ENSEEIHT
SII

Parcours Sciences du numérique
Rapport

Optimisation du Stockage d'Images pour le Machine Learning

Élèves :
Ayoub NAJMEDDINE
Hamza ZOUGARI BELKHAYAT

Encadrants :
Martial COULON
Kaveena PERSAND

6 juin 2022

Table des matières

Table des figures

1 Introduction

L'utilisation d'algorithmes de compression dans les tâches de machine learning telles que le regroupement et la classification est apparue dans divers domaines, parfois avec la promesse de réduire les problèmes de sélection explicite des caractéristiques. Selon un autre point de vue, les algorithmes de compression transforment implicitement les chaînes de caractères en vecteurs d'espaces de caractéristiques implicites, et les mesures de similarité basées sur la compression calculent la similarité dans ces espaces de caractéristiques. Pour souligner ce point, nous trouvons des connexions théoriques et empiriques entre les modèles vectoriels de machine learning traditionnels et la compression, ce qui encourage la fertilisation croisée dans les travaux futurs.

2 Compression d'images

2.1 Première définition (bon à savoir)

En traitement du signal, **la compression des données** (ou aussi codage source ou réduction de débit binaire) est le processus qui consiste à **coder des informations en utilisant moins de bits que la représentation originale**. La compression *sans perte* réduit le nombre de bits en identifiant et en éliminant les redondances statistiques. Aucune information n'est perdue dans la compression sans perte. La compression *avec perte* réduit le nombre de bits en supprimant les informations inutiles ou moins importantes. En général, un dispositif qui effectue la compression des données est appelé **encodeur**, et celui qui effectue l'inverse du processus (décompression) est appelé **décodeur**.

Ainsi, **la compression d'images** est un type de compression de données appliqué aux images numériques, afin de réduire leur coût de stockage ou de transmission. Les algorithmes peuvent tirer parti de la perception visuelle et des propriétés statistiques des données d'image pour fournir des résultats supérieurs à ceux des méthodes de compression de données génériques utilisées pour d'autres données numériques.

2.2 Notions de base

1. **Taux de compression :** mesure de la performance d'un algorithme de compression de données informatiques, il correspond au rapport de la taille du fichier compressé sur la taille du fichier initial. A noter que le choix d'un très haut taux de compression entraîne inévitablement une perte de qualité du fichier original.
2. **Qualité de la méthode de compression :** elle est souvent mesurée par le rapport signal/bruit de crête. Ce dernier calcule la quantité de bruit introduite par une compression avec perte de l'image, mais le jugement subjectif du spectateur est également considéré comme une mesure importante, voire la plus importante.
3. **Complexité de l'algorithme de compression :** peut référer au coût du calcul ou la mémoire requise pour les opérations effectuées par l'algorithme adopté.

2.3 Pourquoi les techniques de compression d'images sont-elles nécessaires ?

Voici quelques-unes des applications potentielles de la compression d'images :

- **Les données occupent moins de place lorsqu'elles sont comprimées**. Par conséquent, cela permet de stocker plus de données avec moins d'espace disque. Ceci est particulièrement utile dans le domaine des soins de santé, où les images médicales doivent être archivées, et où le volume des ensembles de données est massif.
- **Certaines techniques de compression d'images consistent à extraire les composantes les plus utiles de l'image** (ACP), qui peuvent être utilisées pour la synthèse ou l'extraction de caractéristiques et l'analyse de données.
- **Le gouvernement fédéral ou les agences de sécurité ont besoin de maintenir des enregistrements de personnes d'une manière prédéterminée, standard et uniforme**. Ainsi, les images provenant de toutes les sources doivent être compressées pour avoir une forme, une taille et une résolution uniformes.

3 Une approche de compression d'images grâce aux méthodes de machine learning

Une caractéristique commune à la plupart des images est que les pixels voisins sont fortement corrélés et contiennent donc des informations très redondantes. L'objectif de base de la compression d'image est de trouver une représentation d'image dans laquelle les pixels sont moins corrélés. Récemment, au lieu d'effectuer une transformation de fréquence, une approche basée sur le machine learning a été proposée pour la compression d'images. Du point de vue du machine learning, **deux problèmes fondamentaux se posent**. Premièrement, comment *sélectionner* les pixels les plus représentatifs, ce qui est essentiellement un problème d'apprentissage actif. Les pixels sélectionnés, ainsi que l'image en échelle de gris, sont stockés dans le processus de codage. Deuxièmement, comment combiner les informations de couleur et d'échelle de gris des pixels pour *apprendre* un modèle, ce qui est essentiellement un problème d'apprentissage semi-supervisé.

Dans une première partie du projet, et selon l'évolution des résultats, il s'agira de traiter la classification d'images compressées en niveaux de gris pour une première reconnaissance directe et rapide de l'efficacité des méthodes utilisées.

3.1 Un premier exemple : chats et chiens

Les datasets chats vs. chiens sont des jeux de données de vision par ordinateur standard qui consiste à classer les photos contenant soit un chien, soit un chat.

Bien que le problème semble simple, il n'a été résolu efficacement qu'au cours des dernières années grâce à l'utilisation de **réseaux neuronaux convolutifs d'apprentissage profond (CNN)**.

3.2 Préparation des datasets

Une séléction manuelle est à réaliser en premièr lieu selon ses propres choix. Une simple recherche google images permet de rassembler un certain nombre de photos de chats et chiens, en couleur et aux formes et tailles différentes, à ranger dans deux dossiers distincts bien sûrs, qu'on s'avisera de nommer respectivement `cats` et `dogs`.

Les photos devront être remodelées avant la modélisation afin que toutes les images aient la même forme. Il s'agit souvent d'une **petite image carrée**, et il existe de nombreuses façons d'y parvenir, bien que la plus courante soit une simple opération de redimensionnement. Des entrées plus petites signifient un modèle plus rapide à entraîner, et cette préoccupation domine généralement le choix de la taille de l'image. Dans ce cas, nous suivrons cette approche et choisirons une petite taille fixe.

3.3 La compression ...

Pour étudier l'effet de la compression sur l'entrainemant de nôtre réseaux de neuronnes, on s'intéresse au concept de **décomposition en valeurs singulières (SVD)** qu'on a appris pendant le cours de *Calcul Scientifique*, et est liée essentiellement à la factorisation d'une matrice. Une matrice A peut être factorisée comme $A = U\Sigma V$ où les colonnes de U sont constituées des **"vecteurs singuliers gauches"**, les colonnes de V sont constituées des "vecteurs singuliers droits" et les entrées diagonales de Σ sont constituées des **"valeurs singulières"**.

La matrice A peut être approximée, c'est-à-dire compressée en factorisant la matrice à l'aide de SVD et en la recomposant en utilisant un sous-ensemble de vecteurs singuliers et de valeurs singulières au lieu de les utiliser tous.

La valeur r dans le programme peut être ajustée pour différents résultats. Une valeur élevée de r est une meilleure approximation de l'image d'origine tandis qu'une valeur inférieure est moins bonne. Vous trouverez ci-dessous des exemples sur une image de test avec différentes valeurs de r :

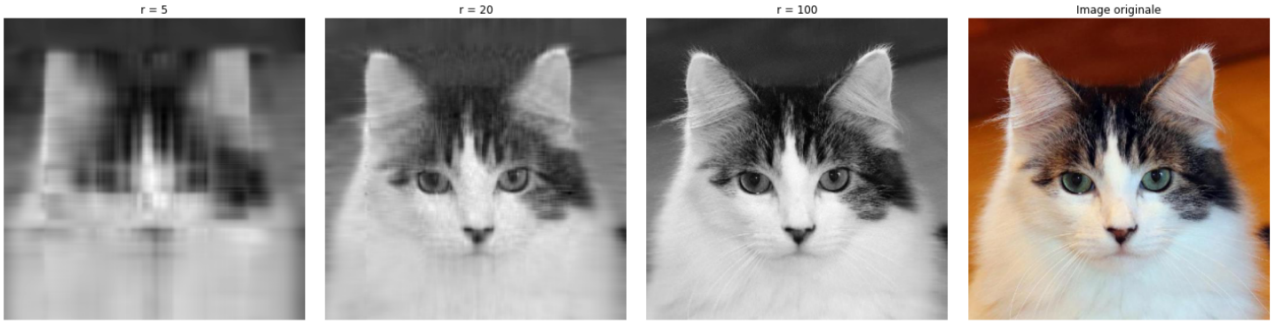

FIGURE 1 – Niveaux de compression d'images par la méthode de l'SVD.

