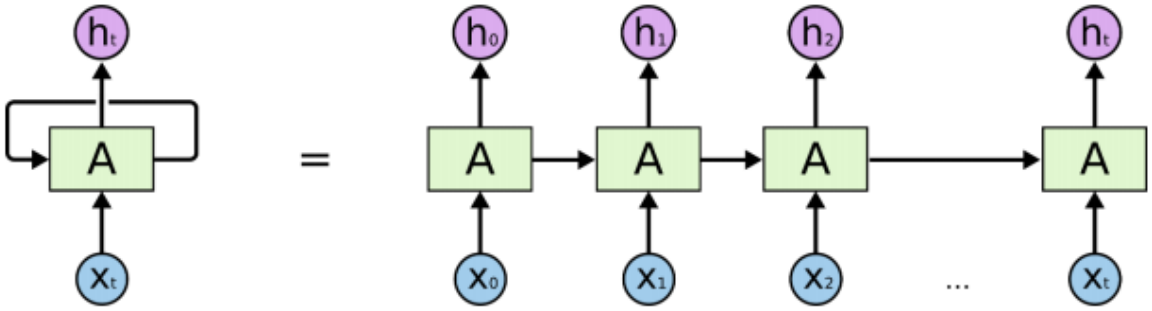

Figure 1.21. Un réseau neuronal récurrent déroulé. [13]

Cette nature en chaîne révèle que les réseaux neuronaux récurrents sont intimement liés aux séquences et aux listes. Ils sont l'architecture naturelle du réseau de neurones à utiliser pour de telles données. [13]

Au cours des dernières années, il y a eu un succès incroyable en appliquant les RNN à une variété de problèmes : la reconnaissance de la parole, la modélisation du langage, la traduction, le sous-titrage des images ...etc. [13]

- **Les réseaux de neurones convolutionnels (CNN ou Convolutional Neural Networks)**

Les réseaux de neurones convolutifs sont un type particulier de réseaux de neurones multicouches. Ils sont largement utilisés dans les problèmes de reconnaissance de formes. Comparé à d'autres techniques, les réseaux de neurones convolutifs présentent notamment l'avantage de la reconnaissance de modèles visuels avec un prétraitement minimal et une variabilité extrême des modèles. Il est à noter aussi l'architecture exceptionnelle que nous traiterons en détail dans la sous-section suivante. [14]

- **Architecture de réseaux de neurones convolutifs**

Une architecture CNN est formée de couches distinctes qui transforment le volume d'entrée en un volume de sortie. Nous présentons quelques types de ces couches qui sont le plus couramment utilisés [14] :

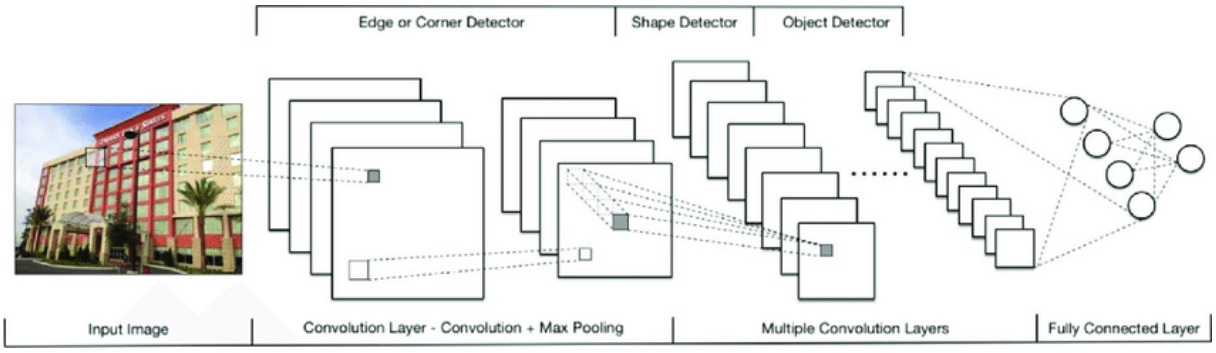

Figure 1.23. Une architecture de réseaux de neurones convolutifs. [14]

- **La couche convolutive**

La couche Convolutive applique des filtres sur ces entrés (exemple une image couleur), Chaque filtre est petit en termes de dimensions de hauteur, de largeur et notamment la profondeur de ce filtre qui doit être la même que la profondeur de l'entrée (les images couleurs ayant de profondeur de 3). Le filtre se positionne initialement au coin supérieur gauche sur la matrice d'image. Il exécute une multiplication entre ces valeurs et celle de la section sélectionnée de la matrice ce qui génère une nouvelle valeur. Ce processus se répète sur chaque nouvelle position (le déplacement est d'un pas vers la droite). La collection de ces valeurs générées construit une matrice de résultats appelée la carte de caractéristiques (feature map). Avant que l'étape de convolution soit effectuée, nous devons définir trois paramètres qui contrôlent et affectent directement la taille de la carte de caractéristiques (feature map). Ces paramètres sont les suivants :

- Premier paramètre : Profondeur de la couche (Depth) :

La profondeur de carte de caractéristiques (feature map) est identique au nombre de filtres que nous utilisons pour l'opération de convolution. La convolution effectuée sur l'image à l'aide de trois filtres distincts produit trois cartes de caractéristiques (feature map) différentes

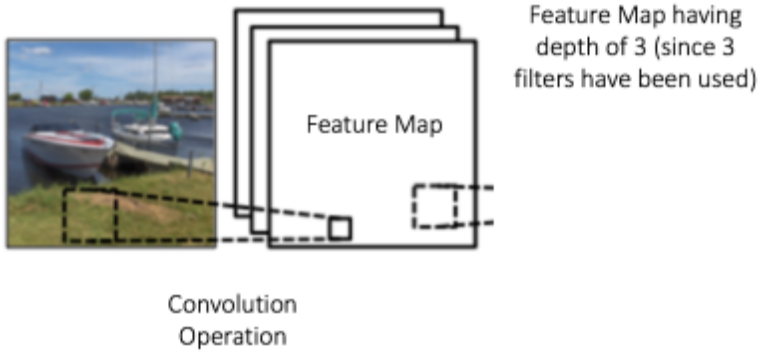

Figure 1.24. Le Profondeur de carte de caractéristiques (feature map).

