Le génie pour l'industrie

SYS828 - Systèmes biométriques

Rapport de laboratoire

À Montréal, le 15/06/2023

Professeur : Rafael M. O. Cruz
Auxiliaire de laboratoire : Bilal Alchalabi
Étudiant : Victor Rios

Sommaire

I) Introduction

Mise en situation

La reconnaissance de visages est un domaine de recherche en constante évolution, et la conception d'un système d'identification de visages robuste à partir d'images statiques présente des défis passionnants. Nous étudierons les divers aspects liés à cette conception, en explorant les classificateurs utilisés, l'impact des algorithmes de réduction de dimensionnalité sur l'apprentissage et la performance des classificateurs, aussi bien individuellement qu'en combinaison.

L'utilisation de la reconnaissance faciale est devenue courante dans de nombreux domaines, tels que la sécurité, l'authentification d'utilisateurs, la surveillance, et bien d'autres encore. Cependant, ces applications nécessitent des systèmes d'identification de visages capables de gérer des situations variées, telles que des changements d'expression, d'éclairage, de pose et d'apparence. Notre objectif est de concevoir un système capable de relever ces défis et de fournir des résultats fiables et précis.

Figure 1 : Identification de visages (dreamstime, 2023)

Problématique

Le principal défi de la conception d'un système d'identification de visages réside dans la complexité des caractéristiques du visage et dans la variabilité des conditions dans lesquelles les images peuvent être capturées. Nous devons déterminer quels classificateurs sont les plus adaptés pour traiter ces variations et évaluer l'efficacité de différents algorithmes de réduction de dimensionnalité pour améliorer l'apprentissage et la performance du système.

Objectifs

Nos objectifs lors de cette étude de conception sont les suivants :

- Évaluer plusieurs classificateurs couramment utilisés pour la reconnaissance faciale, tels que les Support Vector Machine (SVM), les k-Nearest Neighbors (k-NN) et les réseaux de neurones convolutifs (CNN).

- Examiner les performances de chaque classificateur en utilisant différents algorithmes de réduction de dimensionnalité, tels que Principal Component Analysis (PCA) et Linear Discriminant Analysis (LDA).

- Comprendre comment la combinaison de différents classificateurs et algorithmes de réduction de dimensionnalité peut améliorer la précision et la robustesse du système d'identification de visages.

Structure du document

Ce rapport aborde la conception d'un système d'identification de visages robuste à partir d'images statiques. Dans un premier temps, il examine divers classificateurs utilisés, l'impact des algorithmes de réduction de dimensionnalité sur l'apprentissage et la performance des classificateurs, ainsi que la combinaison de ces approches. Dans une seconde partie, on s'intéresse à la méthodologie qui décrit les protocoles expérimentaux, la base de données et les indicateurs de performance. Ensuite, les résultats et la discussion comparent les approches analytiquement et quantitativement. En conclusion, des recommandations sont formulées pour améliorer le système d'identification de visages. Enfin, un approfondissement par l'étude des réseaux siamois sera étudié pour cette tâche.

II) Synthèse des approches

Techniques de compression de données

Les techniques de compression de données sont utilisées pour réduire la taille des images ou des caractéristiques des visages tout en préservant les informations essentielles nécessaires à l'identification. Elles permettent de réduire les exigences en matière de stockage et de traitement (dimensionnalité), tout en maintenant des performances d'identification des visages.

Techniques d'extraction de caractéristiques

L'objectif de l'extraction et de la sélection de caractéristiques est d'identifier les caractéristiques importantes pour la discrimination entre classes. En effet, elle est essentielle pour représenter les visages de manière informative et discriminante afin de faciliter leur identification. Après avoir choisi le meilleur ensemble de caractéristiques, il s'agit de réduire la dimensionnalité de l'ensemble des caractéristiques en trouvant un nouvel ensemble, plus petit que l'ensemble original, contenant la plupart de l'information.
La performance des classificateurs n'augmente pas indéfiniment avec la taille du vecteur de caractéristiques. Après avoir augmenté en fonction du nombre de caractéristiques, l'efficacité atteint un plateau et quelquefois se met à décroître au-delà d'un certain nombre de caractéristiques. Ce phénomène est appelé malédiction de la dimensionnalité. En plus, le coût (temps + mémoire) de la classification augmente avec la taille de ce vecteur.