3.4 Introduction au modèle CNN de base

Dans ce qui suit, nous pouvons développer **un modèle de réseau neuronal convolutif de base** pour le jeu de données chats vs chiens.

Un modèle de base établira une performance de modèle minimale à laquelle tous nos autres modèles pourront être comparés, ainsi qu'une architecture de modèle que nous pourrons utiliser comme base d'étude et d'amélioration.

Les principes architecturaux généraux des modèles VGG constituent un bon point de départ, car ils ont une structure modulaire d'architecture facile à comprendre et à mettre en œuvre.

L'architecture consiste à empiler des **couches convolutives** avec de petits filtres 3×3 suivis d'une couche de mise en commun maximale. Ensemble, ces couches forment un bloc, et ces blocs peuvent être répétés où le nombre de filtres dans chaque bloc est augmenté avec la profondeur du réseau, comme 32, 64, 128, 256 pour les quatre premiers blocs du modèle. Le remplissage est utilisé sur les couches convolutives pour garantir que les formes de hauteur et de largeur des caractéristiques de sortie correspondent aux entrées.

Nous pouvons explorer cette architecture sur le problème chats vs chiens et comparer un modèle avec cette architecture avec 1, 2 et 3 blocs.

Chaque couche utilisera la fonction d'activation ReLU et l'initialisation des poids He, qui sont généralement les meilleures pratiques. Par exemple, une architecture de style VGG à 3 blocs où chaque bloc possède une seule couche de convolution pouvant être définie dans Keras.

3.5 Développement du modèle CNN de base

Nous allons décrire un modèle ajusté sur l'ensemble de données et pouvant être personnalisé pour définir différents modèles de base, par exemple des versions du modèle avec 1, 2 ou 3 blocs de style VGG.

Aussi, le problème étant une tâche de classification binaire, des valeurs de 0 ou 1 seront à **prédire**. Une couche de sortie avec un nœud et une activation sigmoïde sera utilisée et le modèle sera optimisé en utilisant la fonction de perte d'entropie croisée binaire.

3.6 Entraînement et test du modèle

Nous pouvons ensuite ajuster le modèle à l'aide de l'itérateur `train` et utiliser l'itérateur `test` comme ensemble de données de validation pendant la formation. Le nombre d'étapes pour les itérateurs `train` et `test` doit être spécifié : **il s'agit du nombre de lots qui composent un "epoch"** [1]. Il peut être spécifié via la longueur de chaque itéra-

1. Un **epoch** est un terme utilisé dans le machine learning qui indique le nombre de passages de l'ensemble des données d'apprentissage que l'algorithme a effectué. Les ensembles de données sont généralement regroupés en **batches** (surtout lorsque la quantité de données est très importante). Certaines personnes utilisent le terme "itération" de manière vague et font référence au passage d'un batch dans le modèle comme une itération.

teur, et sera le nombre total d'images dans les répertoires train et test divisé par la taille du lot.

NB : Le modèle sera ajusté pendant 20 époques, un petit nombre pour vérifier la facilité de l'apprentissage.

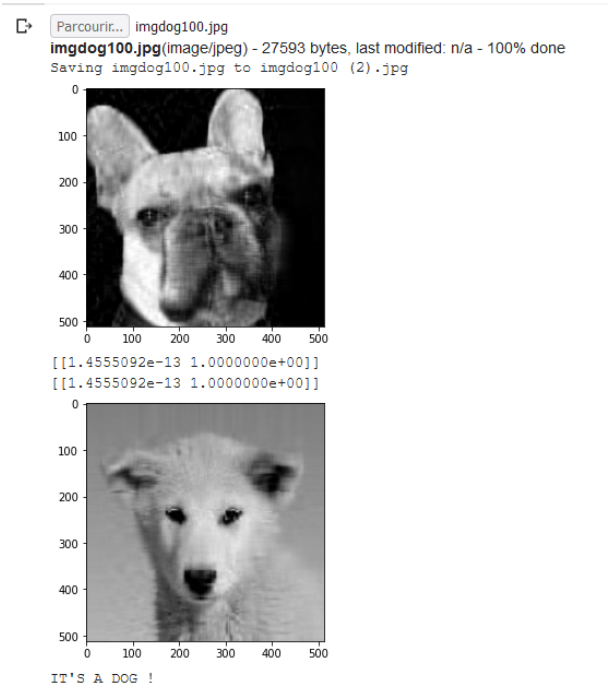

FIGURE 2 – Test du modèle par une image de chien.

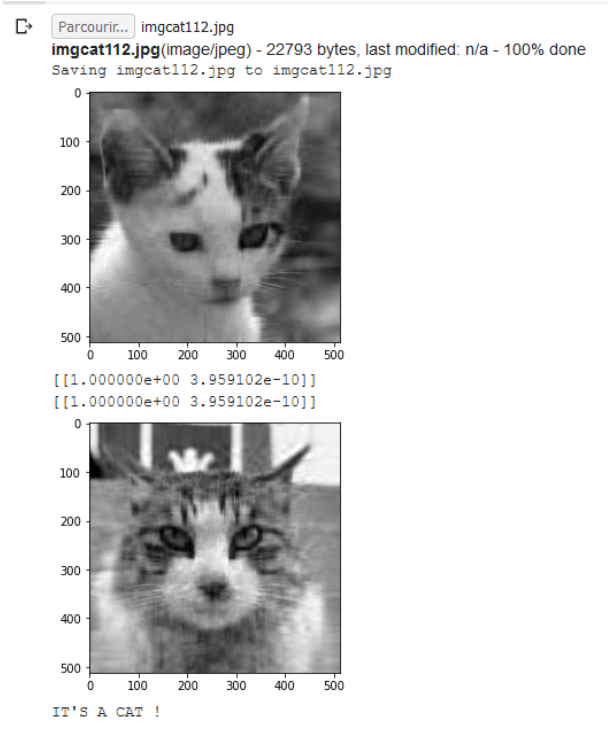

FIGURE 3 – Test du modèle par une image de chat.

3.7 History et accuracy

Enfin, nous pouvons créer un graphique de l'historique collecté pendant l'entraînement, stocké dans le répertoire `history`.

L'historique contient **l'accuracy** [2] et la perte du modèle sur les ensembles de données de test et d'entraînement à la fin de chaque epoch. Les tracés linéaires de ces mesures en fonction des epoch d'apprentissage fournissent des courbes pouvant être utilisées pour savoir si le modèle est **surajusté**, **sous-ajusté** ou s'il est **bien ajusté**.

Voici alors les courbes obtenues selons différentes valeurs des composantes principales calculées par SVD (notées r) :

2. L'accuracy du modèle est définie comme le nombre de classifications qu'un modèle prédit correctement, divisé par le nombre total de prédictions effectuées. C'est un moyen d'évaluer la performance d'un modèle, mais certainement pas le seul.

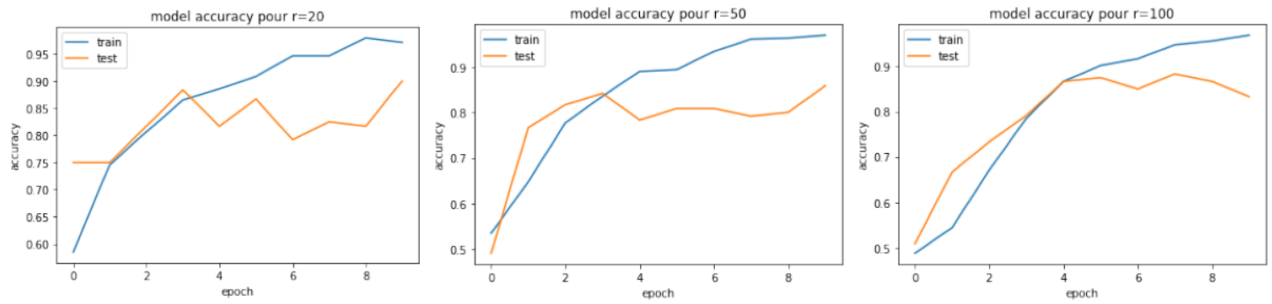

FIGURE 4 – Accuracy en fonction du nombre d'epoch

En examinant ce graphique, nous pouvons voir que le modèle est **bien ajusté** quand l'ensemble de données de formation est à environ 8 époch. Des résultats plus **raisonnables** sont aussi remarqués pour des valeurs élevées de r, ce qui peut être justifié par l'utilité des composantes principales dans la détermination des données essentielles relatives à l'identification d'une image quelconque.