- Deuxième paramètre : Le pas (Stride)

Le pas (stride) est le nombre de pixels par lequel nous glissons notre filtre sur la matrice d'image. Le pas (stride) est 1 par défaut. Notons que l'utilisation d'un pas plus grand produira des cartes de caractéristiques (feature map) plus petites.

- Troisième paramètre : La marge à zéro

Le remplissage (padding) est une marge de zéro qui est placée autour de l'image. Dans de nombreux cas, le filtre ne se glisse pas parfaitement sur la matrice d'image. Cela nous pousse à choisir entre deux options :

- ✓ remplir (padding) l'image avec des zéros (zéro-padding)
- ✓ conserver uniquement une partie valide de l'image en supprimant la partie où le filtre ne la couvre pas.

Notons que l'utilisation du remplissage permet de résoudre un autre problème pertinent puisque si nous continuons à appliquer des couches convolutives, la taille du volume diminuera plus rapidement.

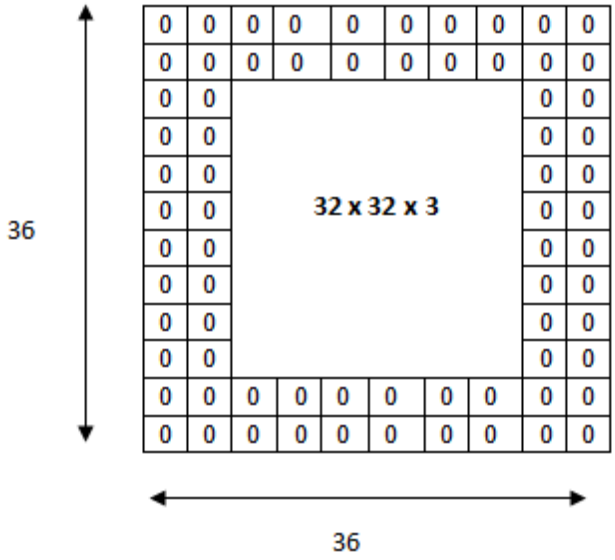

Figure 1.25. Remplissage de l'image avec des zéros.

- **La couche de pooling**

La couche de pooling prend un filtre (normalement de taille 2x2) et un pas de même longueur. Il s'applique ensuite au volume d'entrée et génère le nombre maximal dans chaque région autour de laquelle le filtre se convole. L'idée derrière l'utilisation de pooling dans les réseaux de neurones convolutifs est que l'emplacement exact d'une entité n'est pas aussi important que son emplacement relatif par rapport aux autres entités. Cette couche réduit, également, considérablement la dimension spatiale (la longueur et la largeur mais pas la profondeur) du volume d'entrée.

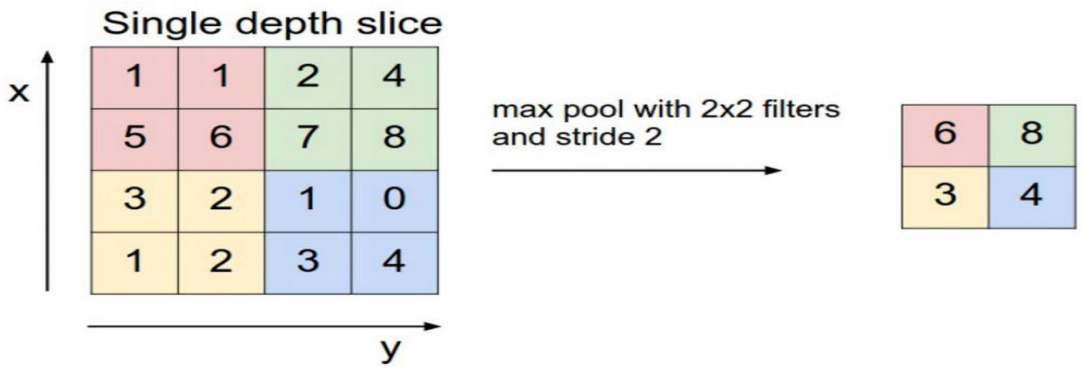

Figure 1.26. La fonction Max pooling.

- **Les couches de correction**

Le ReLu (Rectified Linear Unit) est l'une des fonctions d'activation les plus utilisée dans l'apprentissage en profondeur. Il introduit la non-linéarité dans le réseau convolutif et il est généralement appliqué après la couche convolutive. La couche ReLu modifie toutes les valeurs négatives des entrées à zéro en appliquant la fonction f(x) à toutes les valeurs du volume d'entrée

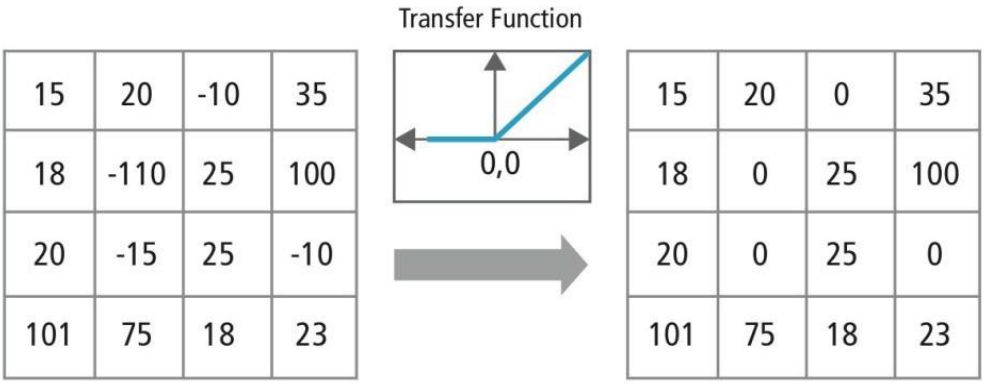

Figure 1.27. La fonction ReLu.

- **Les couches entièrement connectées**

Après plusieurs couches de pooling et de convolution, l'apprentissage des caractéristiques sur le réseau est terminé. Notre réseau passe maintenant à la classification via des couches entièrement connectées (fully connected layers). Chaque neurone d'une couche se connectant à l'autre, ce qui signifie que cette couche est connectée à toutes les activations précédentes. Cette couche prend un volume d'entrée et génère un vecteur à N dimensions où N est le nombre de classes connues par le réseau.

