Non-Negative Matrix Factorization

Paul-Henri Icher, Nathan Leclercq, Samuel Carpentier

April 14, 2024

1 Introduction

La Factorisation en Matrices Non Négatives (NMF, pour Non-negative Matrix Factorization) est une technique de réduction de dimensionnalité et d'analyse de données.
Le but de cette approche est d'approximer une matrice par deux autres dont toutes les entrées sont non-négatives.
Cette propriété de non-négativité permet une bonne interprétabilité de la réduction, chaque composantes étant "positives" on n'a pas de calques "négatifs". Par exemple, si on travaille sur des données images, chaque classe correspond à un assemblage fini de composantes. Ce fait est illustré dans la suite du rapport. Chaque composante peut donc, en théorie, être plus facilement interprétable que pour une PCA (chaque composante peut être vu comme un élément simple et significatif et on peut la raccrocher facilement au contexte original).

La technique de NMF est utilisée dans de nombreux domaines comme par exemple :

- L'analyse de texte pour la détection de sujets, l'identification de motifs et de thèmes récurrents dans de grands ensembles de documents, facilitant ainsi la classification et le résumé automatique de textes. On illustrera cela dans la suite de ce document en comparant PCA et NMF pour la classification de texte.

- On peut aussi l'utiliser dans le traitement d'images, où nous décomposons les données en parties constitutives, ouvrant la voie à des applications telles que la reconnaissance faciale, la segmentation d'images, et l'amélioration de la qualité d'image. Nous réalisons une expérimentation dans la suite de ce document à partir du dataset MNIST.

- On trouve aussi des applications de cet algorithme en bioinformatique, dans l'analyse d'expressions génétiques, aidant les chercheurs à découvrir des patterns biologiques significatifs dans des données génomiques complexes [Brunet et al., 2004].

2 Description de l'algorithme

2.1 Description

La NMF est appliquée pour décomposer une matrice V de dimensions m×n en le produit de deux matrices W et H, où W est une matrice de dimensions m×k et H une matrice de dimensions k×n, avec k le nombre de composantes ou de caractéristiques cachées. On veut minimiser la différence entre V et le produit WH, sous la contrainte que W et H ne contiennent aucune valeur négative.

2.2 Pseudo-code

- Initialisation : On choisit des valeurs initiales pour W et H.

- Répéter jusqu'à convergence : Alternativement, ajuster H pour minimiser l'erreur de reconstruction avec W fixé, puis ajuster W de manière similaire avec H fixé.

- Critère d'arrêt : L'algorithme s'arrête lorsque la diminution de l'erreur de reconstruction entre deux itérations successives est inférieure à un seuil prédéfini ou après un nombre maximal d'itérations.

Critère d'arrêt : L'algorithme s'arrête lorsque la diminution de l'erreur de reconstruction entre deux itérations successives est inférieure à un seuil prédéfini ou après un nombre maximal d'itérations.

2.3 Initialisation

Pour l'initialisation, nous avons deux options : soit un choix aléatoire (risque de se perdre dans un minimum local); soit un choix basé sur les données (approche k-means ou svd pour avoir un départ plus proche de la valeur optimale). Voici les choix les plus courant pour l'initialisation de NMF :

- L'initialisation aléatoire est la plus simple mais potentiellement la moins efficace. Computationnellement parlant, c'est la plus directe.

- La SVD peut accélérer la convergence en fournissant un point de départ "informé", mais nécessite une modification pour garantir la non-négativité. Comme les matrices sont des matrices triangulaires, se baser sur les valeurs singulières semble être un bon choix d'initialisation.
- La NND SVD, en fournissant directement des matrices non-négatives, est souvent considérée comme offrant un bon équilibre entre simplicité et efficacité de convergence. Même argument que pour la SVD, mais sans nécessité de bricolage.

Comme les matrices doivent être non-négatives, on préferera NND SVD, une variante de SVD respectant la contrainte de non-négativité.