Les déductions précédentes peuvent être résumées dans un seul et même graphique pour plus de lisibilité :

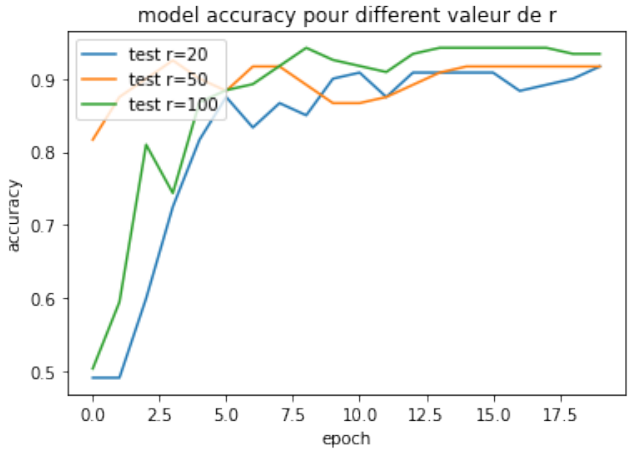

FIGURE 5 – Précision de la validation.

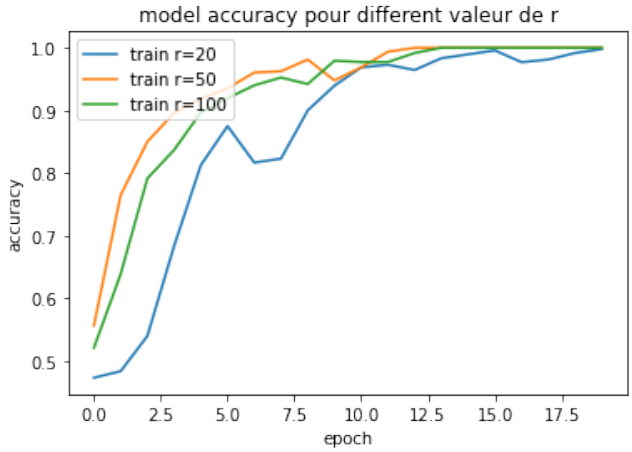

FIGURE 6 – Précision de l'apprentissage.

Comme on peut l'observer sur les figures ci-dessus, la précision sur l'ensemble d'apprentissage est presque parfaite avec environ 99,9%. tandis que la précision sur l'ensemble de test atteint à peine 88% pour un taux de compression élevé ($r = 20$). Cela indique que le réseau a du mal à se généraliser.

4 Prochaines approches possibles

Nous constatons une tendance à l'amélioration des performances avec l'augmentation de la capacité, mais aussi un cas similaire d'overfitting survenant de plus en plus tôt dans l'exécution.

Les résultats suggèrent que le modèle bénéficiera probablement de techniques de régularisation. Il peut s'agir de techniques telles que le **dropout**, la **décroissance des poids** et l'**augmentation des données**. Ces dernières peuvent également améliorer les performances en encourageant le modèle à apprendre des caractéristiques qui sont davantage invariantes par rapport à la position en élargissant l'ensemble de données d'entraînement.

5 Première conclusion

L'examen des courbes d'apprentissage du modèle pendant la formation a montré que le modèle présentait de forts signes de *surajustement*. Nous pouvons explorer deux approches pour tenter de remédier à ce surajustement : **la régularisation par abandon** et **l'augmentation des données**.

Ces deux approches devraient ralentir le taux d'amélioration pendant la formation et, éventuellement, contrer le surajustement de l'ensemble de données de formation. Ainsi, nous augmenterons le nombre d'époch d'apprentissage de 20 à 50 pour donner au modèle plus d'espace pour le raffinement.

6 Classification des images CIFAR-10

6.1 Jeu de données CIFAR-10

CIFAR est un acronyme qui signifie *Canadian Institute For Advanced Research* (Institut canadien de recherches avancées). Le jeu de données CIFAR-10 a été développé en même temps que le jeu de données CIFAR-100 par des chercheurs de l'institut CIFAR.

Le jeu de données est composé de 60 000 photographies couleur de 32×32 pixels d'objets appartenant à 10 classes, comme des grenouilles, des oiseaux, des chats, des bateaux, etc. Les étiquettes de classe et leurs valeurs entières standard associées sont énumérées ci-dessous :

- 0 : avion
- 1 : automobile
- 2 : oiseau
- 3 : chat
- 4 : cerf
- 5 : chien
- 6 : grenouille
- 7 : cheval
- 8 : bateau
- 9 : camion

Il s'agit de très petites images, beaucoup plus petites qu'une photographie typique, et l'ensemble de données était destiné à la recherche en vision par ordinateur.

6.2 Utilité algorithmique

CIFAR-10 est un jeu de données bien connu et largement utilisé pour évaluer les algorithmes de vision par ordinateur dans le domaine de l'apprentissage automatique. Le problème est "résolu". Il est relativement simple d'obtenir une précision de classification de 80 %. Les meilleures performances sont obtenues par les réseaux de neurones convolutifs à apprentissage profond, avec une précision de classification supérieure à 90 % sur le jeu de données de test.

L'exemple que nous avons proposé charge l'ensemble de données CIFAR-10 à l'aide de l'API Keras et crée un graphique des neuf premières images de l'ensemble de données d'apprentissage. L'exécution de l'exemple charge les jeux de données d'entraînement et de test CIFAR-10 et fournie leur forme. Nous pouvons voir qu'il y a 50 000 exemples dans le jeu de données d'entraînement et 10 000 dans le jeu de données de test et que les images sont effectivement carrées avec 32×32 pixels et en couleur, avec trois canaux.

Un graphique des neuf premières images de l'ensemble de données est également créé. Il est clair que les images sont en effet très petites comparées aux photographies modernes ; il peut être difficile de voir ce qui est exactement représenté dans certaines des images étant donné la résolution extrêmement basse. Cette faible résolution est probablement la cause des performances limitées que les algorithmes haut de gamme sont capables d'atteindre sur ce jeu de données :

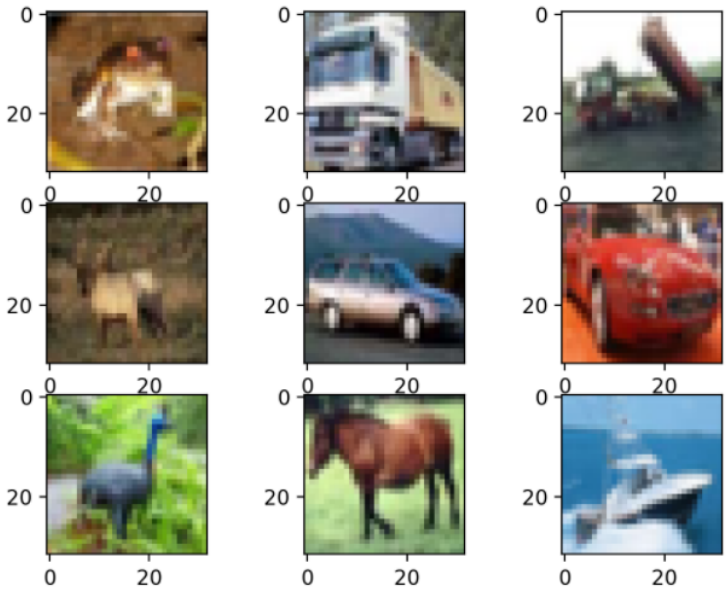

FIGURE 7 – Tracé d'un sous-ensemble d'images de la base de données CIFAR-10

6.3 Evaluation de notre modèle

Une fois le modèle défini, nous devons l'**ajuster** et l'**évaluer**.

