Rapport Data Visualisation

Ben Gao

Master 1 MIASHS

Enseignant : Miguel PALENCIA-OLIVAR

Année : 2022/2023

l'Université lyon2

Sommaire

1 Introduction

Ce projet est réalisé dans le cadre du cours Data Visualisation, et a pour objectif d'apporter des contributions au projet de l'entreprise HALIAS, dont je fais partie en tant qu'alternant.

1.1 Présentation du projet de l'entreprise

Le projet de l'entreprise HALIAS consiste à créer un IA qui pourra détecter automatiquement des fuites de pétroles autour des plate-formes pétrolière des clients. Etant donné une image, l'IA devra pouvoir dire si la fuite est présente. Ceci pourrait être vu comme un problème de classification binaire. Mais en raison du manque d'images avec fuite, des méthodes de détection d'anormlies sont retenues par l'entreprise. Avec ces méthodes, nous n'avons besoins que des images sans fuite pour entraîner le model.

Pour surveiller ces plate-formes, l'entreprise a décidé d'utiliser les images SAR (Synthetic-aperture radar) du satellite sentinel-1 que l'ESA (European Space Agency) donne accès à tout le monde gratuitement. Plus précisément, nous pouvons à tout moment téléchager une image prise sur l'emplacement d'un plate-forme donné, puis la passer dans le model entraîné pour détecter une éventuelle fuite.

1.2 Présentation du dataset

Le dataset utilisé pour ce projet est TenGeoP-SARwv (Wang Chen, 2018). Il est de 17 GB, et contient plus de 37,000 images SAR (Synthetic-aperture radar) de sentinel-1, divisées en dix classes et labelisées en fonction de leurs spécificités géographiques : F pour Pure Ocean Waves, G pour Wind Streaks, H pour Micro Convective Cells, I pour Rain Cells, J pour Biological Slicks, K pour Sea Ice, L pour Iceberg, M pour Low Wind Area, N pour Atmospheric Front et O pour Oceanic Front. Etant donné qu'une plateforme pétrolière est rarement installée dans les zones que K,L,N et O représentent, nous ne les avons pas utilisés.

2 Problématique et Méthodes utilisés

La visualisation d'un dataset d'images n'est pas évidente. Parce que les images sont de grandes dimensions (la plus petite étant 469*431 dans notre dataset). Chaque pixel est comme une variable, pourtant il n'y a pas d'intérêt de créer des histogrammes pour les pixels. Il est donc nécessaire d'effectuer des méthodes de réduction de dimension afin de visualiser les images dans un graphique.

Ainsi, notre tâche consiste dans un premier temps à réduire les dimensions des données à 1,2 ou 3. Pour ce faire, nous avons testé plusieurs méthodes et finalement décidé de retenir l'ACP, l'ACP à noyau et l'Auto-Eucoder Convolutionnel qui ont donné des résultats intéressants. Les

graphiques obtenus sont des nuages de points en dimension 2 ou 3. Contrarirement à des graphiques purement descriptives (un boxplot par exemple), il nous faut une confirmation postérieure pour savoir si nos visualisations des données réduites ont bien un sens. Nous affichons donc leurs étiquettes pour voir si les données de classe différente forment des clusters.

De plus, nous présentons également la visualisation individuelle des images par wavelet transformation.

3 Librairies

Les librairies suivantes sont utilisées pour la visualisation, la réduction de dimension et les prétraitements :

- *rasterio* : télécharger les images satellite de type Tiff
- *cv2* : redimensionner les images
- *os.listdir* : leture des fichier dans un dossier
- *numpy* : gérer les grandes matrices
- *matplotlib.pyplot* : visualisation des résultats
- *sklearn.preprocessing.StandardScaler* : centrer et réduire les données
- *sklearn.decomposition.PCA* : implémenter l'ACP
- *sklearn.decomposition.KernelPCA* : implémenter l'ACP à noyau
- *tensorflow.keras* : implémenter l'Auto-Encoder et l'Auto-Encoder Convolutionnel
- *sklearn.model_selection.train_test_split* : séparer le jeu d'entraînement et le jeu de validation
- *pywt* : effectuer la transformation wavelet

4 Visualisation et Discussion

4.1 ACP et ACP à noyau

Dans cette partie, nous présentons les graphiques des résultats de l'ACP et l'ACP à noyau.