2.4 Choix du β-loss

Le choix du β-loss influence la mesure de distance ou de divergence utilisée pour calculer l'erreur de reconstruction entre V et WH. Le choix dépend souvent du type de données et de l'application :

$\beta = 2$: Correspond à l'erreur quadratique, adaptée aux données de comptage ou aux mesures de similarité euclidienne.

$$D(V, WH) = \frac{1}{2} \sum_{ij} (V_{ij} - (WH)_{ij})^2$$

$\beta = 1$: Produit une divergence de Kullback-Leibler, utile pour les données probabilistes ou les distributions.

$$D(V||WH) = \sum_{ij} \left(V_{ij} \log \frac{V_{ij}}{(WH)_{ij}} \right)$$

$\beta = 0$: Particulièrement utile pour les données audio ou pour les applications en traitement du signal où les différences relatives sont plus importantes que les différences absolues. On l'appelle aussi divergence Itakura-Saito.

$$D(V||WH) = \sum_{ij} \left(\frac{V_{ij}}{(WH)_{ij}} - \log \frac{V_{ij}}{(WH)_{ij}} - 1 \right)$$

Des valeurs de β entre 1 et 2 peuvent être explorées pour des compromis entre les caractéristiques de ces deux mesures. On prendra alors une composition de la distance de Frobenius (euclidienne) et de la distance de Kullback-Leibler, plus le nombre est proche de 2 et plus la proportion de distance de Frobenius est importante.

Dans ces formules, V_{ij} représente les éléments de la matrice originale V, et WH_{ij} les éléments du produit des matrices factorisées W et H. Ces différentes mesures de distance ou de divergence sont choisies en fonction de l'application et du type de données à traiter.

L'erreur quadratique est généralement adaptée aux données où la précision de chaque point de donnée est importante. La divergence de Kullback-Leibler convient mieux aux données probabilistes ou aux distributions.

2.5 Temps de calcul

Le temps de calcul dépend de la dimension des matrices, du nombre de composantes ciblées (car cela donne plus de possibilités de réduction) et des critères de convergence (plus les matrices WH et V doivent être proches, plus le nombre d'itérations de l'algorithme est élevé). Si on pose W et H matrices de dimensions m x k et k x n, on pourrait exprimer le temps de calcul par

$$T = c * (m * k + k * n) * I$$

où c correspondrait à une constante et I au nombre d'itérations de l'algorithme.

Voici un graphique illustrant cela, généré à partir du code fournit en annexe.

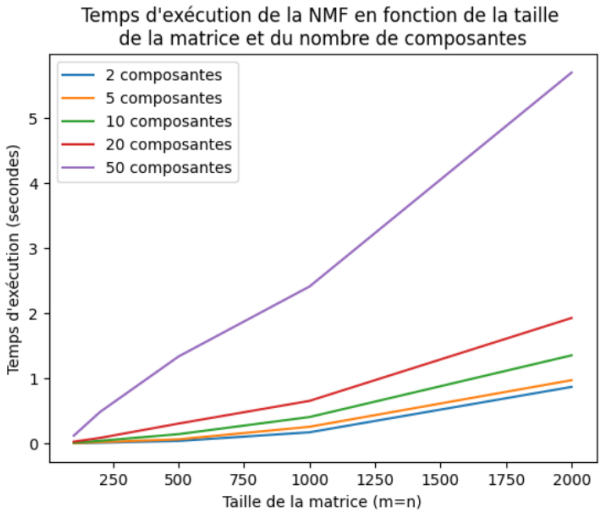

Figure 1: Temps d'execution de NMF

3 Application à un premier dataset : Minst.

Nous avons utilisé le datasets MINST composé d'images de chiffres en nuance de gris comprises entre 0 et 9 qui nous permet de comparer la méthode de

réduction de dimension NMF à PCA.
Nous avons tout d'abord essayé de visualiser les composantes extraites par les deux algorithmes et leurs utilités sur le premier chiffre du dataset. Puis nous avons étendu l'expérimentation sur les 1000 premières données pour comparer les performances de reconstruction des images à partir des composantes extraites des données originels ainsi qu'une analyse de durées d'exécution de chaque algorithme.

3.1 Extraction des composantes

Les images MINST sont en 28x28. Elles ont donc une dimentionnalité de 784. Dans nos expérimentations sur ces données, les algorithmes PCA et NMF cherchent à réduire la dimentionnalité à 50 dimensions.
Note : Dans les exemples de visualisations des composantes nous avons fixé la dimension à 15 pour avoir une meilleur représentation des processus.

3.1.1 Extraction par NMF

Nous initialisons la matrice de NMF aléatoirement ("random") et nous définissons la dimension de la décomposition à 50. Dans la suite du document nous feront exploration de meilleurs hyperparamètres pour cette tâche.

NMF est un modèle qui permet de décomposer notre matrice V en 2 matrices W (coefficients) et H (composantes).