Les méthodes d'extraction de caractéristiques à partir de l'intensité d'une image du visage peuvent être divisées en trois catégories : les méthodes globales (exploitant toute la région visage), les méthodes locales (exploitant des caractéristiques locales: yeux, nez, etc.), et les méthodes hybrides. On se limite principalement aux méthodes globales où l'image du visage au complet peut être vue comme un vecteur dans un espace ayant autant de dimensions que de pixels dans l'image, et représentant la variation d'intensité (en niveau de gris).

- **PCA (Principal Component Analysis)** :

PCA est une méthode de réduction de dimensionnalité non supervisée qui permet de transformer un ensemble de variables corrélées en un nouvel ensemble de variables non corrélées appelées "composantes principales". Ces composantes principales sont ordonnées par ordre d'importance, où la première composante principale capture le maximum de variance des données et ainsi de suite. En réduisant le nombre de dimensions, PCA permet de simplifier la représentation des

données tout en conservant au maximum l'information essentielle. Pour la reconnaissance faciale, PCA est utilisée pour extraire les caractéristiques les plus importantes des images de visages et réduire la redondance dans les données. De plus, les composantes principales (eigenfaces) représentent les directions dans lesquelles les données sont les plus dispersées, et elles peuvent être utilisées comme des caractéristiques discriminantes. Ainsi, PCA permet de **réduire la dimensionnalité** et d'**extraire les caractéristiques**.

Figure 2 : Eigenfaces et algorithme PCA (pawangfg, 2021)

- **LDA (Linear Discriminant Analysis) :**

LDA est également une méthode de réduction de dimensionnalité. Son objectif est de trouver un sous-espace dans lequel les classes de visages sont mieux séparées les unes des autres. Cette technique vise à maximiser la distance entre les moyennes des classes et à minimiser la variance intra-classe. Cela permet de trouver des caractéristiques discriminantes qui sont plus utiles pour la tâche de classification, telle que la reconnaissance de visages, où les classes représentent différentes identités. LDA est particulièrement utilisée dans des tâches de classification. Elle est particulièrement utile lorsque les classes ne sont pas linéairement séparables dans l'espace d'origine, car elle cherche à trouver un sous-espace avec lequel elles le seront. Ainsi, LDA permet de **réduire la dimensionnalité** et **séparer les classes** d'un problème de manière discriminante.