Voici la projection des images sur les deux axes principaux obtenus par l'ACP. La distribution des images dans l'espace réduit est très intéressante : Il semble que certains classes sont quasiment séparables vis-à-vis du premier axe principal. Figure 4.1.

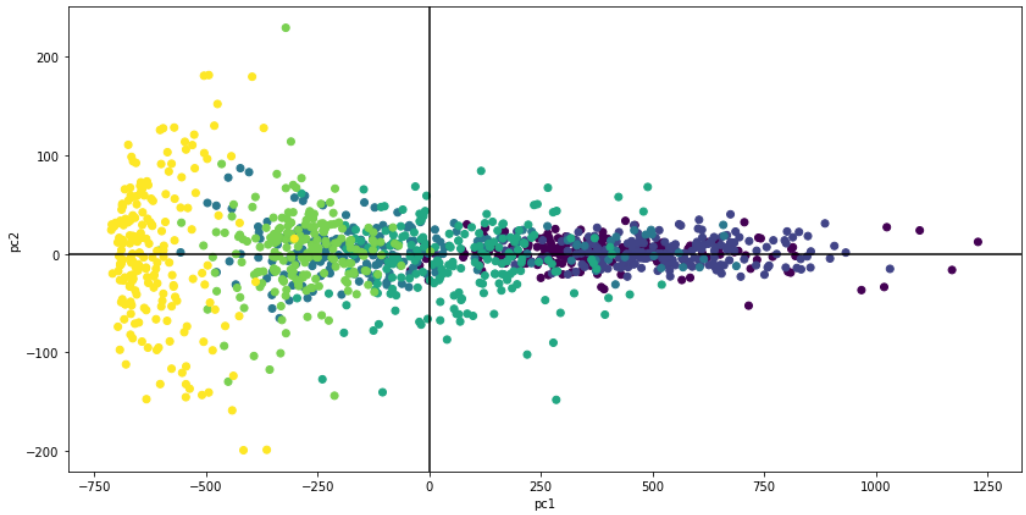


Figure 4.1: Projection sur 2 axes principaux

Le premier axe représente environ 88% d'inertie, le deuxième axe ajoute peu d'inertie (0.7%) par rapport au premier axe, le troisième axe encore moins (0.6%). Ainsi, la visualistaion en 3D n'est pas plus intéressante que celle en 2D. Figure 4.2

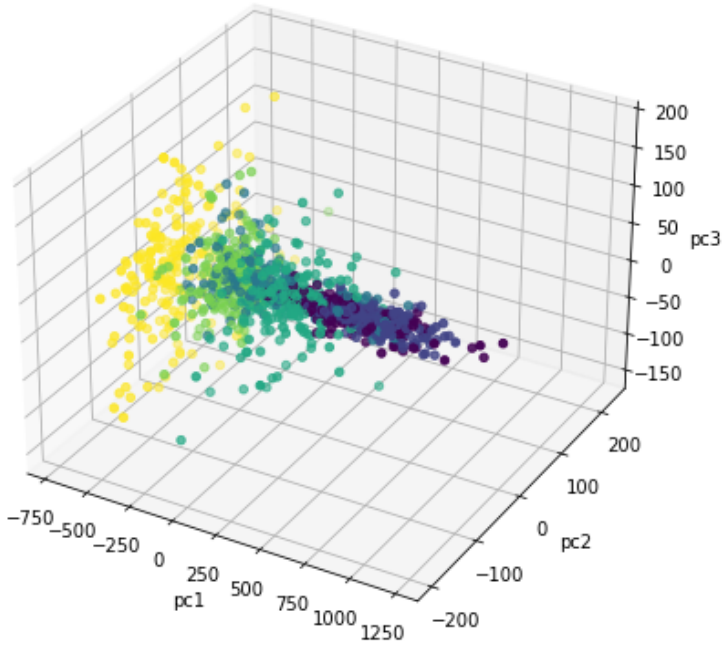


Figure 4.2: Projection sur 3 axes principaux

Nous présentons également ce qu'on appelle les eigen-images, qui sont juste les axes principaux de l'espace propre. Tous les images du dataset peuvent être reconstruites par une combinaison linéaire des eigen-images. Figure 4.3.