➢ **La machine de Boltzmann profonde (DBN ou Deep Belief Network)**

DBN est un réseau de neurones constitué de plus d'une machine Boltzmann restreinte (RBM). RBM et DBN ont été conçus par Geoffrey Hinton. [14]

RBM est un algorithme utile pour la réduction des dimensions, la régression, le filtrage collaboratif, classification, apprentissage de fonctionnalités et modélisation de thèmes.

Il C'est un réseau de neurones à deux couches peu profond composé d'une couche d'entrée visible et une couche de sortie cachée. Ils sont les blocs de construction de DBN. Pour chaque RBM la couche visible représente les inputs et la couche cachée reçoit les

inputs de la couche visible, fait les calculs ensuite elle envoie les résultats à la couche visible. Dans la DBN chaque couche cachée du RBM représente la couche visible de la RBM suivante comme le montre la **figure 1.28**. []

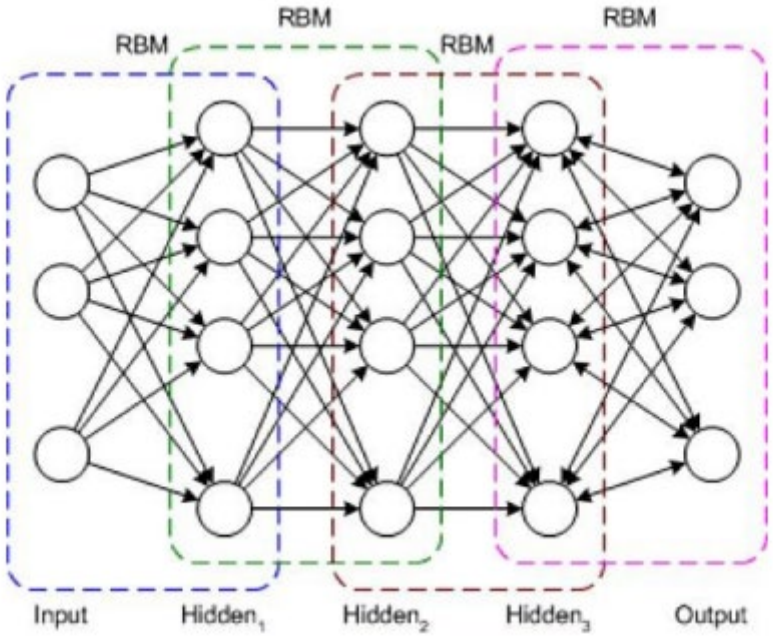

Figure 1.28. DBN.

Les DBN ont été utilisés pour générer et reconnaître des images (reconnaissance faciale), des séquences vidéo et des données de capture de mouvement. La NASA utilise DBN pour classer des téraoctets d'imagerie satellitaire à haute résolution et très diversifiée.

- **Domaines d'applications de Deep Learning**

Des applications du Deep Learning sont utilisées dans divers secteurs, de la conduite automatisée aux dispositifs médicaux.

- Conduite automatisée
- Aérospatiale et défense
- Recherche médicale
- Automatisation industrielle
- Électronique
- Assistants virtuels
- La reconnaissance faciale
- Traitement du langage naturel

- Génération de texte
- Divertissement télévisé
- Agrégation de nouvelles
- Reconnaissance visuelle
- Détection de fraude
- Ect……

Conclusion

L'objectif principal de ce chapitre était, dans un premier lieu, la présentation des généralités sur la compression fractale d'image et le Deep Learning.

Nous avons constaté qu'il y a un lien, même fort, entre les deux axes de recherche que les chercheurs ont déjà commencé à creuser là-dessus

Dans le chapitre suivant, nous allons voir, les différentes contributions assez récentes dans le domaine de la compression d'image optimisée par le Deep Learning. Ensuite, nous détaillons notre méthode de compression fractale profonde.

Chapitre 2

Système de Compression profonde proposé

" Les scientifiques sont des gens affligés

par le besoin maladif de tout expliquer."

Grégoire LACROIX

Introduction

L'amélioration de la compression fractale d'image est récemment devenue le centre d'attention des chercheurs dans ce domaine, et comme nous l'avons dit précédemment, les chercheurs étaient intéressés à améliorer deux choses, à savoir, la classification des sous-blocs et la recherche de quartie.

Dans ce chapitre, nous illustrerons certains des travaux importants réalisés en compression , puis nous proposerons notre approche basée sur l'apprentissage profond

2.1 Les travaux connexes

En 2016, George T. et al. [15], ont proposé un cadre général pour la compression d'images à débit variable et une nouvelle architecture basée sur des réseaux récurrents convolutionnels et déconvolutionnels.

Leurs modèles répondent aux principaux problèmes qui ont empêché les réseaux de neurones à codage automatique de concurrencer les algorithmes de compression d'image existants:

- les réseaux n'ont besoin d'être formés qu'une seule fois (pas par image), quelles que soient les dimensions de l'image d'entrée et le taux de compression souhaité;
- les réseaux sont progressifs, ce qui signifie que plus il y a de bits envoyés, plus la reconstruction d'image est précise;
- l'architecture proposée est au moins aussi efficace qu'un auto-encodeur standard formé à cet effet pour un nombre donné de bits.

Sur une référence à grande échelle de miniatures 32 * 32, cette approche offre une meilleure qualité visuelle que (sans en-tête) JPEG et JPEG2000, avec une taille de stockage réduite de 10% ou plus.

En 2017, Shibani S et al. [16] décrivent le concept de compression générative, la compression des données à l'aide des modèles génératifs, et suggèrent qu'il est utile de poursuivre pour produire des reconstructions plus précises et visuellement agréables à des niveaux de compression beaucoup plus profonds pour les données d'image et vidéo. Ils démontrent également que la compression générative est de plusieurs ordres de grandeur plus résistante aux taux d'erreur sur les bits que les schémas de codage de longueur variable traditionnels.

