Rapport du TP3 du projet AD/CS

Ouhou Houda - Bouisse Yassir - El Jai Reda

Département Sciences du Numérique - Première année
2019-2020

Contents

List of Figures

Introduction :

Ce TP concrétise toutes les connaissances acquises lors des TPs précedents pour les exploiter dans la reconnaissance faciale. Il exploite notammement l'ACP appliquée à des matrices dont les lignes correspondent aux images de personnes vectorisées. La très grande taille de ce jeu de données va nous mener à faire de choix judicieux sur le plan algorithmique et informatique pour pouvoir traiter ces données et les classifier.

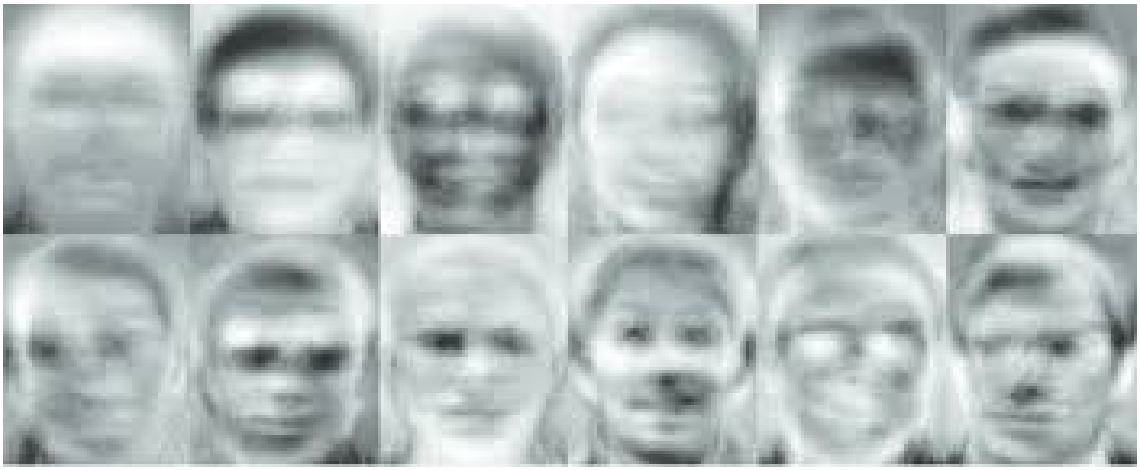

Figure 1: Eigenfaces

Exercice 1 : Analyse en composantes principales

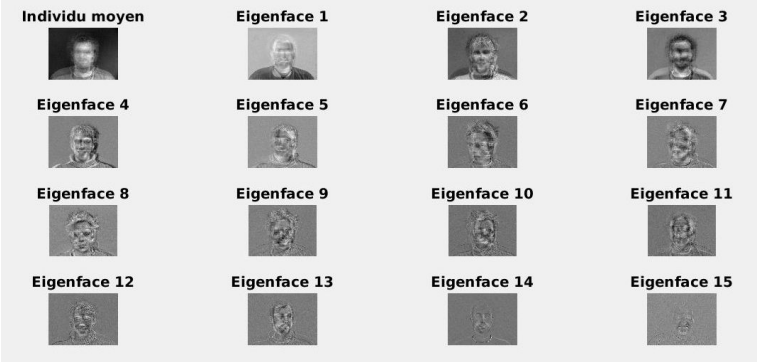

Figure 2: Eigenfaces et individu moyen

Grâce à l'ACP appliquée à la matrice Σ_2 qui a les même valeurs propres de Σ mais qui est de taille bien plus inférieure, nous avons pu retrouver les eigenfaces.

Exercice 2 : Projection sur les eigenfaces

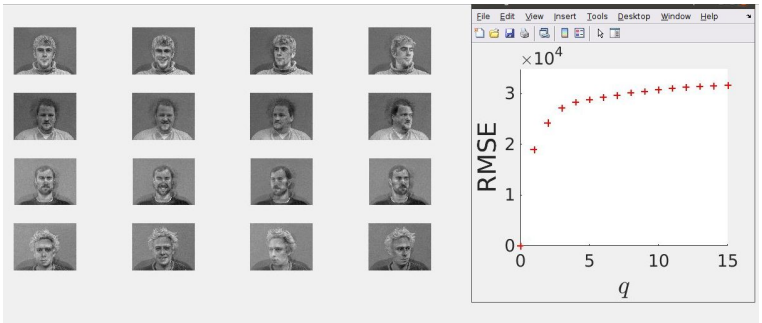

Figure 3: Images recontruites à partir des q première composantes et RMSE en fonction de q

Nous avons projeté notre jeu de données sur les q premières composantes principales, puis nous avons reconstruit les images originales à partir de ces projections.Nous remarquons en traçant le RMSE qui traduit la différence entre les images réelles et reconstruites que plus q est grand, plus l'image recontruite se rapproche de l'originale.

Exercice 3 : Application à la reconnaissance des visages

Nous avons défini donnees1.mat au lieu de donnees.mat afin d'étudier tous les 37 individus.