La raison pour laquelle le premier axe est si efficace et dominant est vraisemblablement la luminosité. Les images de la class F (Pure Ocean Waves) représentées en violet sont beaucoup plus lumineuses que celles de la class M (Low Wind Area) représentées en jaune. Figure 4.4

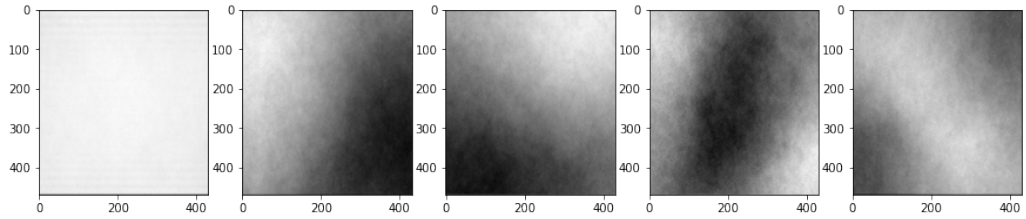
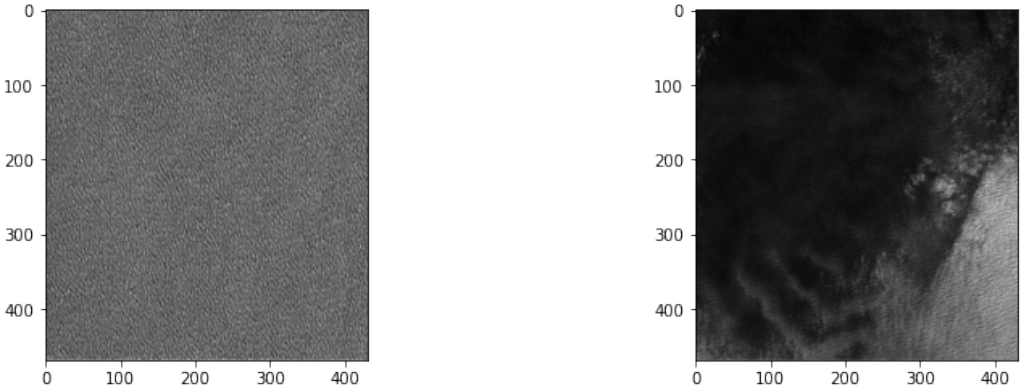


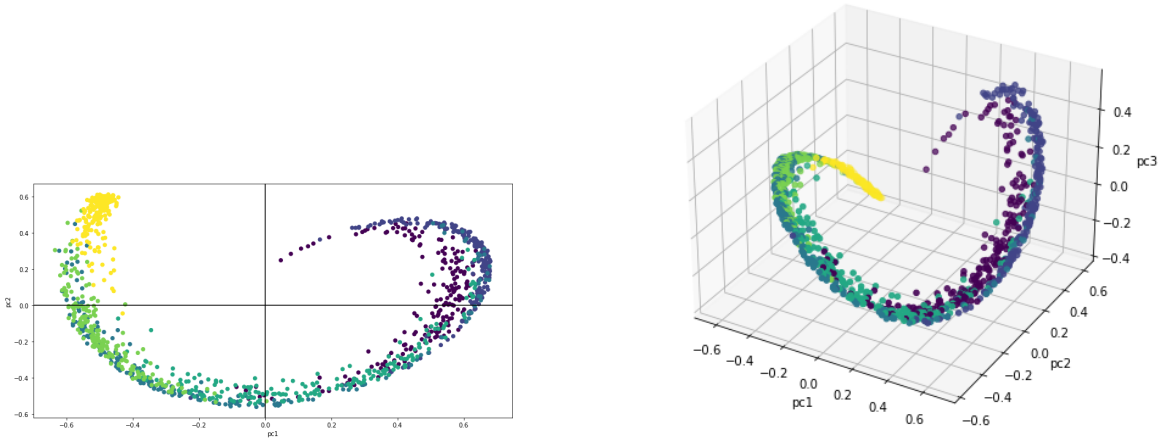
Figure 4.3: les 5 premiers eigen-images



(a) une image de la classe F : Pure Ocean Waves (b) une image de la classe M : Low Wind Area