Nous avons V (1000x784), W (1000x50) et H (50x784) avec :

- 1000 : Le nombre d'images originels
- 50 : Le nombre de caractéristiques extraites des images originels (Liaison entre W et H)
- 784 (pour H) : Chaque caractéristique est un vecteur de 784 dimensions (Une caractéristique peut donc est visualisée)

H contient l'ensemble des 50 caractéristiques sous la formes d'images 28x28. (Visualisation sur la figure 2.) W contient 1000 colonnes représentant l'ensemble des poids à appliquer sur les composantes pour recréer une approximation de l'image originale.

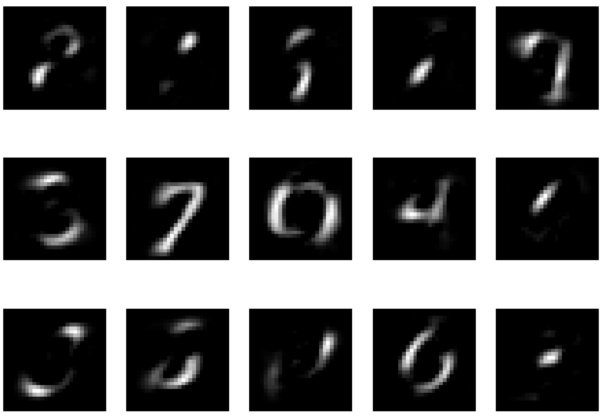

Figure 2: Exemple de visualisation de 15 composantes extraites par NMF sur MINST (1000 chiffres)

On observe que ces composantes ressemblent à des fragments de chiffres. Cela nous sera utile pour la reconstruction des images par la suite.

3.1.2 Précisions sur l'update de W et H

Après l'initialisation des matrices (dans notre cas aléatoirement), les mises à jour "alternées" de W et H sont effectuées pour minimiser la fonction de coût qui mesure la différence entre la matrice originale V et le produit WH. Les formules multiplicatives de mise à jour de H et de W dépendent de la fonction de loss à minimiser.

Le processus de mise à jour est généralement le suivant :

1. **Mise à Jour de W :**
   W est mise à jour en fixant H et en optimisant W pour minimiser la fonction de coût $\|V - WH\|$. Lors de cette mise à jour, H est considérée comme constante.

2. **Mise à Jour de H :**
   Après la mise à jour de W, H est mise à jour en fixant la nouvelle W et en optimisant H pour minimiser la même fonction de coût. Ici, W est considérée comme constante.

3.1.3 Extraction par PCA

Nous avons aussi essayé d'extraire les composantes apprises par une PCA. Cependant nous pouvons voir (Fig 3) que même si les composantes ressemblent vaguement aux "ombres" des chiffres, elles sont beaucoup moins humainement interprétables que les composantes extraites par la NMF car elles représentent des variations majeures entre les chiffres.

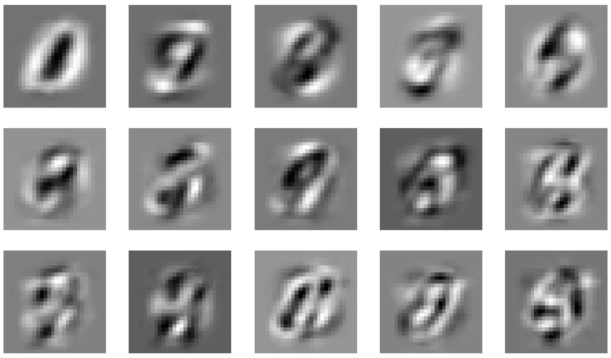

Figure 3: Exemple de visualisation de 15 composantes extraites par PCA sur MINST (1000 chiffres)

Notons que les composantes de PCA affichées sont en réalité les vecteurs propres extraits des données, représentant les plus grandes variabilité entre chaque chiffres.
Vu que nous avons fait une PCA avec 50 composantes. A partir des 1000 données originales la PCA extrait et stocke donc:

- Les 50 valeurs propres
- Les 50 vecteurs propres (Ex : Fig. 3)

A partir de ces valeurs propres, on calcule un "score" correspondant à la proportion de variance pour chaque valeur propre. Cela nous permet d'expliquer l'impact de chaque vecteur propre dans la reconstruction de l'image.