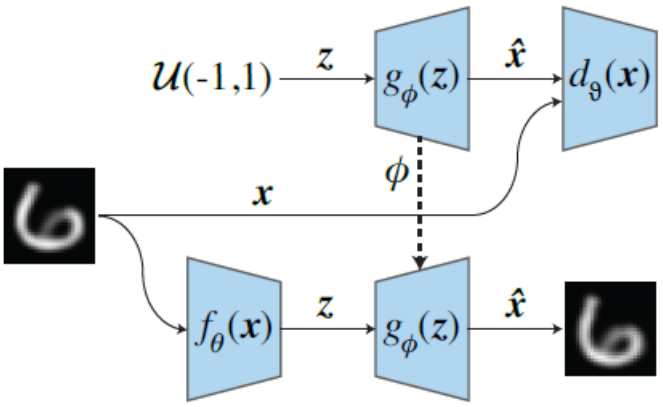

Figure 2.1 : Compression générative pour les données image [16]

En 2018, Zhengxue C. et al [17] ont présenté une architecture de compression d'image avec perte, qui utilise les avantages de l'auto-encodeur convolutif (CAE) pour obtenir une efficacité du codage. Premièrement, ils conçoivent une nouvelle architecture CAE pour remplacer les transformations conventionnelles et former ce CAE en utilisant une fonction de perte de taux de distorsion. Ensuite, pour générer un plus compact d'énergie représentation, ils utilisent l'analyse en composantes principales (ACP) pour faire pivoter les cartes de caractéristiques produites par CAE, puis appliquez le codeur de quantification et d'entropie afin de générer les codes. Les résultats expérimentaux démontrent que cette approche surpasse les algorithmes traditionnels, en réduisant de 13,7% du taux BD sur la base de données Kodak images par rapport à JPEG2000. De plus, cette méthode maintient une complexité modérée similaire à JPEG2000.

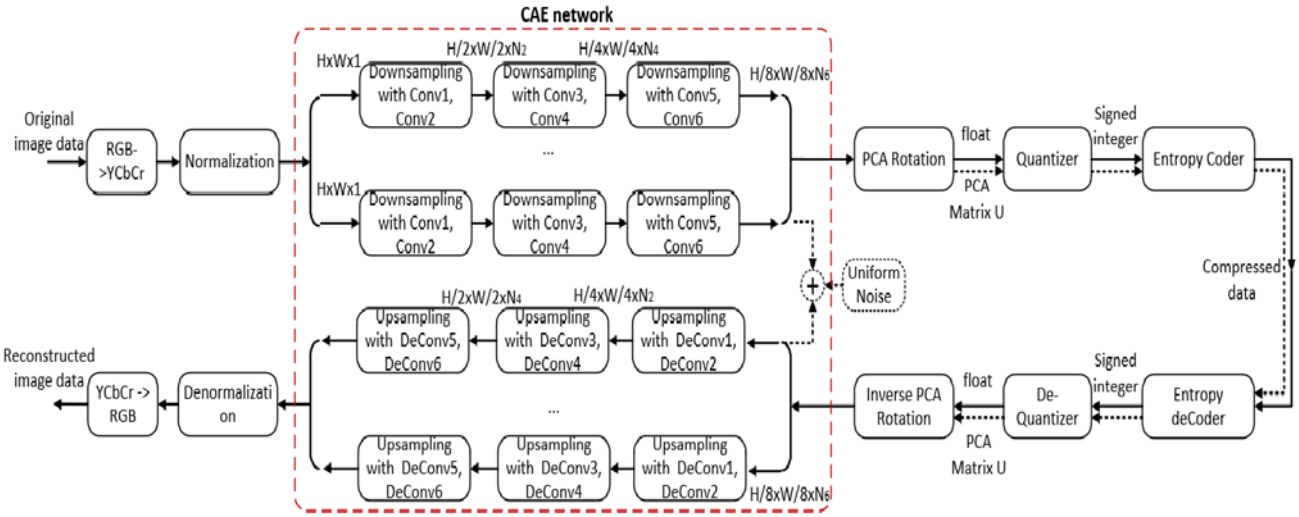

Figure 2.2 : Architecture du système proposé par Zhengxue C. et al. [17]

Dans la même année, Mu Li et al. [4], motivé par le fait que le contenu d'informations

locales est spatialement variant dans une image, ont proposé d'admettre que :

le débit binaire des différentes parties de l'image est adapté au contenu local, et le débit binaire sensible au contenu est attribué suivant un contenu pondéré du vecteur d'importance. La somme du vecteur d'importance peut ainsi servir d'alternative continue à l' estimation de l'entropie discrète pour contrôler le taux de compression. Le binariseur (la fonction est introduite pour approximer le fonctionnement binaire en rétropropagation pour le rendre différenciable). L'encodeur, décodeur, binariseur et vecteur d'importance peuvent être conjointement optimisé de bout en bout. Et un codeur entropique convolutionnel est présenté pour la compression sans perte du vecteur d'importance et des codes binaires. Dans la compression d'image à faible débit binaire, les expériences montrent que ce système surpasse considérablement JPEG et JPEG 2000 par l'index de similarité structurelle (SSIM), et peut produire un résultat visuel bien meilleur avec des bords nets, des textures riches et moins d'artefacts.

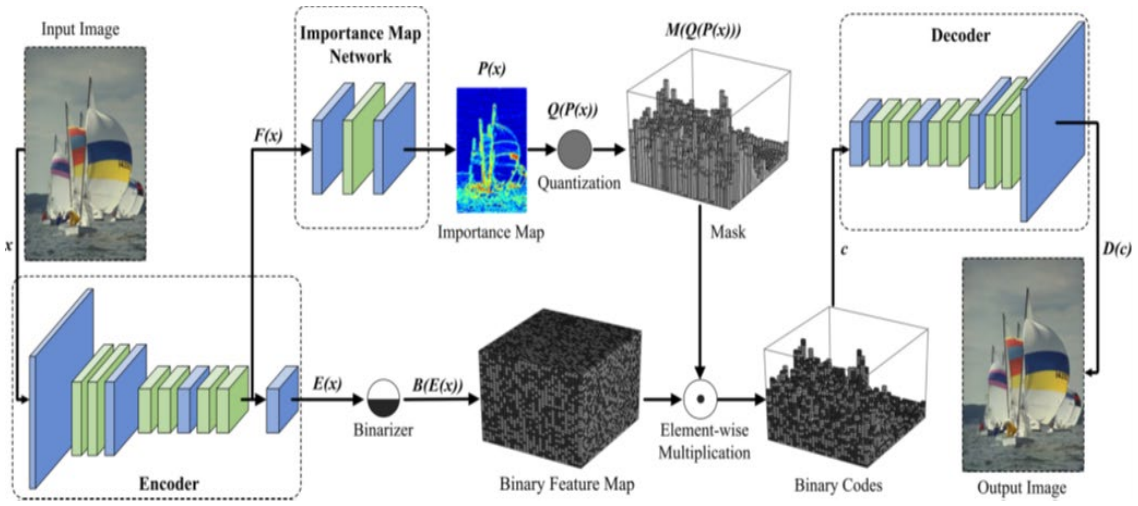

Figure 2.3 : Système proposé par Mu Li et al. [18].