Figure 4.4: Illustration de la différence de luminosité

La visualisation des résultats de l'ACP à noyau est plus estétique, mais il semble qu'elle n'est pas significativement meilleure que celle de l'ACP. Figure 4.5



(a) ACP à noyau : 2 axes principaux (b) ACP à noyau : 3 axes principaux

Figure 4.5: Visualisations pour l'ACP à noyau

4.2 AutoEncoder Convolutionnel

Dans cette partie, nous présentons les graphiques obtenus par l'Auto-encoder convolutionnel. Etant un réseau de neuronnes convolutionnel, il est plus adéquat pour le traitement des images. La dimension de l'espace latent (bottleneck) a été fixée à 3 pour pouvoir visualiser les données

par un nuage de point. Après 10 epochs, les représentations latentes ont été trouvées, voici les visualisations en 2D et 3D. Figure4.6 et Figure4.7.

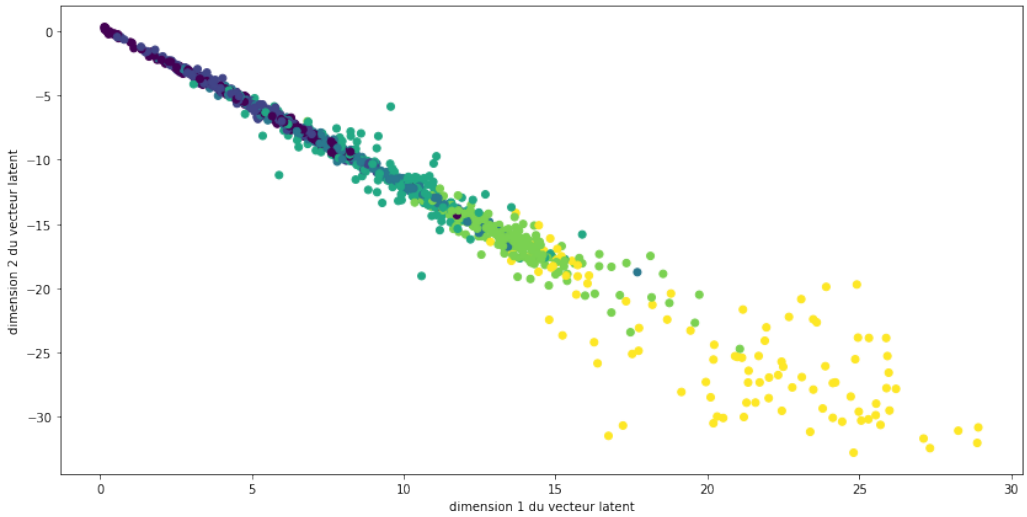


Figure 4.6: Visualisation des données dans l'espace réduit de dimension 2

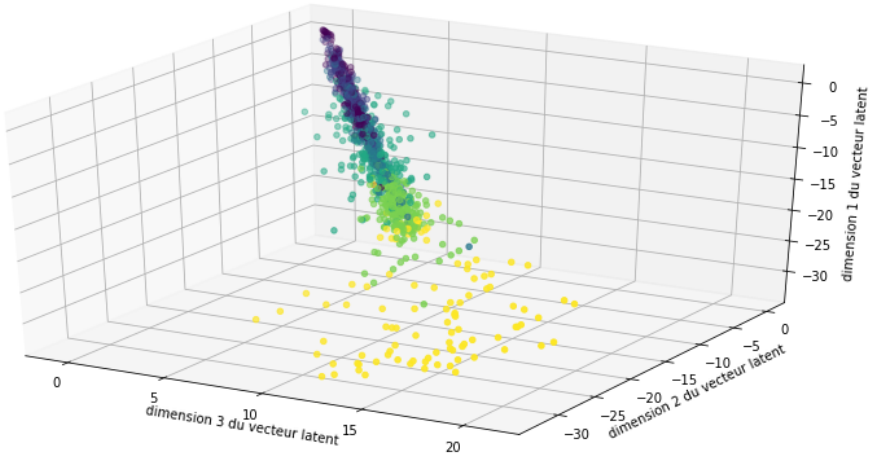


Figure 4.7: Visualisation des données dans l'espace réduit de dimension 3

Si nous regardons de près, l'axe sur lequel les données s'étirent le plus resemble beaucoup l'axe principal de l'ACP sauf qu'il n'est pas normalisé, et les directions qui lui sont orthogonales sont donc le deuxième et le troisième axe de l'ACP. C'est un résultat étonnant pour deux raisons suivantes :