L'ajustement du modèle nécessite de spécifier le nombre d'époch d'apprentissage et la taille du lot. Pour l'instant, nous utiliserons un nombre générique de 10 époch d'entraînement et une taille de lot modeste de 64.

Il est préférable d'utiliser un ensemble de données de validation séparé, par exemple en divisant l'ensemble de données d'entraînement en ensembles d'entraînement et de validation. Nous ne diviserons pas les données dans ce cas, et utiliserons plutôt l'ensemble de données de test comme ensemble de données de validation pour garder l'exemple *simple*.

L'ensemble de données de test peut être utilisé comme un ensemble de données de validation et évalué à la fin de chaque époch d'apprentissage. Il en résultera une trace des scores d'évaluation du modèle sur les jeux de données de formation et de test à chaque époch :

```
Epoch 1/10
78/78 [==============================] - 26s 279ms/step - loss: 1.7475 - accuracy: 0.3701 - val_loss: 45.0146 - val_accuracy: 0.1076
Epoch 2/10
78/78 [==============================] - 24s 294ms/step - loss: 1.5150 - accuracy: 0.4824 - val_loss: 63.6303 - val_accuracy: 0.1351
Epoch 3/10
78/78 [==============================] - 23s 281ms/step - loss: 1.3955 - accuracy: 0.5267 - val_loss: 73.3440 - val_accuracy: 0.1256
Epoch 4/10
78/78 [==============================] - 23s 282ms/step - loss: 1.3010 - accuracy: 0.5709 - val_loss: 141.4397 - val_accuracy: 0.1286
Epoch 5/10
78/78 [==============================] - 31s 386ms/step - loss: 1.2192 - accuracy: 0.6038 - val_loss: 116.6864 - val_accuracy: 0.1380
Epoch 6/10
78/78 [==============================] - 23s 282ms/step - loss: 1.1652 - accuracy: 0.6198 - val_loss: 253.8134 - val_accuracy: 0.1128
Epoch 7/10
78/78 [==============================] - 23s 279ms/step - loss: 1.1028 - accuracy: 0.6442 - val_loss: 253.6351 - val_accuracy: 0.1247
Epoch 8/10
78/78 [==============================] - 23s 279ms/step - loss: 1.0597 - accuracy: 0.6571 - val_loss: 149.5810 - val_accuracy: 0.2063
Epoch 9/10
78/78 [==============================] - 22s 279ms/step - loss: 1.0022 - accuracy: 0.6866 - val_loss: 232.4452 - val_accuracy: 0.1669
Epoch 10/10
78/78 [==============================] - 22s 275ms/step - loss: 0.9708 - accuracy: 0.6865 - val_loss: 171.9665 - val_accuracy: 0.1780
```

FIGURE 8 – Evolution numérique des epoch et accuracy correspondantes

Nous remarquons des **fluctuations de valeurs d'accuracy particulièrement faibles** mais attestant de l'inneficacité de l'entraînement usuel sur de tels modèles. Le graphique suivant permettra plus de clarté :

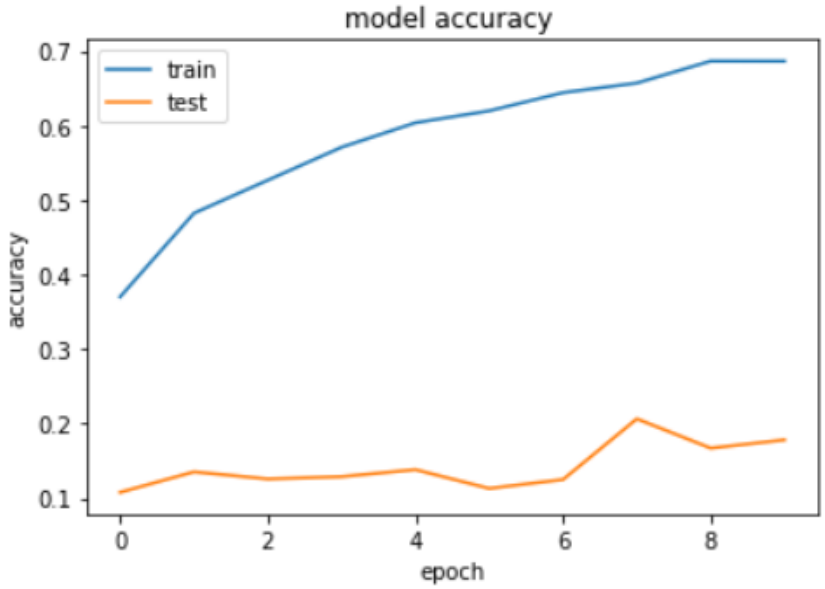

FIGURE 9 – Accuracy en fonction du nombre d'epoch (CIFAR-10)

NB : Les résultats peuvent varier en raison de la nature stochastique de l'algorithme ou de la procédure d'évaluation, ou des différences de précision numérique. Dans ce cas,

nous pouvons voir que le modèle a atteint une précision d'apprentissage d'un peu moins de 70 %.

Nous pouvons voir que le modèle s'adapte rapidement à l'ensemble de données de test. Ceci est clair si nous regardons le graphique, nous pouvons voir que la performance du modèle sur l'ensemble de données d'entraînement (bleu) continue à s'améliorer tandis que la performance sur l'ensemble de données de test (orange) s'améliore, puis commence à se détériorer dès le premier epoch.

6.4 Canny edge detector x CIFAR-10

6.4.1 Filtre canny edge detector

Les ***bords*** sont des caractéristiques importantes dans une image car ils représentent des changements d'intensité locaux significatifs. Ils fournissent des indices importants pour séparer des régions au sein d'un objet ou pour identifier des changements d'illumination.

La plupart des applications de télédétection, telles que le recalage d'images, la segmentation d'images, la séparation de régions, la description et la reconnaissance d'objets, utilisent la détection des contours comme étape de prétraitement pour l'extraction de caractéristiques. Les images réelles, telles que les images de télédétection, peuvent être corrompues par un bruit ponctuel. Le vrai problème est de savoir comment améliorer les images de télédétection bruyantes et extraire simultanément les bords.

Canny edge detector est basé sur le calcul de l'amplitude du gradient au carré. Les maxima locaux de l'amplitude du gradient qui sont supérieurs à un certain seuil sont ensuite identifiés comme des bords. Cette méthode de détection des maxima locaux avec seuil est appelée **suppression des non-maximums**, ou **NMS**. La motivation de l'opérateur de bord de Canny était de dériver un opérateur "*optimal*" dans le sens où il minimise la probabilité de détecter plusieurs fois un bord, minimise la probabilité de ne pas détecter un bord et minimise la distance entre le bord signalé et le vrai bord. Les deux premiers critères traitent de la question de la **détection**, c'est-à-dire qu'étant donné qu'un bord est présent, le détecteur de bord trouvera-t-il ce bord (et aucun autre bord). Le troisième critère concerne la **localisation**, c'est-à-dire la précision avec laquelle la position d'un bord est signalée. Il y a un compromis entre la détection et la localisation : **plus le détecteur est précis, moins la localisation est précise et vice-versa**.

En utilisant un filtre **canny ddge detector**, mis en œuvre pour l'extraction de caractéristiques et comme outil d'amélioration des images de télédétection, **le résultat avec des images usuelles est robuste avec un niveau d'amélioration très élevé**.

6.4.2 Pour un dataset CIFAR-10

Le jeu de données choisi pour évaluer notre modèle étant le CIFAR-10, les entrées sont déjà détériorées afin d'évaluer la robustesse accrue comparée au modèle CNN proposé précedemment, et qui sera aussi présenté plus tard (pour CIFAR-10). Dans le premier flux,une version en niveaux de gris du CIFAR-10 est utilisée, et dans le second flux, une extraction des bords - créée avec le détecteur Canny en question - du premier flux, sont présentées respectivement dans la figure ci-contre, avec les valeurs d'abscisses et ordonnées correspondantes :

FIGURE 10 – Application du filtre canny edge detector sur une image aléatoire de la dataset CIFAR-10

La figure confirme clairement que notre signal de détection des bords **n'ajoute pas d'informations cohérentes au réseau**, comme prévu. De plus, le flux en niveaux de gris n'a pas une bonne performance en soi.