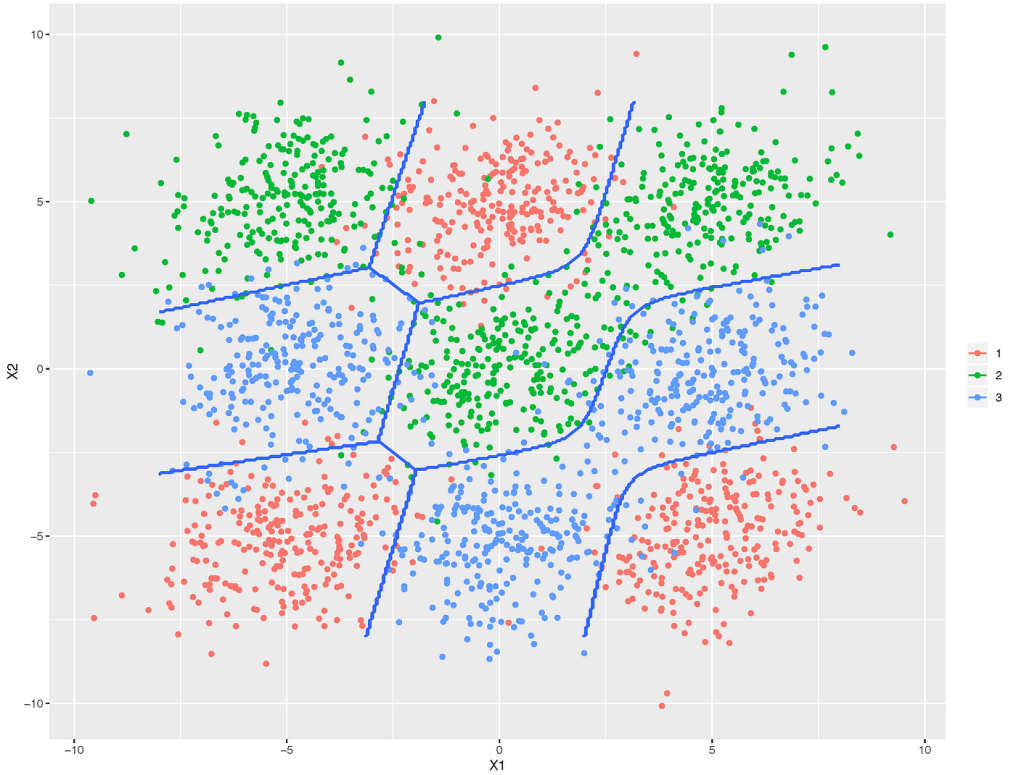


Figure 3 : Algorithme PCA (YANG Xiaozhou, 2020)

- **Réseaux siamois :**

Les réseaux de neurones siamois sont utilisés pour apprendre une représentation dite "embeddings" des visages, où les visages de la même personne sont rapprochés dans l'espace, tandis que les visages de personnes différentes sont éloignés les uns des autres. Cela facilite la comparaison des embeddings pour l'identification des visages. On reviendra sur ce modèle dans la partie d'approfondissement.

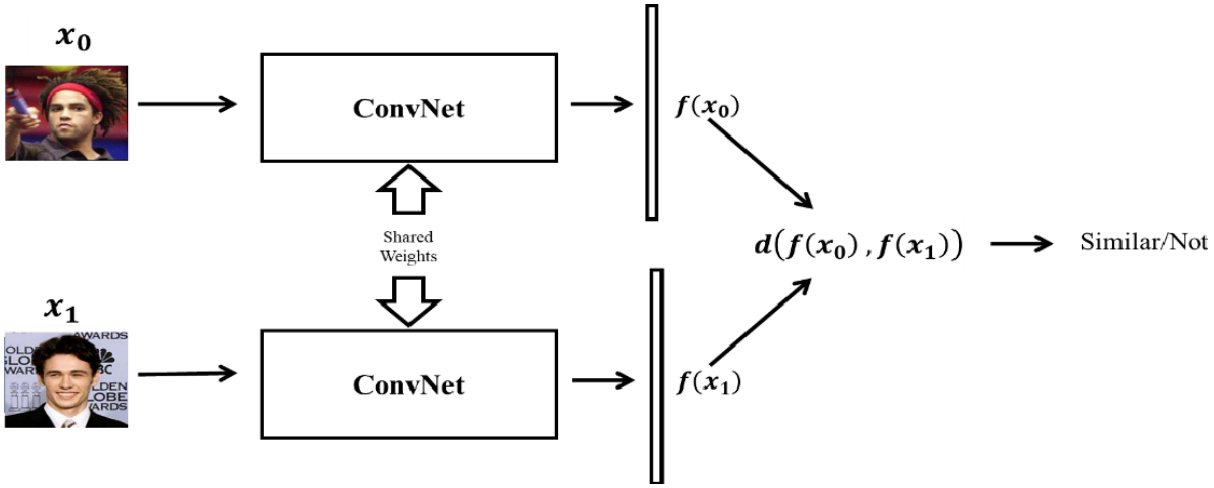


Figure 4 : Réseaux siamois pour l'identification de visages (Mohsen Heidari, Kazim Fouladi-Ghaleh, 2020)

Techniques de classification

La conception d'un système d'identification de visages robuste repose sur l'utilisation de diverses techniques de classification. Ces méthodes jouent un rôle essentiel dans l'assignation des visages aux bonnes classes, facilitant ainsi leur reconnaissance.

