Débruitage d'image par analyse en composantes principales :

Livrable 2 : Code console

ING1 – Groupe 4

13 mai 2025

EL MOUDDEN Nassef
GUERIN Enzo
GOMEZ Vincent
PHANITHAVONG Manon
RODRIGUEZ Ruben

Professeurs encadrants : FASSI Dieudonné, FORTIN Nisrine, RANISAVLJEVIC Elisabeth

Lien vers le dépôt GitHub du projet

Sommaire

1 Introduction

Dans le cadre de notre Situation d'Apprentissage et d'Évaluation (SAÉ), nous devons réaliser un projet portant sur le débruitage d'images par Analyse en Composantes Principales (ACP).

Dans un contexte pédagogique, ce projet vise à approfondir nos connaissances techniques et méthodologiques en ingénierie, tout en nous permettant de mesurer notre aptitude à organiser, mener et documenter un projet bien structuré et collaboratif. Plus précisément, l'objectif principal est d'implémenter une méthode performante de réduction de bruit dans des images numériques, en se basant sur la parcimonie, la redondance et régularité intrinsèque des données visuelles.

Cette méthode est particulièrement pertinente dans un contexte actuel où la réduction dimensionnelle est nécessaire, non seulement pour améliorer la qualité des traitements d'images, mais aussi pour répondre à des enjeux environnementaux et sociaux par la diminution du volume de données traitées.

De plus la SAÉ présente un intérêt significatif en termes pédagogiques et professionnels puisqu'elle nous permet de développer des compétences essentielles pour notre future carrière d'ingénieur. Elle favorise notamment une approche rigoureuse de la résolution de problèmes, l'adoption de pratiques collaboratives, et l'intégration de bonnes pratiques environnementales et sociales dans les processus techniques. Notamment le projet permettra d'évaluer spécifiquement ces cinq compétences clés :

1. Apprendre à apprendre : Adopter une démarche réflexive.
2. Maîtriser la complexité : Abstraire un problème donné de manière scientifique.
3. Conduire un projet : adopter une démarche analytique et synthétique.
4. Collaborer et communiquer : Contextualiser et présenter le problème.
5. Innover : adopter une pensée systémique.

2 Méthodologie

2.1 Analyse en Composantes Principales

L'ACP, ou Analyse en Composantes Principales, est une méthode utilisée pour réduire la dimension d'un ensemble de données tout en gardant un maximum d'information. Pour obtenir cette réduction, il faut projeter nos variables sur un nouveau jeu de composantes qui forment une base orthonormée et qui récupère un maximum d'information. Pour savoir quelle composante récupère un maximum d'information, on regarde la valeur propre associée à la composante et on regarde la plus grande. Par propriété, on a que les valeurs propres sont toutes comprises entre 0 et 1, que leur somme vaut 1 et que leur valeur représente la quantité d'information gardée. Pour réduire la dimension, il suffit de garder les composantes qui gardent le plus d'information et on s'arrête quand la composante

suivante ne donne pas assez d'information par rapport à la règle du coude : on coupe sur le dernier grand saut d'information.

Pour les étapes du projet, on a besoin de l'ACP. Voici les étapes du projet :

— Créer une image bruitée X_b à partir d'une image d'origine X_0. Le bruit est gaussien, centré et de variance σ^2. Des cas sont à étudier si la valeur du pixel se trouve au-dessus de 255 et en-dessous de 0. On utilisera la fonction **noising**(X_0,σ).

— Extraire des patchs à partir de l'image X_b. On implémentera les fonctions **extractPatchs**(X_b) et renvoie une collection de patchs de l'image bruitée X_b

— Vectoriser les patchs en transformant des patchs de taille s x s en un vecteur de taille s^2.

— Effectuer une Analyse en Composantes Principales à partir des vecteurs des patchs et garder les composantes gardant le maximum d'information. On utilisera une librairie qui a déjà implémenté une méthode d'ACP car si on l'implémente nous-même, on aura trop d'erreurs de calcul sur la diagonalisation de la matrice avec les valeurs propres par rapport à un code optimisé. On cherche à avoir une image à la fin la mieux débruitée.