6.5 Conventional gaussian filter x CIFAR-10

Après un filtrage canny edge detector, un deuxième filtrage s'avère nécessaire pour la suppression du bruitage additionnel, surtout en ce qui est de datasets tel CIFAR-10. On adoptera ainsi une piste étroitement liée à ce qui a été considéré pour les premiers exemples de chats et chiens.

6.5.1 Ré-entrée en matière des CNN

Nous avons récemment assisté à un regain d'attention pour les **réseaux de neurones convolutifs (CNN)** en raison de leurs performances élevées pour les tâches de reconnaissance visuelle à grande échelle. L'architecture des CNN est relativement *simple* et consiste en des couches successives organisées de manière hiérarchique ; chaque couche implique des convolutions avec des filtres appris, suivies d'une non-linéarité ponctuelle et d'une opération de sous-échantillonnage appelée "feature pooling". Il a été observé empiriquement que la représentation d'image résultante est invariante aux perturbations de l'image et qu'elle code des motifs visuels complexes, propriétés utiles pour la reconnaissance visuelle. **L'apprentissage des CNN reste cependant difficile**, car les réseaux à haute capacité peuvent impliquer des milliards de paramètres à apprendre, ce qui nécessite à la fois une puissance de calcul élevée, par exemple des `GPU`, et des techniques de régularisation appropriées.

6.5.2 Filtre gaussien conventionnel

Il s'agit ici d'une tentative d'adaptation et d'extension de la stratégie de filtrage gaussien d'image proposée pour les datasets de chats et chiens. Nous développons un filtre gaussien adaptatif et anisotrope agissant sur les normales du maillage.

Le filtre gaussien est largement utilisé dans le traitement des signaux et des images depuis de nombreuses années. Le filtrage gaussien ou dérivé gaussien est à plusieurs égards optimal pour les applications nécessitant des filtres passe-bas ou des moyennes mobiles. Dans cet article, une technique de suppression du bruit très efficace basée sur un filtre gaussien dynamique modifié est présentée. Appelé filtre lisse, le filtre gaussien est connu pour être plus efficace pour conserver les détails et les légères frontières que les autres filtres. les détails et les légères frontières que les autres filtres.

6.6 Résultat de la combinaison des deux filtrages

Voici un état d'art du fruit des pécédentes opérations :

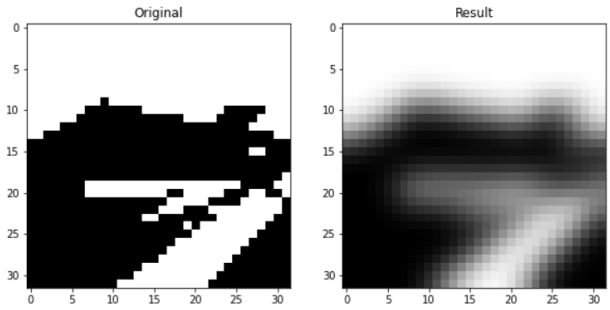

FIGURE 11 – Application des filtres canny edge detector et conventional gaussian sur une image aléatoire de la dataset CIFAR-10

Comme on pouvait s'y attendre, l'image obtenu est difficile, voir impossible à identifier par le modèle. La majorité des informations essentielles résumant sa description fûrent supprimées, que ce soit au niveau des bords, des maillages, des contours ...

La combinaison des noyaux et des réseaux de neurones convolutifs a montré que le **mélange des idées de ces deux concepts est infructueux**, puisque nous obtenons des performances proches de l'état de l'art sur plusieurs ensembles de données tels que CIFAR-10, avec des architectures simples et sans augmentation des données.

7 Pourquoi CIFAR-10 et pas MNIST ?

Un problème d'amélioration des performances des réseaux neuronaux convolutifs est considéré. Un paramètre de l'ensemble de formation est étudié. Ce paramètre est la taille du lot. L'objectif est de trouver un impact de la taille du lot de l'ensemble d'entraînement sur la performance. Pour obtenir des résultats cohérents, divers ensembles de données sont utilisés. Il s'agit de MNIST et de CIFAR-10. **La simplicité de l'ensemble de données MNIST** s'oppose à **la complexité de l'ensemble de données CIFAR-10**, bien que l'ensemble de données le plus simple comporte 10 classes, tout comme l'ensemble de données le plus complexe. Pour obtenir des résultats de test acceptables, différentes architectures de réseaux neuronaux convolutifs sont sélectionnées pour les jeux de données MNIST et CIFAR-10, avec respectivement deux et cinq couches convolutives. L'hypothèse concernant la dépendance de la précision de reconnaissance par rapport à la valeur de la taille du lot est confirmée : **plus la valeur de la taille du lot est grande, plus la précision de reconnaissance est élevée**. Une autre hypothèse concernant l'impact du type de la valeur de la taille du lot sur la performance du CNN n'est *pas confirmée*.

Le problème de l'amélioration des performances des CNN est pertinent et n'est toujours pas résolu. L'ajustement des paramètres de formation des CNN est l'un des moyens d'accomplir cette tâche. Le paramètre de formation de la taille du lot a été étudié dans cet article. comme précisé dans le paragraphe précédent, **le paramètre de la taille du lot a un effet crucial sur la précision de la reconnaissance d'images**. Plus la valeur du paramètre est grande, plus la précision de la reconnaissance d'images est élevée.

D'autre part, une valeur élevée de la taille du lot entraîne des coûts de calcul énormes. Les résultats de l'étude n'ont pas confirmé l'hypothèse de l'impact d'un certain type sur la valeur de la taille du lot : ni les nombres à la puissance deux ni les nombres multiples de dix n'entraînent un changement critique dans la *précision* de la reconnaissance. Par conséquent, la taille optimale du lot varie entre 200 et plus, en fonction des ressources de calcul.

8 Seconde conclusion

Les résultats prouvent que les filtres proposés *n'échappent pas* au flou élevé introduit et dépassent les résultats d'un filtre médian, surtout lorsqu'il s'agit de fortes densités de bruit.

En fin de compte, l'approche proposée **combine le comportement d'un filtre passe-bas dynamique intelligent** qui élimine uniquement les hautes fréquences correspondant au bruit et **une approche statistique basée sur un filtre qui élimine efficacement le bruit impulsif et conserve les détails et les frontières inchangés**, mais sans pour autant démontrer de meilleurs performances aux niveaux d'un cadre CIFAR-10 à titre d'exemple.

9 Classification pour image standard en couleur

Nous avons remarqué dans ce qui précède des résultats moyenement satisfaisants avec CIFAR-10. Une telle démarche de filtarge tient son inneficacité de par la définition même du type d'images qui a été adopté. Il sera alors question dans cette partie de s'assurer de la possibilité d'obtenir de meilleurs résultats (même légèrement) pour des images en couleur (des chats une fois de plus).

9.1 Prétraitement

Toutes les images peuvent être traitées quelle que soit leur valeur (0..1 ou 0..255) ou leur couleur (Binaire, Niveaux de gris, RGB ou RGBA). Mais pour des raisons de temps d'exercice, nous avons décidé de garder le prétraitement comme d'habitude. L'image est quantifiée à 0..255 niveaux de gris. Bien qu'augmenter le contraste pourrait également l'améliorer, nous n'avons pas procédé ainsi car cela aurait rendu l'image plus nette, donc plus de bruit et rendu le flou gaussien obsolète.

Aucun changement n'a été effectué dans cette partie. Nous avons essayé de modifier le paramètre `w`. Au lieu de 11 précédemment, il a été porté à 15 car ajouter plus de contexte au noyau serait mieux, mais avec un sigma de 1.4, 11 est juste un bon chiffre. Changer le sigma ne donnera pas un meilleur résultat car une valeur plus élevée effacerait et élargirait le bord réel, et une valeur plus faible ferait apparaître du bruit.