- **Support Vector Machine (SVM) :**

Les SVM sont des algorithmes de classification populaires qui cherchent à trouver l'hyperplan optimal qui sépare les données en différentes classes. Ils sont efficaces pour traiter des problèmes de classification binaire et peuvent être étendus pour gérer des tâches multi-classes. En effet, en formulation duale, le SVM peut être étendu avec soft margin pour modéliser le chevauchement ou le bruit dans les données. L'utilisation de SVM peut également résoudre des problèmes non linéaires en projetant les patrons dans un espace de plus grande dimensionnalité à l'aide de fonction (kernel) noyau (Gaussien ou RBF, polynomial…) Une fois que l'hyperplan optimal est trouvé, les SVM utilisent une fonction de décision pour classer de nouveaux échantillons. Cette fonction attribue à un nouvel échantillon une classe en fonction de sa position par rapport à l'hyperplan.

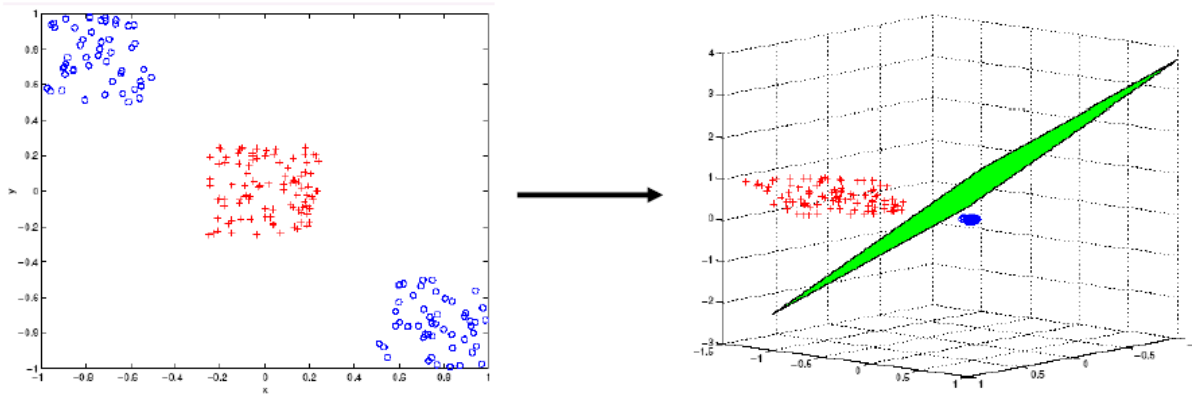
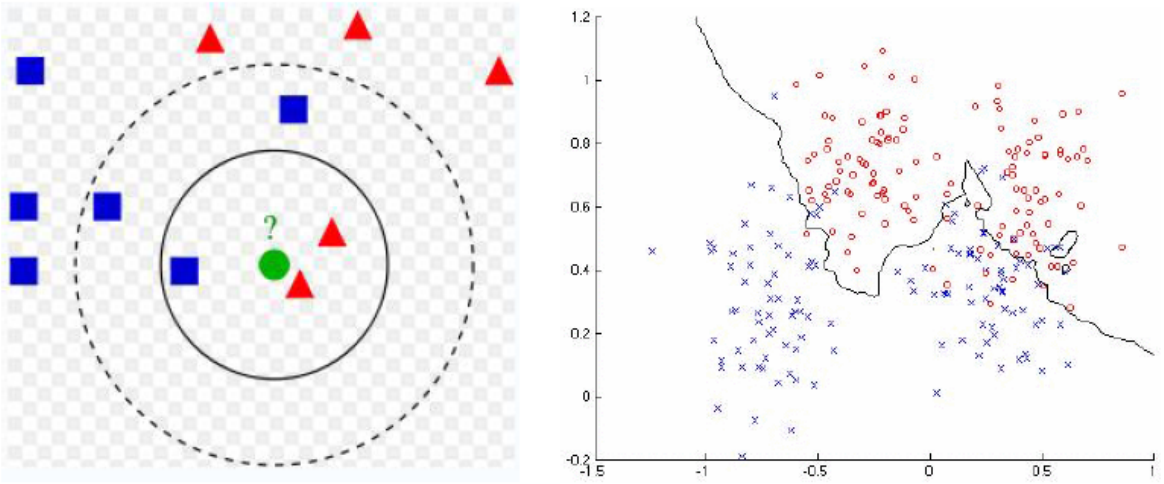