1. Il est bien connu qu'un auto-encodeur avec un seul couche entièrement connecté, une fonction d'activation linéaire et la fonction de coût l'erreur moyenne quadratique est étroitement liée à l'ACP (Baldi & Hornik, 1989) (Bourlard & Kamp, 1988). Cependant, dans notre réseau de neuronnes, la fonction d'activation non linéaire (Relu) est ajouté presque à la fin de chaque couche, la fonction loss utilisée est l'entropie croisée et au nous avons implémenté un CNN beaucoup plue complexe qu'une seule couche.

2. les données ont juste été remis à l'échelle [0,1], mais n'ont pas subi de standarisation au niveau de pixel comme pour l'ACP.

A l'aide de ces visualisations, nous pouvons voir que le résultat de l'autoencoder convolutionnel n'est pas meilleur que celui de l'ACP. Pourtant, ce dernier est beaucoup moins exigent au niveau computationnel.

De plus, puisque nous fixons la dimension du *bottleneck* à 3, la reconstruction par le decoder n'est pas satisfaisante. Figure 4.8. La reconstruction des images avec le *bottleneck* à dimension $8 * 8 * 64$ est bien meilleure. Figure 4.9.

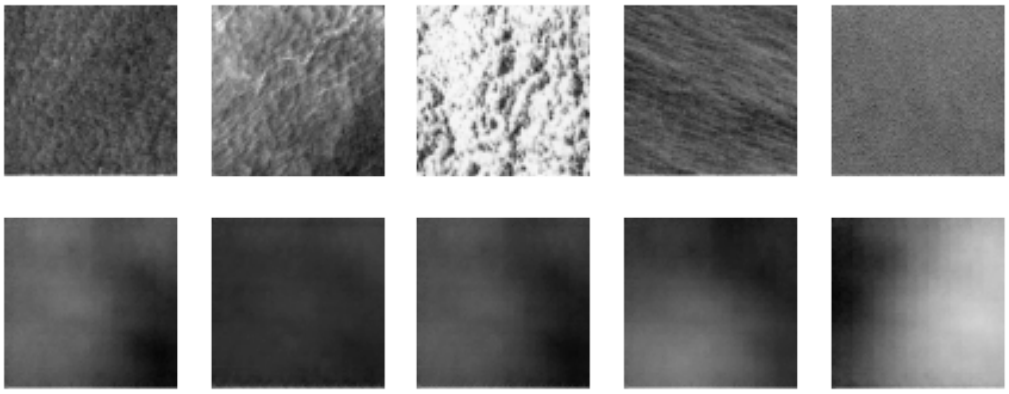


Figure 4.8: Reconstruction des images avec *bottleneck* à dimension 3

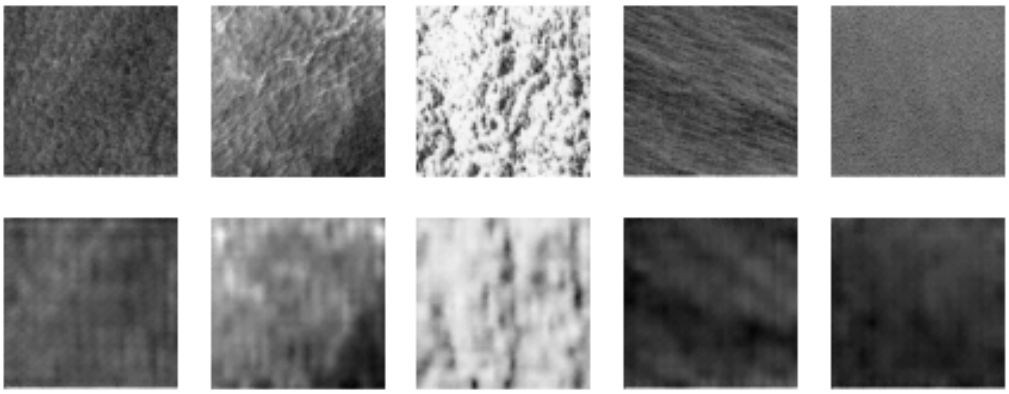
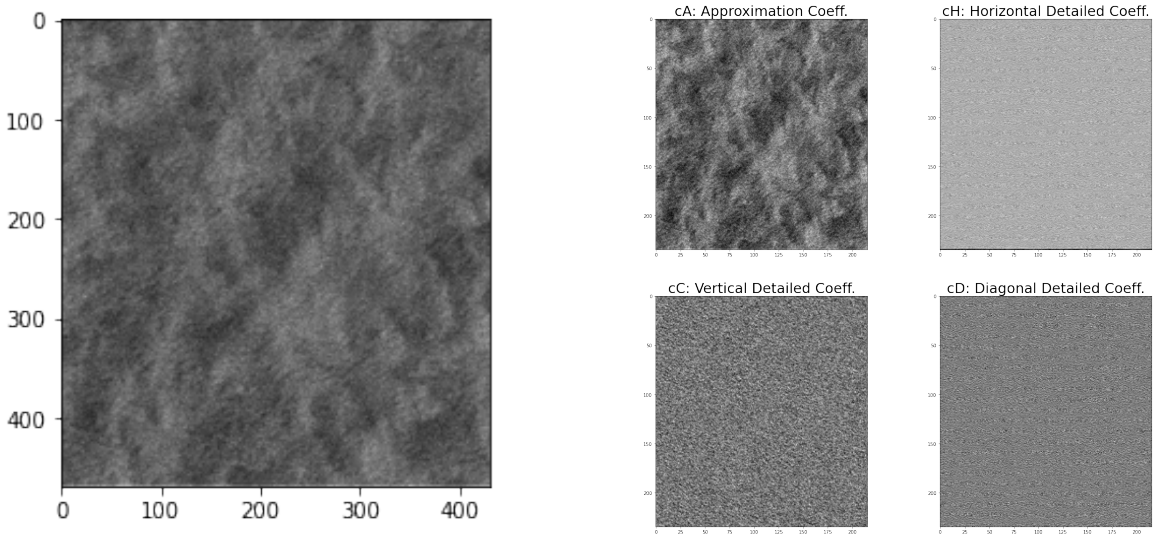


Figure 4.9: reconstruction des images avec *bottleneck* à dimension $8 * 8 * 64$

5 Transformation Wavelet

Dans l'espoir de pouvoir représenter les images dans un histogramme avec la variable qui est le coefficient d'une onde, j'ai tenté de mettre en place la décomposition wavelet. Cependent, il s'avère que pour une image en 2D qui représentent des données discrètes, la décomposition watelet discrète sert plutôt à la compression ou le prétraitement des images. Nous présentons ici donc la visualisation d'image individuelle après la transformation wavelet, vu que cela pourrait être utile pour mon projet d'alternance. Figure 5.1