FIGURE 12 – Mise en valeur des niveaux de gris

9.2 Lissage

Il a été dit que le filtre adaptatif fonctionne mieux comme mentionné dans un article de *Bing Wang* et *Shaosheng Fan*. Le seul compromis est que nous faisons `n` itérations et que nous utilisons la formule de distance euclidienne pour deux noyaux de gradient (ce qui signifie que le noyau de gradient est implémenté deux fois. Pour être précis, `n+1` fois).

9.3 Dégradation des images au passage par un filtre gaussien 2D

Essayons d'abord avec le **filtre gaussien classique** :

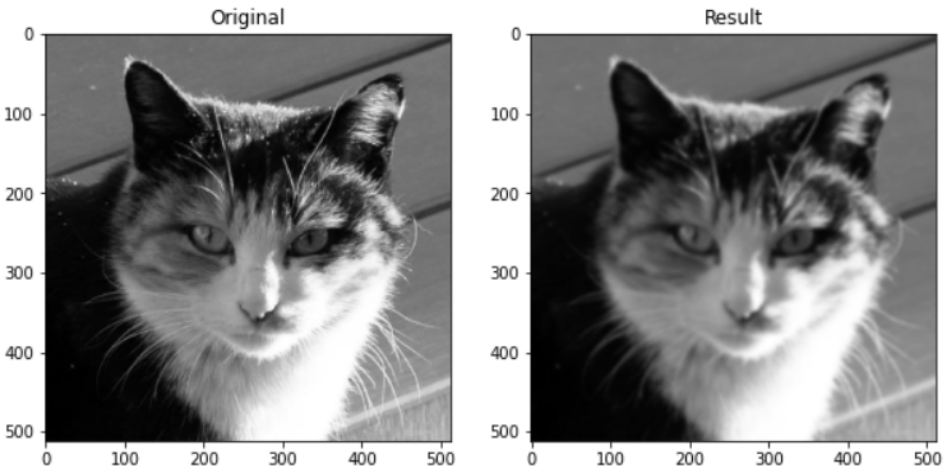

FIGURE 13 – Comparaison entre image originale et résulat

9.4 Première évaluation

Une fois le modèle défini, nous devons l'ajuster et l'évaluer comme fait auparavant. L'ajustement du modèle sera réalisé avec un nombre d'époch qu'on choisira égal à 26 et une taille de lot de 64. Voici ainsi la partie finale de l'évolution numérique déduite :

```
Epoch 10/25
160/160 [==============================] - 9s 54ms/step - loss: 0.0741 - accuracy: 0.9667 - val_loss: 0.5690 - val_accuracy: 0.8500
Epoch 11/25
160/160 [==============================] - 9s 54ms/step - loss: 0.0599 - accuracy: 0.9792 - val_loss: 0.7575 - val_accuracy: 0.8500
Epoch 12/25
160/160 [==============================] - 9s 54ms/step - loss: 0.0482 - accuracy: 0.9854 - val_loss: 0.5625 - val_accuracy: 0.8667
Epoch 13/25
160/160 [==============================] - 9s 54ms/step - loss: 0.0831 - accuracy: 0.9688 - val_loss: 0.8617 - val_accuracy: 0.8417
Epoch 14/25
160/160 [==============================] - 9s 53ms/step - loss: 0.0950 - accuracy: 0.9750 - val_loss: 0.5472 - val_accuracy: 0.8000
Epoch 15/25
160/160 [==============================] - 9s 54ms/step - loss: 0.0496 - accuracy: 0.9833 - val_loss: 0.6566 - val_accuracy: 0.8583
Epoch 16/25
160/160 [==============================] - 9s 53ms/step - loss: 0.0058 - accuracy: 1.0000 - val_loss: 0.8926 - val_accuracy: 0.8583
Epoch 17/25
160/160 [==============================] - 9s 54ms/step - loss: 0.0016 - accuracy: 1.0000 - val_loss: 0.9809 - val_accuracy: 0.8583
Epoch 18/25
160/160 [==============================] - 9s 57ms/step - loss: 5.5664e-04 - accuracy: 1.0000 - val_loss: 1.0392 - val_accuracy: 0.8583
Epoch 19/25
160/160 [==============================] - 9s 54ms/step - loss: 3.1619e-04 - accuracy: 1.0000 - val_loss: 1.0874 - val_accuracy: 0.8583
Epoch 20/25
160/160 [==============================] - 9s 53ms/step - loss: 2.1153e-04 - accuracy: 1.0000 - val_loss: 1.1286 - val_accuracy: 0.8583
Epoch 21/25
160/160 [==============================] - 9s 54ms/step - loss: 1.5362e-04 - accuracy: 1.0000 - val_loss: 1.1634 - val_accuracy: 0.8583
Epoch 22/25
160/160 [==============================] - 9s 54ms/step - loss: 1.1297e-04 - accuracy: 1.0000 - val_loss: 1.1980 - val_accuracy: 0.8583
Epoch 23/25
160/160 [==============================] - 9s 53ms/step - loss: 8.4165e-05 - accuracy: 1.0000 - val_loss: 1.2269 - val_accuracy: 0.8583
Epoch 24/25
160/160 [==============================] - 9s 54ms/step - loss: 6.4835e-05 - accuracy: 1.0000 - val_loss: 1.2543 - val_accuracy: 0.8583
Epoch 25/25
160/160 [==============================] - 9s 54ms/step - loss: 5.0799e-05 - accuracy: 1.0000 - val_loss: 1.2788 - val_accuracy: 0.8583
```

FIGURE 14 – Evolution numérique des epoch et accuracy correspondantes (pour les 15 dernièrs epoch)

Comme attendu, un résultat quasi-similaire à l'exemple des chats et chiens est remarqué pour ce premier filtrage. **L'accuracy atteint bien la valeur 1** témoignant d'une conformité test apprentissage dans le modèle utilisé. Il semble alors que le filtre adaptatif ait fait son travail assez bien : **il élimine le bruit mais préserve les bords réels**.

9.5 Calcul du PSNR et du SSIM pour les datasets compressées par la svd pour un nombre de composantes principales égale 50

Le **rapport signal/bruit de crête (PSNR)** et la **similarité de l'indice structurel (SSIM)** sont deux outils de mesure largement utilisés pour évaluer la qualité des images. En particulier dans l'image stéganographique, ces deux instruments de mesure sont utilisés pour mesurer la qualité de l'imperceptibilité. PSNR est utilisé plus tôt que SSIM, est facile, a été largement utilisé dans diverses mesures d'images numériques et a été considéré comme testé et valide. Le SSIM est un outil de mesure plus récent qui est conçu sur la base de trois facteurs, à savoir la luminance, le contraste et la structure, afin de mieux correspondre au fonctionnement du système visuel humain.

Les résultats obtenus pour l'exemple précedent sont comme suit :

PSNR : 26.99100209033164 , **SSIM** : 0.743821117465575

Si le niveau de bruit augmente, la qualité de récupération de l'image de sortie se détériore également. Les dégradations suivant la SVD s'adaptent ainsi pour une amélioration positive des valeurs indicatives, fournissant par suite les calculs :

PSNR : 28.83465742408829 , **SSIM** : 0.7838102175738757

Ceci indique qu'il existe un lien analytique simple entre le PSNR et le SSIM, qui fonctionne pour les dégradations courantes telles que le flou gaussien, le bruit gaussien additif, la compression jpeg et jpeg. Nous avons également entrepris une étude expérimentale afin d'évaluer la sensibilité du PSNR et du SSIM à ces dégradations, c'est-à-dire comment les

valeurs du paramètre associé à chacune de ces dégradations affectent les valeurs du PSNR et du SSIM. L'étude a révélé que la PSNR est plus sensible au bruit gaussien additif que le SSIM, alors que l'inverse est observé pour la compression jpeg compression. Les deux mesures ont une sensibilité légèrement similaire au flou gaussien et à la compression jpeg. Dans tous les cas, nous avons observé que le PSNR et le SSIM sont plus sensibles au bruit gaussien additif qu'au flou gaussien. Il apparaît alors que les valeurs du PSNR peuvent être prédites à partir du SSIM et vice-versa. Le site PSNR et le SSIM diffèrent principalement sur leur degré de sensibilité aux dégradations de l'image.