3.2 Reconstruction des données

3.2.1 Reconstruction par NMF

Le processus pour recréer une approximation de l'image originel se fait en combinant les composantes et en faisant varier leurs intensités en fonction des poids.

La figure ci-dessous (Fig. 4) représente bien cette reconstruction. Nous avons fait varier l'intensité d'affichage des composantes utilisées pour la reconstruction du chiffre 5 en fonction des poids présent dans la matrice W.

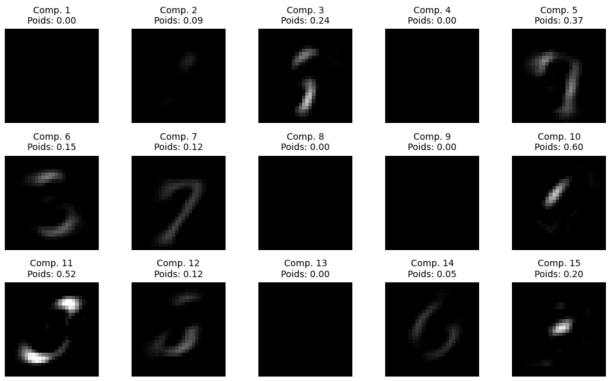

Figure 4: Exemple de visualisation de reconstruction du chiffre 5 à partir de 15 composantes extraites par NMF sur MINST (1000 chiffres). L'intensité de l'utilisation de la composante étant pondéré par le poids dans W

NMF est très intéressante car les composantes sont non-négatives, permettant une interprétation en termes de contribution additive des caractéristiques permettant ainsi une détection des patterns ou de features sous-jacentes dans les données.

3.2.2 Reconstruction par PCA

Soit S le vecteur des scores pour une image, qui contient les coefficients de projection de cette image sur les axes des composantes principales, et soit P la matrice contenant les 50 vecteurs propres (où chaque colonne correspond à un vecteur propre). La reconstruction R de l'image originale à partir de ses composantes réduites est donnée par la formule :

$$R = S \times P^T$$

où P^T est la transposée de P, avec des dimensions de 784x50. Le produit matriciel $S \times P^T$ résulte en un vecteur de 784 éléments qui représente l'image reconstruite.

3.2.3 Comparaison entre PCA et NMF

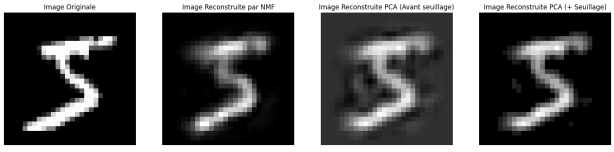

Figure 5: Comparaison de reconstruction de la première image de MINST en ayant utilisé NMF et PCA

L'association des différentes composantes selon les poids prévu dans W permettent donc d'obtenir

une reconstruction approximé la première image du dataset. Les deux algorithmes aproxime le 5, cependant on peut voir pour la PCA que le fond de l'image reconstruite est un peu grisé, ce qui nous éloigne de notre image d'origine. La plage des pixels sur l'image d'origine étant compris entre 0 et 1, pour luter contre le phénomène nous avons donc décidé de définir un seuil à 0.1. (Fig 5) Ce qui supprime cette couleur grisâtre et améliorera possiblement la qualité de la reconstruction.

3.3 Hyperparamètre tunning et comparaison des performances

Afin d'optimiser les algorithmes nous utilisons une gridsearch en faisant varier les hyperparamètres tel que :

- Le nombre de composantes à extraire des données
- Pour NMF nous faisons varier l'initialisation des matrices
- Pour NMF, le choix du beta (Cité dans 2.4)

Pour comparer la qualité des reconstruction nous utilisons la moyenne des MSE(Mean Squared Error) entre les images reconstruites et les images d'origines.

Notons que pour notre cas nous ne pouvons pas utiliser le solver 'cd' avec kullback-leiber. Nous utiliseront donc le solver 'mu'. Nous ne pouvons pas non plus utiliser un beta <= 0 quand la matrice d'origine (V) contient des zeros, donc nous n'utiliseront pas itakura-saito. Aussi, le solveur de mise à jour multiplicative ('mu') ne peut pas mettre à jour les zéros présents dans l'initialisation, et conduit donc à de moins bons résultats lorsqu'il est utilisé conjointement avec init='nndsvd'. Nous n'utiliseront donc pas cet initialisation et préféreront nndsvda et nndsvdar à la place. Nous fixons aussi le nombre d'itérations max pour NMF à 200 afin de limiter le temps d'exécution excessif. Nous pourrions augmenter cette valeur pour essayer d'atteindre la convergence.