Figure 5 : SVM non-linéaires (SYS828, 2023)

- **k-Nearest Neighbors (k-NN):**

L'algorithme k-NN est une méthode simple de classification où un échantillon est classifié en fonction de la classe majoritaire parmi ses k voisins les plus proches dans l'espace des caractéristiques. En effet, les échantillons qui se ressemblent dans l'espace des caractéristiques sont susceptibles d'appartenir à la même classe.
C'est une approche par modélisation simple qui approxime la frontière de décision localement. Il n'y a pas d'entraînement proprement dit et les calculs sont seulement effectués lors de la classification. On utilise généralement la distance euclidienne pour calculer la similarité et définir le voisinage. Le classificateur k-NN est une méthode non paramétrique qui ne nécessite pas d'établir une hypothèse au

préalable sur la nature des distributions de données. Une grande valeur de k réduit l'effet du bruit sur les données, mais définie des frontières de décisions sans tenir compte de particularités locales, alors qu'une valeur trop petite de k peut rendre l'algorithme sensible au bruit.



(a) Classification (b) Frontière de décision

Figure 6 : Algorithme k-NN (SYS828, 2023)

- **Convolutional Neural Networks (CNN) :**

Les CNN sont des modèles d'apprentissage profond largement utilisés pour la reconnaissance d'images et la vision par ordinateur. Ils sont capables d'apprendre des caractéristiques discriminantes à partir des images de visages grâce à des couches convolutives et de réaliser une classification précise.
Les couches convolutives constituent le cœur des CNN. Elles appliquent des filtres (kernels) sur l'image pour extraire des caractéristiques locales, comme les contours et les textures, à différentes échelles et positions. On retrouve généralement une activation par la fonction ReLU qui atténue les valeurs négatives à zéro.
Après les convolutions, des couches de pooling sont utilisées pour réduire la dimension spatiale des caractéristiques extraites, tout en préservant les informations les plus importantes. Le sous-échantillonnage réduit la taille des données tout en maintenant la robustesse du modèle.
Enfin, les caractéristiques apprises sont transmises aux couches entièrement connectées qui effectuent la classification finale en attribuant des probabilités aux différentes classes.
Les CNN sont entraînés par rétropropagation, où les poids des connexions entre les neurones sont ajustés itérativement en fonction de l'erreur entre les prédictions du modèle et les étiquettes réelles du jeu de données d'entraînement.