9.6 Dégradation des images au passage par un filtre adaptatif

Maintenant, le **filtre adaptatif**, qui ne traite que les pixels bruyants et utilise une taille de fenêtre fixe.

Cette technique a une grande capacité d'acclimatation et de changement en fonction de l'intensité du bruit. Plus précisément, le seuillage doux est appliqué aux zones à fort bruit, tandis que le filtre de variation totale est appliqué aux zones à faible bruit. L'auto-adaptation et la stabilité de la technique de filtrage adaptatif proposée ont permis à cette technique d'obtenir des performances optimales de réduction du bruit et de préserver les détails à haute fréquence spatiale (par exemple, les bords nets). Deuxièmement, étant donné qu'un seuil trop petit laisse la plus grande partie du bruit sans le supprimer, à l'inverse, un seuil trop grand ne permet pas de conserver les informations significatives de l'image telles que les bords. Par conséquent, dans l'algorithme que nous proposons, l'application de la fonction de filtrage adaptatif dans est basée sur un seuil adaptatif en plusieurs valeurs.

Pour ce dernier, nous allons utiliser l'opérateur Sobel et `n = 5` :

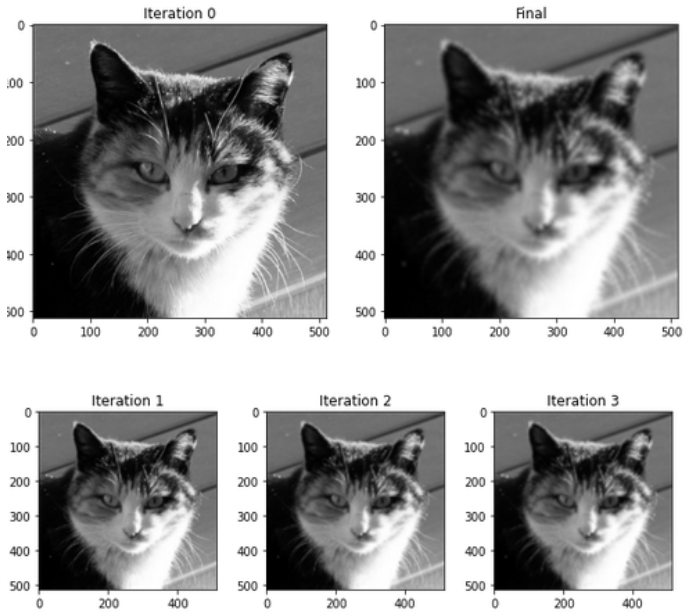

FIGURE 15 – Comparaison entre image originale et résulat (2)

Nous constatons que nous nous sommes rapprochés de la valeur du pixel original en utilisant la valeur des pixels voisins et la fenêtre adaptative. Nous avons ensuite obtenu les valeurs approximatives des pixels bruyants laissés par la première étape. Ensuite, nous avons montré que le filtre médian appliqué différemment est plus performant que les autres méthodes connues pour toutes les densités de bruit selon les résultats du rapport signal/bruit de crête et de la similarité structurelle. Nous devons maintenant nous demander : existe-t-il une réponse positive à la question de savoir s'il est possible d'obtenir de meilleurs résultats grâce à un processus de remplissage et de fenêtrage différent ?

9.7 Accuracy : Training

La détection des bords dans une image bruyante comporte de nombreux dangers tels que le défaut de précision, selon cela, dans l'algorithme que nous proposons, nous appliquons la méthode de clustering K-means sur l'image débruitée à la place de l'image bruyante. On obtient ainsi les répartitions suivantes au niveau du modèle exploité :

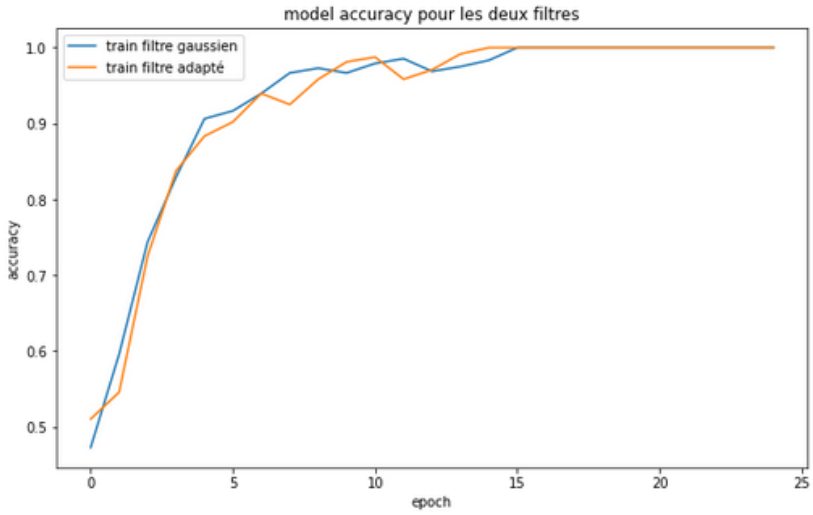

FIGURE 16 – Accuracy en focntion du nombre d'epoch pour les deux types de filtrage

L'overfitting a été réduit ou retardé, bien que les performances puissent commencer à stagner vers la fin de l'exécution. Les résultats suggèrent que des époques d'entraînement supplémentaires peuvent entraîner une amélioration supplémentaire du modèle. Il pourrait également être intéressant d'explorer un taux d'abandon légèrement plus élevé après les blocs VGG en plus de l'augmentation des époques d'entraînement.

9.8 Accuracy : Test

Dans cette partie, les performances de l'algorithme que nous proposons sont réalisées sur six images différentes en niveaux de gris 8 bits. Ces images standard de 8 bits en niveaux de gris ont été corrompues par un bruit blanc gaussien additif de moyenne nulle (AWGN) et utilisées pour évaluer la performance de sept algorithmes de débruitage différents. Toutes ces images contiennent des régions texturées et lisses. Ces images sont présentées selon trois critères d'évaluation utilisées pour comparer les performances des différents algorithmes : l'indice PSNR (Peak Signal to Noise Ratio), l'indice SSIM (Structural Similarity) et la qualité visuelle.

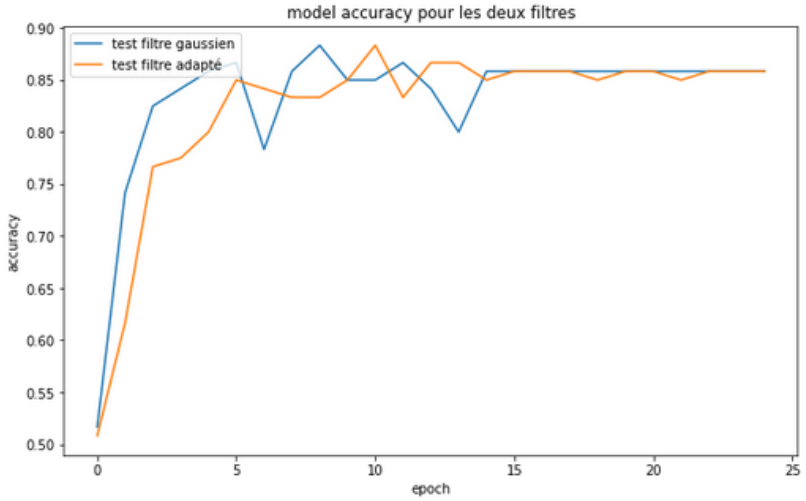

FIGURE 17 – Accuracy en focntion du nombre d'epoch pour les deux types de filtrage

Dans cette étape, nous appliquons d'abord une transformation linéaire au groupe mentionné à l'étape précédente. Ensuite, nous utilisons le filtrage adaptatif pour éliminer le bruit. Dans le filtrage adaptatif, le seuillage doux est appliqué aux zones de l'image qui sont fortement corrompues par le bruit, tandis que le filtre de variation totale est appliqué aux zones qui comportent de faibles niveaux de bruit. Pour obtenir les estimations de tous les blocs groupés, nous inversons la transformée linéaire. Cela s'explique par des estimations ramenées par des blocs à leur position d'origine.