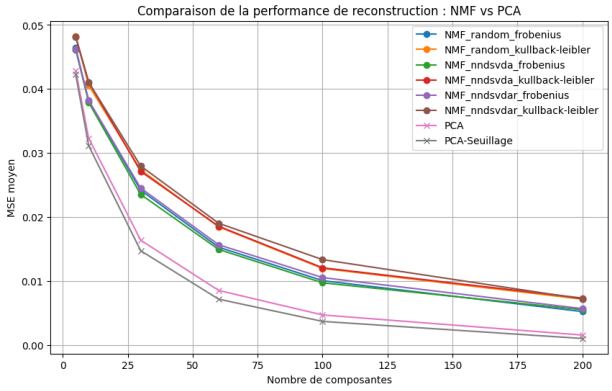

Figure 6: Comparaison de performances utilisant le MSE moyen en fonction du nombre de composantes extraites par la PCA, NMF et leurs variantes

PCA avec seuillage semble être la meilleure approximation moyenne des images d'origine. L'algorithme semble globalement meilleure que NMF pour la reconstruction d'images à partir des composantes extraites. Notons que nous voyons que l'approximation des chiffres est meilleure en fonction de l'augmentation du nombre de composantes extraites.

La réduction de dimension est moins un objectif en soi que le moyen d'optimiser d'autres processus comme la visualisation de données complexes, l'accélération des algorithmes ou encore l'apprentissage automatique. Dans cette optique, notre étude souligne que, malgré l'efficacité de la PCA avec seuillage pour la reconstruction d'images, les caractéristiques extraites par NMF offrent des avantages en termes d'interprétabilité et de pertinence pour des applications spécifiques. Le choix entre ces méthodes ne devrait donc pas se baser uniquement sur la fidélité de reconstruction, mais aussi prendre en considération les exigences de l'application cible.

3.4 Durées d'exécution

Afin de faire une comparaison des rapidités d'exécutions nous avons enregistré les temps d'exécutions pour chaque algorithmes

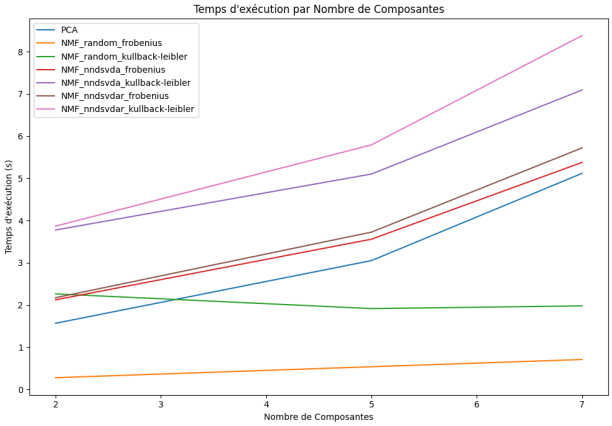

Figure 7: Comparaison de performances utilisant le MSE moyen en fonction du nombre de composantes extraites par la PCA, NMF et leurs variantes

PCA tend à être assez rapide par rapport à NMF en raison de sa simplicité algorithmique. PCA implique principalement des opérations de décomposition en valeurs singulières qui sont bien optimisées et moins coûteuses en termes de calcul comparativement aux itérations nécessaires pour converger vers une solution en NMF.

La figure 7 nous montre bien que les différentes initialisations et fonctions de coût en NMF (random, nndsvda, nndsvdar; frobenius, kullback-leibler) influencent le temps de convergence.

Selon nos observations l'algorithme le plus rapide semble être le nmf_random_frobenius. Notons que le nombre d'itérations pour atteindre la convergence est exponentielle en fonction du nombre de composantes à calculer, le temps d'exécution l'est donc aussi.

4 Application à un second dataset : données textuelles.

A présent, nous souhaitons comparer la méthode de réduction de dimension de NMF avec la PCA pour un dataset comprenant des données textuelles. Du fait de la taille des données et du nombre de features une fois encodées, nous utiliserons ici uniquement la version NMF de sklearn avec les paramètres de bases (matrice initialisées au hasard et $\beta = 2$, ou distance de frobenius).