— Effectuer une projection dans le sous-espace des axes principaux en effectuant un seuillage. Différents seuillages et différents seuils sont à étudier comme le seuillage doux et le seuillage dur avec différents λ.

— Reconstruire une estimation de l'image X_0 à partir de la projection et seuillage.

2.2 Explication des différentes fonctions d'extraction de patches

On a implémenté 3 fonctions **extractPatchs**. On va après comparé ces méthodes sur le temps de calcul, sur le nombre de patchs extraits sur une même image et sur leur impact écologique en se comparant comme une grande entreprise de la tech. Et dans un prochain livrable, on comparera ces méthode dans le vrai cas de l'ACP pour voir si la redondance d'information est vraiment importante ou non.

2.2.1 extractPatchs1

Dans cette méthode, on extrait sur une image de taille lxc des patchs carrés de taille s*s avec s $\leq$ l et s $\leq$ c. Le premier patch est extrait à la coordonnée (0,0) et on se déplace par un décalage de 1 d'abord sur les colonnes et après avoir parcouru la première ligne, on se décale de 1 sur les lignes et on revient à la première colonne et ainsi de suite. Cette méthode représente celle avec le plus de redondance car on a ici le maximum de patch qui se croise.

2.2.2 extractPatchs2

Dans cette méthode, on extrait sur une image de taille lxc des patchs carrés de taille s*s avec s $\leq$ l et s $\leq$ c. Le premier patch est extrait à la coordonnée (0,0) et on se déplace

par un décalage de s/2 d'abord sur les colonnes et après avoir parcouru la première ligne, on se décale de s/2 sur les lignes et on revient à la première colonne et ainsi de suite. Cette méthode possède moins de redondance que la première.

2.2.3 extractPatchs3

Dans cette méthode, on extrait sur une image de taille l x c des patchs carrés de taille c*p+1 x c*p+1 avec p un pourcentage passé entre paramètre qui est compris entre 0 et 1. Pour la suite, on pose s=c*p+1 Le premier patch est extrait à la coordonnée (0,0) et on se déplace par un décalage de s/3 d'abord sur les colonnes et après avoir parcouru la première ligne, on se décale de s/3 sur les lignes et on revient à la première colonne et ainsi de suite. Cette méthode représente une extraction de patchs adaptatifs à l'image, les paramètres comme le pourcentage et le décalage seront étudiés dans le prochain livrable pour voir quels valeurs sont les plus efficaces.

2.2.4 extractPatchs4

Même méthode que la **extrtactPatch2** mais ici on a un décalge s. Dans cette méthode, on a plus aucune redondance d'information n'est présente. Cette méthode n'a pas encore été implémentée.

3 Démonstration de la décomposition en base orthonormale

Hypothèses : $(u_1, \ldots, u_{s^2})$ forment une base orthonormée de $\mathbb{R}^{s^2}$
Montrons que :

$$V_k = m_V + \sum_{i=1}^{s^2} u_i'(V_k - m_V)u_i$$

Soit $V_k \in \mathbb{R}^{s^2}$, on part de l'identité :

$$V_k = m_V + V_k - m_V$$

Comme la famille $(u_1, \ldots, u_{s^2})$ forme une base orthonormée de $\mathbb{R}^{s^2}$, on sait que tout vecteur $x \in \mathbb{R}^{s2}$ peut s'écrire comme :

$$x = \sum_{i=1}^{s^2} \langle x, u_i \rangle u_i$$

En particulier, on applique ceci à $x = V_k - m_V$, ce qui donne :

$$V_k - m_V = \sum_{i=1}^{s^2} \langle V_k - m_V, u_i \rangle u_i$$

Ainsi :

$$\begin{aligned} V_k &= m_V + (V_k - m_V) \\ &= m_V + \sum_{i=1}^{s^2} \langle V_k - m_V, u_i \rangle u_i \\ &= m_V + \sum_{i=1}^{s^2} u_i'(V_k - m_V) u_i \end{aligned}$$

Ce qui conclut la démonstration.