9.9 Evolution en fonction du nombre de composantes principales

Avec un couple d'images stéréo réel, il est difficile de séparer l'influence de la compression d'image sur la précision de la correspondance des effets induits par une approximation imparfaite de la relation géométrique entre les deux images. C'est pourquoi nous effectuons les investigations avec des couples stéréo simulés et avec des couples d'images stéréo réels.

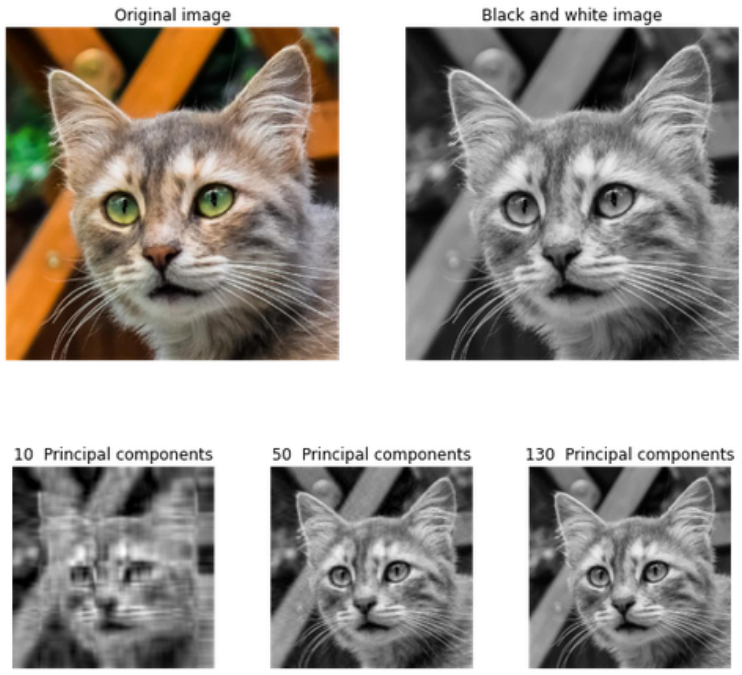

FIGURE 18 – Evolution de la compression suivant le nombre de composantes principales

En considérant une erreur induite par la compression de 0,1 pixel comme un niveau d'erreur acceptable pour la mise en correspondance d'images compressées, le transfert de points dans des paires d'images réelles est limité à des rapports de compression de 1 :5 (max 1 :10) pour les images compressées JPEG, 1 :3,5 (max 1 :6) pour les images couleur compressées si une procédure de mise en correspondance standard des moindres carrés est appliquée. Si le transfert de points est effectué avec un mécanisme d'autocontrôle, des limites pour les taux de compression de 1 :6 (max 1 :10) pour les images compressées par JPEG et 1 :5 (max 1 :7) pour les images compressées par ondelettes sont trouvées à partir de trois types différents d'ensembles de données d'image. Une compression fractale supplémentaire a été incluse dans les expériences. Comme ses performances en matière de qualité d'image et d'erreurs de transfert de points se situaient toujours dans la partie inférieure, elle ne peut être recommandée pour une utilisation en photogrammétrie numérique. Le classement selon lequel la compression des niveaux de gris donne de meilleurs résultats que la compression des normes de couleur était généralement assez serré, mais vrai pour toutes les expériences de transfert de points. En résumé, l'utilisation de la compression JPEG avec perte avec un rapport de 1 :5 ne devrait pas influencer négativement le travail photogrammétrique avec des images numériques. Même dans le transfert de points de haute précision, l'erreur produite à des taux de compression inférieurs ou égaux à 5 est de l'ordre de 0,1 pixel.

9.10 Seconde évaluation

Dans ce travail, un filtre flou basé sur un SVM est proposé. Les pixels bruyants et non bruyants sont classés en fonction de l'ensemble optimal de caractéristiques. Sur la base de la classification SVM, le filtrage est effectué pour ajuster les pixels corrompus et non corrompus. On observe que cette technique élimine le bruit non seulement des bruits impulsionnels de faible densité, mais qu'elle fonctionne aussi modérément sur les bruits impulsionnels de forte densité. Cela peut être dû à la préservation d'un plus grand nombre de détails de l'image pendant la phase de filtrage avec moins d'effet de flou car une plus grande homogénéité est préservée dans l'image reconstruite que dans l'image corrompue. On observe également que la technique proposée améliore non seulement le PSNR mais maintient également le SSIM avec une complexité temporelle modérée. Dans chaque cas, la méthode proposée surpasse la plupart des techniques existantes, même si l'image entraînée est différente de l'image testée. Des expériences ont été menées sur différents types de bases de données et les performances sont satisfaisantes dans tous les cas. En outre, le système a été testé pour d'autres types de bruits où une amélioration suffisante a été obtenue.

Bien que nous ayons proposé cette technique pour les images en échelle de gris, ce travail peut être mis en œuvre pour les images en couleur comme une extension de ce travail. Dans le cadre de travaux futurs, une méthode de repondération de l'importance utilisant une fonction de perte de substitution conçue pour un problème de classification traditionnel pourrait être mise en œuvre pour la classification des pixels bruyants et non bruyants.

10 Conclusion générale

Dans ce travail, **une fusion de treillis est proposée pour les réseaux de neurones convolutifs multi-flux**. La fusion proposée utilise des opérations mathématiques avant chaque couche de mise en commun d'une architecture CNN. En utilisant un réseau comme base de référence et l'ensemble de données CIFAR-10, nous avons implémenté trois versions différentes du modèle : un CNN multi-flux avec fusion tardive, un cross-CNN et notre lattice-CNN. Les résultats expérimentaux montrent que la fusion proposée surpasse tous les modèles susmentionnés d'au moins 28 % par rapport à tous les modèles testés. En outre, la fusion proposée a démontré sa robustesse et sa stabilité, même lorsque des distracteurs sont utilisés comme entrées. Les travaux futurs peuvent se concentrer sur différentes opérations et ensembles de données, en examinant comment les espaces de couleur pourraient améliorer la précision de nos modèles, y compris de nouveaux scénarios de banc d'essai, ensembles de données, modèles et architectures de CNN, et d'autres approches multi-flux disponibles (par exemple, LSTM, RNN, ...). En outre, des flux supplémentaires doivent être analysés et testés.

La mesure de la qualité de ce qui constitue l'image est une chose qui doit être faite pour mesurer la contribution de la méthode proposée. Plusieurs outils de mesure ont été utilisés en littérature, à savoir PSNR, SSIM et l'histogramme de l'image. Le PSNR est l'outil de mesure le plus apprécié car il est simple à calculer et est considéré comme valide car il a été utilisé dans de nombreux types de recherche sur le traitement des images dans le monde. Mais PSNR a une faiblesse pour mesurer la fidélité du signal où il est très étroitement lié à l'imperceptibilité dans les images stéganographiques. D'après les résultats des tests de cette recherche, le SSIM a une meilleure sensibilité pour détecter les distorsions qui se produisent en raison de l'intégration de messages sur des images en couleur, par rapport au PSNR, ceci est dû à la façon dont le SSIM fonctionne, qui est conçu sur la base du système visuel humain. Le SSIM est calculé sur la base de la luminance, du contraste et de la structure de l'image originale du conteneur et de l'image. Les résultats du tracé de l'histogramme de l'image renforcent également l'hypothèse selon laquelle, logiquement, le SSIM est plus approprié. Ainsi, avec les résultats de cette recherche, il est recommandé d'utiliser les outils de mesure SSIM dans toutes les recherches sur la dissimulation de données, en particulier la compression spatiale d'images.

En guise de conclusion finale, il apparaît que les valeurs du PSNR peuvent être prédites à partir du SSIM et vice-versa. Le PSNR et le SSIM diffèrent principalement par leur degré de sensibilité aux dégradations de l'image. sensibilité aux dégradations de l'image.