Nous allons nous appuyer sur "fetch_20newsgroups", une base de données disponible via sklearn, qui regroupe des billets de forums sur 20 sujets différents :

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Figure 8: Tableau regroupant les 20 sujets du dataset en 6 thématiques globales

L'intérêt d'utiliser un dataset déjà labelisé avec différentes catégories est de pouvoir les comparer avec les dimensions obtenues via nos deux techniques. Plus précisément, comme le montre le tableau ci-dessus, les 20 sujets peuvent être regroupés en 6 grandes thématiques : informatique, sport, science, annonce de ventes, politique et religion.

Au total, 11314 billets ont été enregistrés dans le dataset. Une fois convertis en vecteurs, chaque billet (ou ligne de notre dataset) est décrit par 130107 features. Elles correspondent ici à 130107 tokens (des unigrammes), et sur le rapport entre la fréquence d'apparition des tokens au sein de chaque billet et leur fréquence d'apparition sur l'ensemble du document pour les décrire.

Pour des raisons de puissance de calcul, nous allons nous limiter ici à 2 sujets (alt.atheism, et talk.politics.guns) correspondant aux thématiques "politique" et "religion". Notre dataset final comprend ici 1026 samples décrits par 23363 features.

Sur la base de ce regroupement a priori en 2 thématiques, on s'attend donc à ce que, dans l'idéal, on observe le meilleur "trade-off" entre le nombre de dimensions retenues et quantité d'informations qui soient des mots qui soient représentatifs de ces 2 thématiques.

Pour comparer nos deux méthodes, nous allons utiliser la même métrique que précédemment, à savoir la distance euclidienne entre la matrice reconstruite à l'aide d'une méthode et le dataset encodé initialement. Nous nous limiterons à 20 dimensions réduites pour chacun de nos modèles en raison de nos puissances de calculs. Les résultats sont regroupés dans la figure suivante :

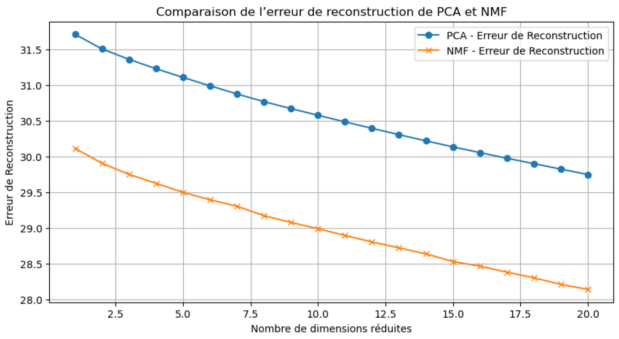

Figure 9: Erreur de reconstruction entre la PCA et la NMF

On remarque ici une évolution conjointe/de même allure de l'erreur de reconstruction entre les deux méthodes. Cependant, la NMF obtient de meilleurs résultats sur notre dataset : les valeurs d'erreur de reconstruction sont plus petites qu'avec la PCA. Ces résultats semble donc mettre en avant le fait que la méthode NMF semble davantage adaptée à ce type de données.

On remarquerons également que les deux courbes ont un comportement linéaire. Cela nous indique qu'il semble difficile, pour le nombre de dimensions réduites observées, de synthétiser les données autour d'un nombre réduit de dimensions équivalent au nombre de thématiques ou sujet traités par les chaînes de caractères encodées. Dit autrement, la part d'information semble diffuse et également répartie entre les dimensions.

Cette observation est confirmée par l'évolution de la part de variance expliquée obtenue avec la PCA :

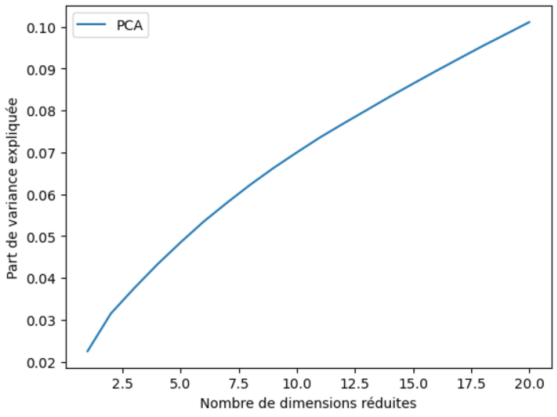

Figure 10: Évaluation de la part cumulée de variance expliquée en fonction du nombre de dimension retenues

Notre hypothèse initiale, qui supposait que l'on puisse retrouver les différentes thématiques des différents messages dans notre réduction de dimension, semble ici invalidée. En effet, on s'attendait dans ce cas à ce que l'on observe, pour notre dataset regroupant des messages ayant trait à des thématiques religieuses ou politiques, une réduction assez nette de l'erreur de reconstruction dès les premières dimensions. En effet, on pouvait supposer qu'une grande partie de l'information sémantique soit redondante entre les mots et que des dimensions indépendantes, comme avec PCA, puisse facilement les distinguer.