Toujours en 2018, Eirikur A. et al. [19], ont proposé un cadre pour la compression d'image extrêmement entrainée basé sur les réseaux génératifs (GAN), obtenant des images visuellement agréables à des débits binaires nettement inférieurs à ceux des méthodes précédentes. Ceci est rendu possible grâce la formulation GAN de compression entrainée combinée à un générateur / décodeur qui fonctionne sur l'image en pleine résolution et est formé en combinaison avec un discriminateur multi-échelle.

De plus, si une carte d'étiquette sémantique de l'image d'origine est disponible, cette méthode peut entièrement synthétiser des régions sans importance dans l'image décodée

telles que les rues et les arbres à partir de la carte d'étiquette, ne nécessitant donc que le stockage de la région préservée et de la carte d'étiquette sémantique. Une étude utilisateur confirme que pour les faibles débits, notre approche est préférée aux méthodes de pointe, même lorsqu'elles utilisent plus du double des bits.

On trouve aussi, Haojie Liu et al. [19], ont proposé une méthode de compression d'image avec perte basée sur des réseaux de neurones convolutionnels profonds (CNN), qui surpasse les méthodes de JPEG2000 et JPEG, en calculant la similarité structurelle multi-échelles (MS-SSIM), au même débit binaire. Pour obtenir une meilleure optimisation de la distorsion du taux (RDO), ils introduisent également un apprentissage de transfert facile à difficile lors de l'ajout d'une erreur de quantification et une contrainte de taux. Enfin, ils évaluent leur méthode sur Kodak public et la base de données de test P / M publié par le Computer Vision Lab de l'ETH Zurich, résultant en moyenne 7,81% et 19,1% de réduction du taux de BD.

En 2019, Wenjing L. et al. [20], ont proposé :

Premièrement, selon la classification précise et rapide des images du réseau neuronal convolutionnel profond et les avantages de recherche rapide et de codage correspondant de la programmation de l'expression génique, il réalise théoriquement le mécanisme d'action du codage hybride de compression d'image fractale en combinant le réseau neuronal convolutionnel et l'expression génique programmation;

Ensuite, il utilise le réseau neuronal convolutionnel profond pour former et classer l'image, et utilise la méthode de segmentation adaptative à quatre arbres pour segmenter l'image classée, générant ainsi le bloc de domaine et le bloc de plage ensemble de classification. Selon le mécanisme d'action de la programmation de l'expression génique dans le codage par compression d'images fractales, il obtient rapidement la solution optimale de codage par compression d'images fractales en recherchant et en codant les sous-blocs du jeu de classification des blocs de plage et du jeu de classification correspondant au domaine.

Enfin, dans l'environnement CPU / GPU, il mène l'expérience comparative avec l'algorithme de compression d'image fractale de base et l'algorithme de compression d'image fractale basé sur un réseau neuronal convolutionnel. Les résultats expérimentaux montrent que cet algorithme proposé surpasse les algorithmes similaires en termes de vitesse et de précision de segmentation d'image ainsi que de vitesse de codage de compression fractale et de taux de compression. Par conséquent, cet algorithme est un algorithme de compression d'image fractale avec segmentation intelligente, encodage

rapide et taux de compression élevé.

2.2 Méthode proposée

Au cours des dernières années, l'apprentissage en profondeur est utilisé pour résoudre de nombreux problèmes dans différents domaines, dans notre approche, nous essaierons de déployer un apprentissage en profondeur pour extraire les caractéristiques des blocs afin de réduire les dimensions, puis nous utilisons k-means clustering pour minimiser le nombre de comparaison entre les blocs de destination et blocs de domaine

Notre approche consiste à partitionner l'image en blocs de domaine et blocs de destination à l'aide de l'ancienne méthode, puis à appliquer des transformations affines sur les blocs de domaine, puis à extraire les caractéristiques des blocs de domaine et de destination à l'aide de l'apprentissage en profondeur, exactement avec l'Auto-encoder pré-entrainer, puis nous utilisons k -means méthode de clustering pour regrouper les caractéristiques extraites, après tout, nous comparons simplement les blocs de domaines et les blocs de destination qui apparaissent dans le même cluster, ce qui aide à réduire considérablement le nombre de comparaison et à accélérer l'opération de compression.

Dans notre méthode ont a utilisé l'algorithme générique de ce type de compression.

Voici les algorithmes génériques de compression et décompression fractale :

Algorithme 1 Algorithme générique de la compression fractale

```
1.  Fonction Compression
2.      Créer un pavage de figures Sources
3.      Créer un pavage de figures destinations
4.      Pour toutes les figures destinations Faire
5.          Pour toutes les figures Sources Faire
6.              Pour toutes les transformations définies Faire
7.                  Appliquer la transformation à la figure Source
8.                  Ajuster la moyenne des couleurs des pixels
9.                  Appliquer la réduction de la figure Source vers la figure destination
10.                 Calculer l'erreur entre le résultat et la figure de destination
11.                 Si l'erreur est minimale pour la figure de destination Alors
12.                     Sauver les modifications effectuées
13.                 FinSi
```

14. **Fin Pour**
15. Ecrire dans le fichier de sortie les valeurs sauvées
16. **Fin Pour**
17. **Fin Pour**
18. **Fin Fonction**

Algorithme générique de la décompression fractale

Algorithme 2 Algorithme générique de la décompression fractale

1. **Fonction** Décompression
2. Créer une image
3. Créer un pavage de figures Sources
4. Créer un pavage de figures destinations
5. **Pour** toutes les itérations désirées **Faire**
6. **Pour** toutes les figures destinations **Faire**
7. Lire dans le fichier les transformations à appliquer
8. Lire le numéro de la figure Source à manipuler
9. Appliquer les transformations à la figure précédemment évoquée
10. Appliquer la réduction de la figure Source vers la figure destination
11. Insérer le résultat dans l'image
12. **Fin Pour**
13. **Fin Pour**
14. **Fin Fonction**