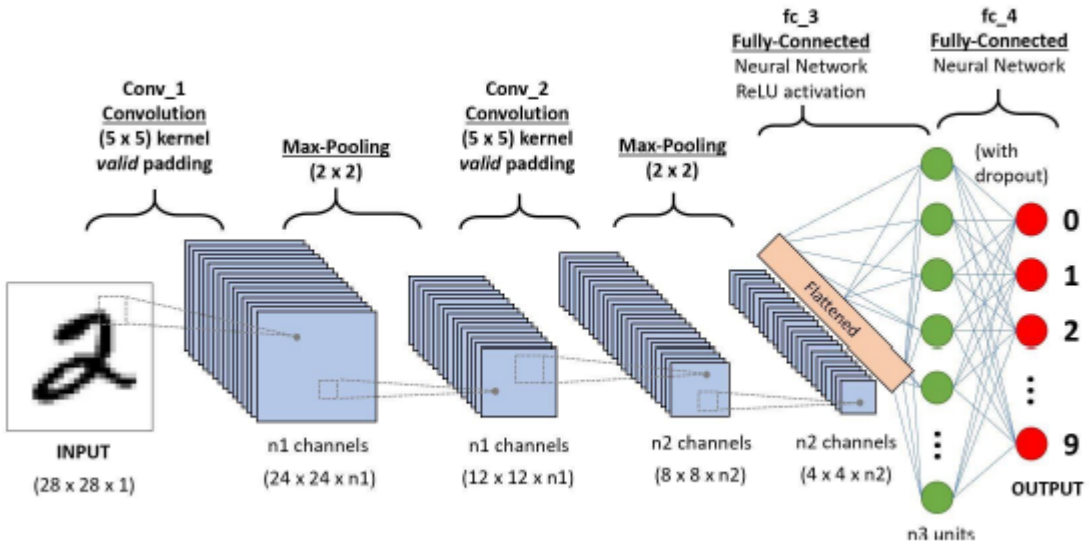


Figure 7 : Exemple d'architecture CNN (SYS828, 2023)

Combinaison de réducteurs

La combinaison de PCA (Principal Component Analysis) et LDA (Linear Discriminant Analysis) est une approche couramment utilisée pour la réduction de dimensionnalité en identification de visages.

L'approche courante pour combinaison de PCA et LDA consiste à utiliser PCA pour réduire la dimensionnalité des données d'origine en un espace de dimension inférieure, puis à appliquer LDA sur les données transformées par PCA. Ainsi, PCA est appliqué sur le jeu de données d'entraînement pour extraire les principales composantes qui expliquent la variation des données. Enfin, LDA est appliqué sur les données réduites et cherche à trouver les axes linéaires qui maximisent la séparation entre les classes tout en maintenant la compacité des données intra-classe.

Ainsi, la combinaison de ces réducteurs permet de bénéficier des avantages des deux approches en projetant les données dans un sous-espace de dimension réduite, et en ajoutant une dimension de séparation guidée par les classes. Cette approche permet d'améliorer la robustesse et la précision des systèmes de reconnaissance de visages

Combinaison de classificateurs

La combinaison de classificateurs (ensemble learning) vise à améliorer les performances et la robustesse des modèles de classification en combinant les prédictions de plusieurs classificateurs.
On retrouve le Bagging (Bootstrap Aggregating), dans lequel plusieurs classificateurs sont formés de manière indépendante en utilisant des échantillons aléatoires avec remplacement du jeu de données d'entraînement. La classe prédite est la classe votée par la majorité des classificateurs. On peut citer l'algorithme Randmom Forest ou Bagged K-NN.
On retrouve également le boosting qui consiste à former les classificateurs de manière itérative et adaptative. Chaque classificateur est formé pour corriger les erreurs commises par les classificateurs précédents. La prédiction finale est obtenue en combinant les prédictions pondérées de chaque classificateur, où les classificateurs les plus performants ont un poids plus élevé. On peut citer l'algorithme AdaBoost ou Gradient Boosting.

Les avantages de la combinaison de classificateurs résident dans le fait qu'elle peut améliorer la précision et la généralisation du modèle, tout en réduisant le risque de surapprentissage.

III) Méthodologie

Protocoles

Afin d'établir une étude analytique de chacun des extracteurs de caractéristiques et des classificateurs ainsi que de leurs combinaisons, nous avons testé chacun de ces algorithmes dans le code en annexe.