4 Résultats

4.1 Comparaison des méthodes d'extraction des patches

Dans cette partie, on compare les 3 méthodes d'extraction des patches. Dans chaque cas, on obtient exactement la même image que celle d'origine en reconstruisant avec les patches. Notre critère de choix se basera alors sur l'impact environnementale des différentes fonctions. Pour évaluer les performances de notre algorithme, nous avons mesuré le temps d'exécution de chaque fonction. Cette mesure a été effectuée en millisecondes (ms), en utilisant l'horloge système à l'aide de la méthode `System.currentTimeMillis()` en Java. La méthode utilisée consiste à enregistrer l'heure avant et après l'exécution de la portion de code concernée, comme suit :

```
long startTime = System.currentTimeMillis();

// Fonction à mesurer

long endTime = System.currentTimeMillis();
long executionTime = endTime - startTime;
System.out.println("Temps d'exécution : " + executionTime + " ms");
```

4.1.1 Estimation de la consommation énergétique

Pour estimer l'énergie, on utilise la formule suivante :

$$\text{Énergie (J)} = \text{durée (s)} \times \text{puissance (W)}$$

En supposant :
— un CPU consommant 25 W en charge,
— une durée de 15 ms pour `extracPatchs1` : $0.015 \times 25 = 0.375$ J,
— une durée de 20 ms pour `extracPatchs2` : $0.02 \times 25 = 0.5$ J,
— une durée de 3 ms pour `extractPatchs3` : $0.003 \times 25 = 0.075$ J,

4.1.2 Conversion en émission de CO_2

Hypothèse : On suppose qu'on est un grande entreprise comme Google. En 2020 on estime à 4 000 milliards de photos téléchargés si on suppose que l'on utilise une fonction d'extraction de patchs sur 0.1% pendant 5 ans quel serait le coût énergétique et environnementale ? L'ADEME estime que 1 kWh équivaut à environ 450 g de CO_2 (dans le monde).

Cela donne :

$$0.1 \times 4000 \times 10^9 \times 5 = 2 \times 10^{12} \text{ exécutions}$$

Conversion en kWh des 3 fonctions :

$$\frac{0.375 \times 2 \times 10^{12}}{3.6 \times 10^6} \approx \mathbf{208000}\text{kWh}$$

$$\frac{0.5 \times 2 \times 10^{12}}{3.6 \times 10^6} \approx \mathbf{278000}\text{kWh}$$

$$\frac{0.075 \times 2 \times 10^{12}}{3.6 \times 10^6} \approx \mathbf{41700}\text{kWh}$$

Différence de CO_2 entre les 3 fonctions :

— Pour `extracPatches1` : $208 \times 450 \approx \mathbf{93}$ tonnes de $CO_\mathbf{2}$,
— Pour `extracPatches2` : $278 \times 450 \approx \mathbf{125}$ tonnes de $CO_\mathbf{2}$,
— Pour `extracPatches3` : $41.7 \times 450 \approx \mathbf{18}$ tonnes de $CO_\mathbf{2}$,

4.1.3 Conclusion

Une optimisation algorithmique peut paraître négligeable à petite échelle, mais à l'échelle d'une infrastructure distribuée ou d'un service utilisé massivement, les économies en énergie et en CO_2 peuvent être considérables. C'est pourquoi on souhaite conserver la fonction `extracPatches3`.

5 Bibliographie

— Wikipedia contributors, *Google Photos*, Wikipedia. Disponible à : https://en.wikipedia.org/wiki/Google_Photos.
— *Définition de l'ACP*, Appviser. Disponible à : https://www.appvizer.fr/magazine/technologie/editeurs/acp-glossaire.
— Groupe 4, *Livrable 1 : Analyse du projet*, Disponible à : https://docs.google.com/document/d/1n1c1Wc7xWqYphTxk9MSXAvmsZzUdiJ5Cpoob_ioLVeE/edit?usp=sharing