2.2.1 Algorithme proposé :

Algorithme proposé : algorithme de compression basé deep

Fonction Compression_deep

Créer un pavage de figures Sources

Créer un pavage de figures destinations

Pour toutes les figures Sources **Faire**

Pour toutes les transformations définies **Faire**

Appliquer la transformation à la figure Source

Ajuster la moyenne des couleurs des pixels

Appliquer la réduction de la figure Source vers la figure destination

Extraire les caractéristiques de la figure transformée (auto-encoder)

Sauver les vecteurs de caractéristique

Fin Pour

Fin Pour

Regrouper les vecteurs de caractéristique(Domaine) avec K-means(Clustering)

Pour toutes les figures destinations **Faire**

Extraire les caractéristiques de la figure destination (auto-encoder)

Sauver les vecteurs de caractéristique

Fin Pour

Regrouper les vecteurs de caractéristique (destination) avec K means(Clustering)

Pour tous les clusters **Faire**

Calculer l'erreur entre la figure Source et destination (dans le mêmecluster)

Si l'erreur est minimale pour la figure de destination **Alors**

Sauver les modifications effectuées

FinSi

Fin Pour

Ecrire dans le fichier de sortie les valeurs sauvées

Fin Fonction

2.2.2 Architecture du système proposé

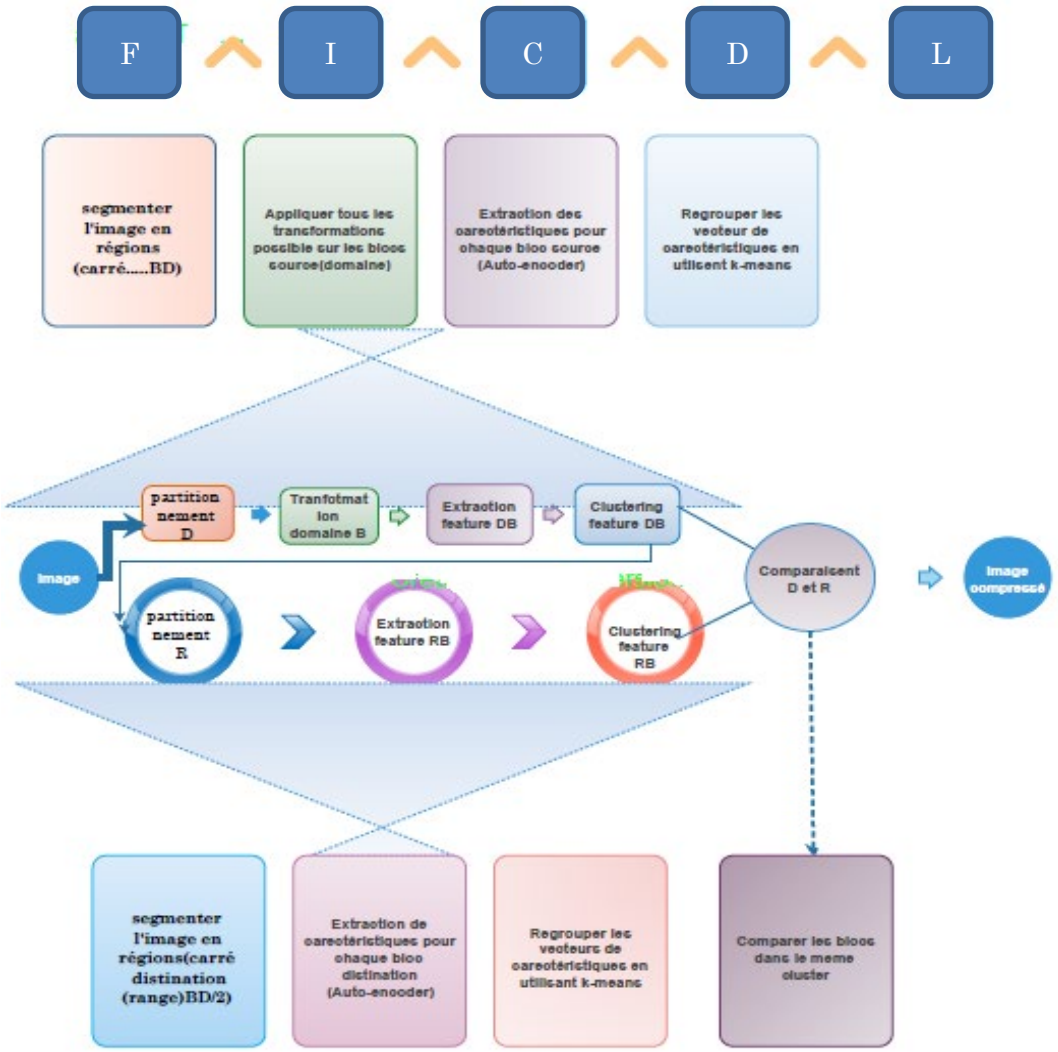

Figure 2.4 : Architecture de notre système

- **Partitionner les domaines blocs et les destinations blocs**

La taille des blocs est des puissances de deux car cela facilite les choses. Les blocs source sont 8 par 8 tandis que les blocs de destination sont 4 par 4.

- **Transformation des domaines blocs**

Cette étape consiste à appliquer tous les transformations possibles aux blocs domaines.

- **Extraction des caractéristiques**

Après avoir transformées les domaines blocs, nous les avons entrées comme un jeu de

teste dans notre auto-encoder (pré-trained sur la data set fashion mnist). Pour extraire les vecteurs de caractéristiques de chaque bloc.

- **Clustering**

Afin d'extraire les vecteurs de caractéristiques des blocs de domaine, nous les passons à l'algorithme k-means pour les regroupées. (Dans notre méthode on a choisi de regrouper à 10 groupes après un ensemble de testes).

- ***Les mêmes étapes sont appliquées sur les blocs destinations sauf transformation***.

- **Comparaison**

Dans cette étape nous comparons simplement les blocs de domaines et les blocs de destination qui apparaissent dans le même cluster.

2.3 Conclusion

Dans ce chapitre nous avons cité les travaux récents et les différentes méthodes proposées dans le domaine de l'amélioration de la compression fractale d'image.

Ainsi nous avons présentées notre méthode de compression basée sur les réseaux de neurones profonds.

Nous allons voir dans le chapitre suivant, nos résultats obtenus, ainsi les différents outils logique et physique utilisées dans notre travaille.