Questions sur la reconnaissance des visages

Question 4:

En s'inspirant de la méthode kppv vu en TP4, on essaie de trouver les 3 plus proches voisins d'un individu parmi les 37 individus générés avec la fonction randi(37). La nouvelle fonction de kppv renvoie, en plus de l'individu reconnu, une matrice contenant les données nouvelles de la requête, la distance entre le résultat de la requête et les images-test, et kppv : les indices de k plus proches voisins. Si cette distance est inférieur au seuil souhaité, (s = 0,5), on la garde et on affiche les images comme le montre la figure 3. Sinon, on relance le programme en cas d'échec. La figure 5 illuste un cas d'échec.

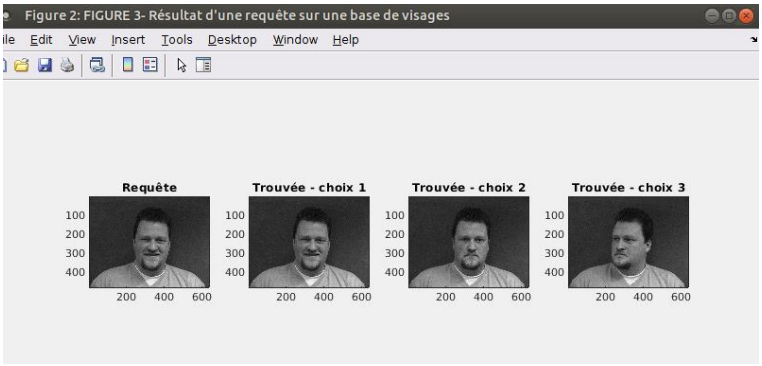

Figure 4: Exemple d'execution réussie du programme

Figure 5: Exemple d'échec de l'execution de l'exercice 3.

Question 5 :

La matrice de confusion nous permet de calculer l'erreur commis lorsqu'on confond la classe réelle des données et la classe prédite à la sortie du classifieur. Dans notre cas, l'erreur à la sortie est égale à 0

```
taux_erreur =

     0
```

Figure 6: Erreur à la sortie du classifieur

Discussion

Question 6 :

Vue la grande taille des données utilisées, on peut penser tout d'abord à utiliser la matrice Σ_2 au lieu de Σ pour travailler sur une matrice de taille inférieure et qui a les mêmes valeurs propres. Ensuite si la taille de la matrice reste relativement petite, on se propose de choisir l'algorithme *subspace_iter_v*3 vu lors du TP2 qui est la plus rapide et la plus efficace des méthodes itératives vu jusqu'à présent. Cependant si notre base de données contient des millions de visages à detecter, on se retrouve tout de même avec de grandes matrices et on prévilégiera plutôt une implantation de Cholesky qui s'avère plus rapide pour de très grandes matrices.

Question 7 :

Comme ceci a été vu lors du TP précedent, pour des matrices de très grandes tailles le solveur dsyev qui implante Cholesky est plus performant que *subspace_iter_v*3. Choisir une implantation de Cholesky semble plus judicieux donc pour traiter des données de telle taille.

Question 8 :

Nous avons défini exercice.m qui crée une matrice de données associés à donneesCouleurs que nous avons stocké dans exercice.mat qui sera utilisé dans exercice3_couleur.m . Comme le montre les figures 7 et 8 obtenues après l'excution du script, en introduisant la couleur, on perd en efficacité de détection de l'individu. D'autres individus avec la même couleur de cheveux et d'habits faussent le résultat. Aisni, On perd en efficacité spectrale lorsqu'on prend en considération les couleurs des images comme le montre le taux d'erreur 0.8400 affiché sur la figure 9.

Figure 7: Exemple 1 : Résultat de la requête avec des images en couleur

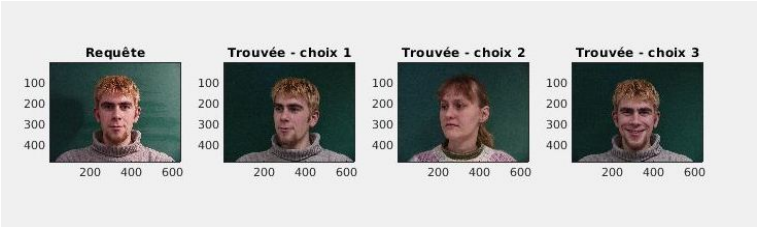

Figure 8: Exemple 2 : Résultat de la requête avec des images en couleur

```
>> exercice3_couleur

taux_erreur =

    0.8400
```

Figure 9: Taux d'erreur avec des imges en couleur

Conclusion

L'ACP est un outil efficace pour la reconnaissance faciale. Afin de retrouver les couples propres nécessaires à la classification, on privilégie un algorithme implémentant la décompostion de Cholesky quand le nombre d'individu est très grand. On a aussi remarqué que l'efficacité spectrale des photos en couleur était plus petite que celle des photos en niveau de gris et que l'exploitation des photos en couleur menaient à des erreurs en mélangeant des individus ayant les mêmes couleurs de cheveux et d'habits.