Pour expliquer nos résultats, on peut considérer que le nombre important de features suite à notre encodage semble ici limiter la capacité de nos réduction. Le format des données et l'encodage choisi peut aussi jouer un rôle important : notre embedding se base en effet sur un calcul de fréquence des unigrammes dans nos chaînes de caractères. Plus le nombre d'unigrammes est important, plus la distance entre les points (ici les chaînes de caractères) sera important : cela illustre concrètement le problème de "malédiction des grandes dimensions".

En terme d'analyses complémentaires, on pourrait proposer de malgré tout se baser sur nos connaissances a priori sur les données. Il serait par exemple intéressant de bénéficier d'une puissance de calcul supplémentaire pour réaliser une PCA et une NMF dont le nombre de dimension est cohérent avec le lemme de Johnson Lindenstrauss. Pour interpréter ces données, on pourrait ensuite afficher une matrice de corrélation entre nos x dimensions obtenues et les features de bases, et observer si l'on observe deux patterns de corrélations à même de distinguer les features en au moins deux catégories : celles associées au champs lexical de la religion pour l'une et au champ lexical de la politique pour l'autre.

5 Conclusion

Ce projet nous a permis d'explorer et de comparer deux techniques de réduction de dimension : la Factorisation en Matrices Non Négatives (NMF) et l'Analyse en Composantes Principales (PCA). Nous avons testé ces méthodes sur des images et des textes pour voir laquelle était la plus adaptée selon les cas d'usage.

Pour les images, nous avons relevé qu'en termes de vitesse de calcul, la PCA a l'avantage grâce à des algorithmes d'optimisation bien établis qui rendent son exécution rapide, surtout comparée à la NMF qui nécessite des itérations plus complexes et donc plus

de temps pour converger. Pour ce type de dataset, la NMF se distingue par sa capacité à produire des composantes toutes positives, ce qui facilite grandement l'interprétation des résultats. En effet, chaque composante pouvait ici être vue comme un élément constructif clair, distinctif des images, et permettait de mieux saisir leurs éléments constitutifs. La PCA, elle, était souvent plus performante pour reconstruire précisément ces données. C'est grâce à sa capacité à capturer les directions qui maximisent la variance. Les composantes de PCA peuvent inclure des valeurs négatives, ce qui peut compliquer leur interprétation, mais elles restent très utiles pour des analyses rapides et efficaces où la reconstruction fidèle des données est prioritaire.

Pour les données textuelles, nous avons observé l'inverse. La NMF permettait en effet d'obtenir des reconstructions avec moins d'erreurs comparé au dataset encodé. Notre hypothèse de départ, consistant à supposer que les dimensions réduites obtenues permettraient de retrouver en partie les thématiques et sujets labellisés dans notre dataset de base n'a pas pu être pleinement explorée du fait de la limitation de notre puissance de calcul. Malgré tout, l'allure des courbes de l'erreur de reconstitution de la NMF et de la PCA semble indiquer que ces dernières suivent une trajectoire linéaire : une réduction dont les dimensions rendraient compte des différentes thématiques est donc difficilement envisageable. On pourrait en revanche proposer d'analyser les pattern de corrélations entre les features et les dimensions réduites pour étayer cette idée.

En conclusion, le choix entre NMF et PCA dépendra fortement du contexte d'application. La NMF est préférable quand l'interprétabilité des composantes est essentielle, tandis que la PCA est recommandée pour des situations nécessitant une reconstruction précise et rapide des données. De plus, la décomposition en valeurs propres de la PCA permet de "quantifier" la part d'information (dans ce cas la part de variance expliquée) fournit par le modèle et ses différentes dimensions. La NMF ne fournit pas une telle métrique nous permettant d'apprécier la "contribution" de chaque dimension à la recomposition de nos données. La prise en compte du type de dataset considéré et de son encodage semble, enfin, déterminant pour le choix de la méthode utilsée.