Chapitre 3 : Résultats et Discussion

« Celui qui dit que deux et deux font quatre, a-t-il une connaissance de plus que celui qui se contenterait de dire que deux et deux font deux et deux ? ».

Jean le Rond d'Alembert

Introduction

Dans le chapitre 2, nous avons détaillé notre approche proposée, qui consiste à utiliser le DeepLearning pour regroupées les blocs sources et destinations.

Dans ce chapitre, nous allons nous intéresser aux résultats expérimentaux obtenus.

1. Présentation des outils de développement

Cette section est consacrée à la description des outils de développement employés dans la conception et l'implémentation de système proposé.

- **Python**

Python est un langage de programmation interprète de haut niveau et orienté objet. La syntaxe simple et facile à apprendre de Python met l'accent sur la lisibilité et réduit donc le coût de la maintenance du programme. Python prend en charge les modules et les packages, raison pour laquelle python est utilisé par une vaste communauté de développeurs et de programmeurs.

- **Keras**

Keras est une application de réseaux de neurones de haut niveau, écrite en Python et capable de s'exécuter sur TensorFlow. Keras a été développé pour permettre une expérimentation rapide. Il est utilisé par les chercheurs pour pouvoir faire des recherches de qualité et passer de l'idée au résultat le plus rapidement possible. Les keras sont utilisés pour construire un prototypage

rapide et prendre en charge les réseaux convolutifs et les réseaux récurrents et s'exécute sur CPU et GPU.

- **Google Colaboratory**

Google Colaboratory est une application en nuage basée sur l'environnement Jupyter, qui ne nécessite aucune installation. L'application Google Colaboratory permet aux utilisateurs d'écrire, d'exécuter, de sauvegarder et de partager leurs codes avec un accès à de puissantes ressources de calcul gratuitement et à partir du navigateur et prend en charge python et bien d'autres langues.

2. Algorithme et Implémentation

- **Architecture auto-encoder**

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 28, 28, 14)	140
max_pooling2d_1 (MaxPooling2	(None, 14, 14, 14)	0
dropout_1 (Dropout)	(None, 14, 14, 14)	0
conv2d_2 (Conv2D)	(None, 14, 14, 7)	889
max_pooling2d_2 (MaxPooling2	(None, 7, 7, 7)	0
dropout_2 (Dropout)	(None, 7, 7, 7)	0
conv2d_3 (Conv2D)	(None, 7, 7, 7)	448
up_sampling2d_1 (UpSampling2	(None, 14, 14, 7)	0
dropout_3 (Dropout)	(None, 14, 14, 7)	0
conv2d_4 (Conv2D)	(None, 14, 14, 14)	896
up_sampling2d_2 (UpSampling2	(None, 28, 28, 14)	0
dropout_4 (Dropout)	(None, 28, 28, 14)	0
conv2d_5 (Conv2D)	(None, 28, 28, 1)	127

Figure 3.1. Architecture auto-encoder

L'algorithme a été écrit en python avec l'utilisation de la bibliothèque KERAS et exécuté sur la plateforme Google Colaboratory (voir figure 3.1)

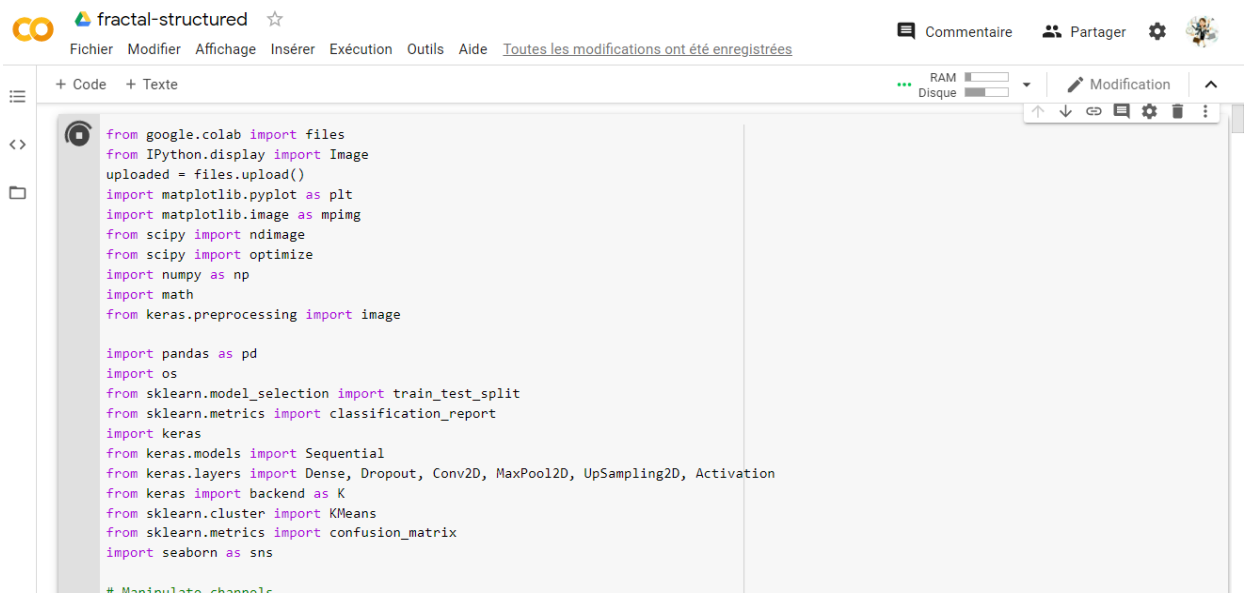

Figure 3.2. Partie du code dans Google Colaboratory

3. Résultats obtenus et discussion

➢ L'algorithme générique

Image	Nombre de comparaisons	Temps (s)	**PSNR**
Monkey	131072	16s	7
lena	131072	7s	16
pep	131072	8s	12
caman	131072	8s	31

Table 3.1. Les résultats de l'algorithme générique

➢ Notre algorithme

Image	Nombre de comparaisons	Temps (s)	**PSNR**
Monkey	13912	3s	10
lena	13387	4s	17
pep	13842	4s	18
caman	15153	4s	**19**