(a) Image avant trasformation (b) Décomposition de l'image

Figure 5.1: Visualisation de l'image par transformation wavelet discrète

Le cA représente l'approximation de l'image avec moins de détails, cH, cC, cD représentent respectivement les détails horizontaux, verticaux et diagonaux de l'image obtenus par une filtre d'ondelette.

6 Conclusion

Les images étant un objet particulier, nous ne pouvons pas les visualiser directement ou après des analyses statitiques élémentaires (analyse bivariée entre autres). Le recours aux méthodes de réduction de dimension est nécessaire. Les images ainsi réduites peuvent être présentées par des nuages de points en 2D ou 3D. Ces visualisations permettent de mieux comprendre les comportements des données.

Ce projet en Data Visualisation m'a permit de me familiariser avce les outils, les packages pour visualiser les données et aussi de me sensibiliser sur l'importance de ces opérations qui sont souvent mises à côté dans les formations de machine learning.

Bibliographiques

Baldi, P., & Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, *2*(1), 53–58.

Bourlard, H., & Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, *59*(4), 291–294.

Wang Chen, T. P. S. J. L. N. E. G. F. R. V. D. C. B., Mouche Alexis. (2018). *Labeled sar imagery dataset of ten geophysical phenomena from sentinel-1 wave mode (tengeop-sarwv)* [Dataset]. FRANCE, USA. Retrieved from `https://www.seanoe.org/data/00456/56796/` doi: https://doi.org/10.17882/56796

Liste des figures