Table 3.2. Les résultats de notre algorithme

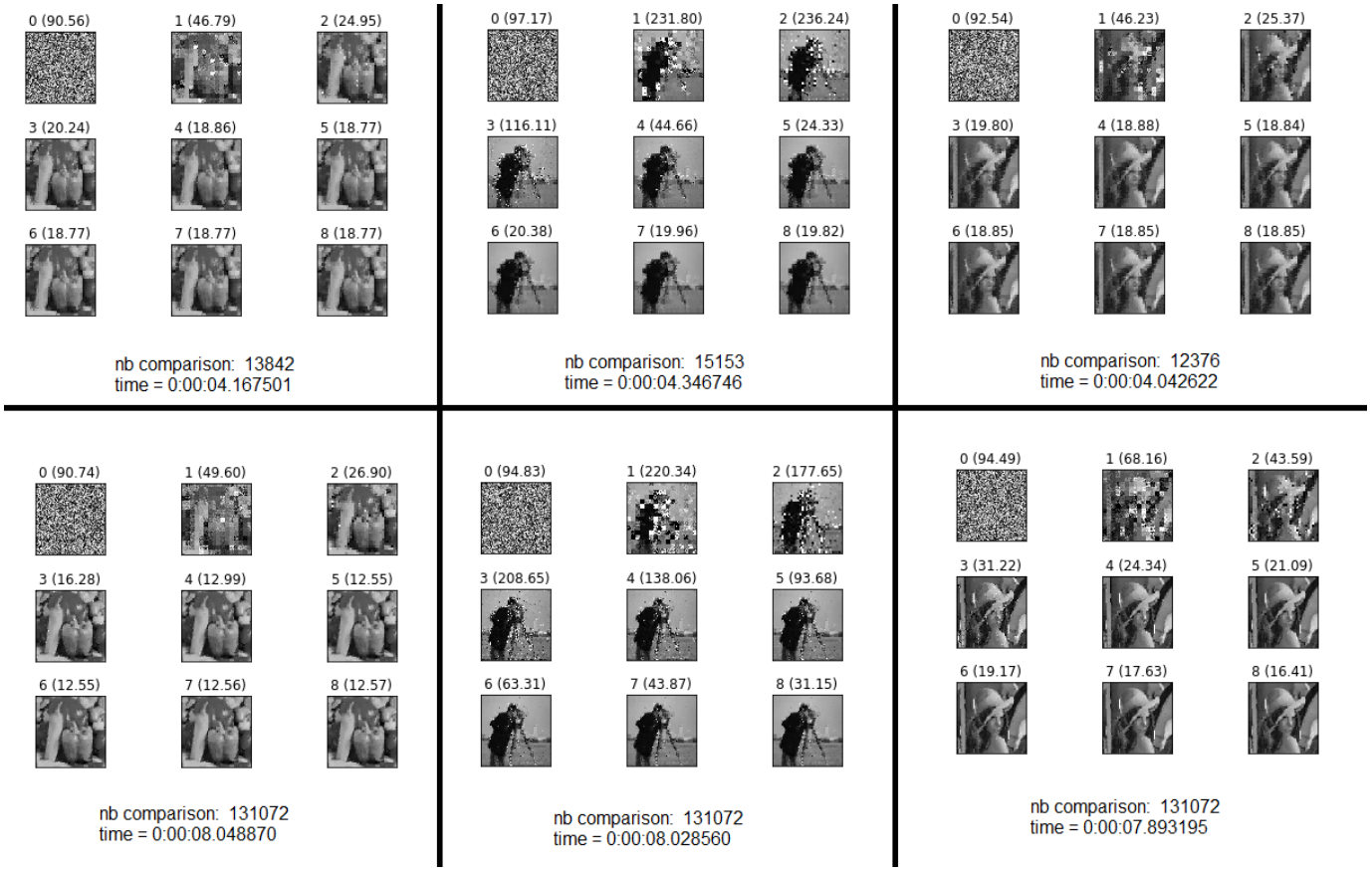

Figure 3.3. Quelques résultats

Conclusion générale

« Ayez le courage de suivre votre cœur et votre intuition. L'un et l'autre savent ce que vous voulez réellement devenir. Le reste est secondaire ».

Steve Jobs

La quantité d'information augmente plus vite que la capacité de stockage. On a besoin donc de compresser les données avant de les utilisées (Acquisition des données scientifiques, médicales, industrielles, téléphonie, conservation du patrimoine, etc.). Donc, le domaine a une longue vie devant soi. Des nouveaux importants algorithmes naissent tous les ans.

Toute compression cherche à éliminer la redondance, soit par une structuration différente, mais réversible, permettant de restaurer l'original (compression sans pertes), soit en supprimant une partie d'information considérée inutile, ou sans importance (méthodes avec pertes, irréversibles). Les méthodes irréversibles offrent un taux de compression beaucoup plus élevé que méthodes sans pertes ! Bien sûr, parfois il est hors de question de perdre l'information. Ceci dit, les critères d'évaluation des techniques de compression sont très variés.

La compression fractale est une méthode de compression irréversible, elle repose sur un principe simple : toute image comporte des similarités et des redondances. Cette méthode de compression consiste donc à détecter la récurrence des motifs à différentes échelles, et tend à éliminer la redondance d'informations dans l'image.

Cette méthode est largement utilisée ces dernières années en vue de sa fidélité approximative à l'image originale, néanmoins elle est beaucoup gourmande en temps de calcul.

D'autre part, les algorithmes d'apprentissage profond est une famille d'algorithmes Machine Learning qui tendent à trouver indépendamment des solutions aux problèmes en

reconnaissant les modèles dans les bases de données. En d'autres termes : le Machine Learning permet aux systèmes informatiques de reconnaître des modèles sur la base d'algorithmes et d'ensembles de données existants et de développer des concepts de solutions adéquats, autre choses, ces algorithmes peuvent s'appliquer généralement à tout type de problème (ou presque).

Tous ces arguments nous ont donné le courage de s'enfoncer un peu dans ce domaine et de proposer un système de compression fractale basé sur un auto-encodeur afin de réduire le temps de compression et conserver au maximum la qualité de l'image reconstruite.

Notre travail nous a permis d'intégrer le concept de Deep Learning dans l'objectif d'optimiser la compression fractale d'images, en effet d'après les résultats obtenus nous constatons une amélioration considérable dans le temps de calcul tout en gardant un aspect qualité assez respectueux.

Les résultats obtenus sont assez encourageants et peuvent faire l'objet d'un papier après validation sur un benchmark d'images.