Nous avons ainsi appliqué l'algorithme PCA et/ou LDA aux données brutes et calculer les taux de prédictions pour les différents classificateurs (SVM, k-NN et CNN). Nous allons analyser leur performance de manière quantitative et qualitative sur la base de données "lfw_people". Pour ce faire, nous avons utilisé plusieurs protocoles différents :

- Apprentissage avec validation

Une méthode généralement utilisée pour déterminer les paramètres de divers classificateurs est l'apprentissage avec validation. Cette technique consiste à séparer les données utilisées pour l'apprentissage en 3 groupes – un premier dédié à l'entraînement, l'autre à la validation et le troisième au test. Nous faisons alors évoluer les paramètres d'un algorithme d'apprentissage à l'aide d'entraînements

successifs avec les mêmes données, et chaque présentation des données d'entraînement est alors appelée époque d'entraînement. Les performances du classificateur sont évaluées avec les données de validation après certaines époques d'entraînement et l'apprentissage est poursuivi jusqu'à ce que le classificateur cesse de s'améliorer. Ainsi, l'ensemble d'entraînement est utilisé pour former les modèles, l'ensemble de validation est utilisé pour ajuster les hyperparamètres et l'ensemble de test est utilisé pour évaluer les performances finales.

- Validation croisée k-fold

Afin d'éviter le biais de sélection des ensembles d'entraînement, de validation et de test, nous avons utilisé la k cross validation dans certains cas. séparer les données en K groupes de données de même taille, puis à utiliser chaque groupe individuellement et alternativement comme base d'entraînement, et les K-1 groupes restant comme base de validation. On réduit la probabilité de se trouver dans un cas particulier via cette méthode et on réduit par conséquent considérablement le biais de nos résultats. On peut ainsi calculer la moyenne des performances sur les k essais pour obtenir des résultats plus robustes et moins biaisés.

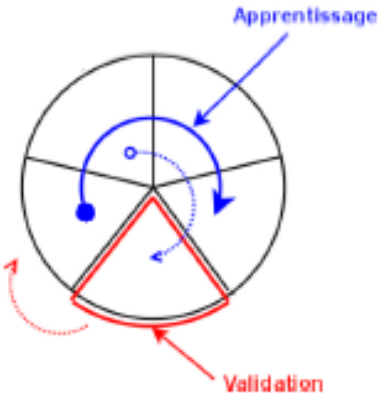


Figure 8 : Lfw_people dataset (Kaggle, 2019)

- Finetuning d'hyperparamètres

Enfin, pour les extracteurs PCA et LDA, nous nous sommes assurés d'ajuster le nombre de composants donnant les meilleures performances en itérant sur les résultats de validation. De plus, nous avons fait la même démarche pour déterminer les meilleurs kernel et régularisations pour le SVM et le nombre de k voisin donnant la meilleure performance pour l'algorithme k-NN.

Si nous avions eu une base de données plus conséquente, nous aurions pu utiliser la validation Hold-out, qui fait plusieurs réplications à l'aide de groupes de données distincts et réduit le coût d'entrainement par rapport à la k cross validation.

Base de données

La base de données "lfw_people" (Labeled Faces in the Wild) est une collection d'images de visages de célébrités prises dans des conditions naturelles et variées. Elle contient des milliers d'images de plus de mille célébrités différentes, avec des expressions faciales, des poses et des conditions d'éclairage variées. Chaque image est étiquetée avec le nom de la célébrité représentée. Cette base de données est utilisée pour évaluer et comparer les performances des algorithmes de reconnaissance de visages dans des situations du monde réel. En raison de la grande variabilité des images, elle représente un vrai défi pour les systèmes de reconnaissance de visages.

Elle est constituée de 5749 classes différentes et de 13233 images au total. Elle contient jusqu'à 255 features réelles.

Figure 9 : Lfw_people dataset (Kaggle, 2019)

Indicateurs de performance

Afin d'évaluer les performances de classificateurs, une base de données de test, indépendante à la base de données d'apprentissage est utilisée. Les indicateurs de performance permettent de quantifier l'exactitude et l'efficacité du modèle dans la prédiction des étiquettes de classe pour de nouvelles données. Chaque algorithme d'apprentissage est évalué en fonction de ses capacités de généralisation, des ressources en mémoire utilisées et du coût de calcul. Les principaux indicateurs de performance utilisés sont :

A) Le taux de classification (accuracy) : ratio de bonnes classifications obtenues par rapport à l'ensemble des données de la base de test. Pour les

problèmes à plusieurs classes, il est également possible d'utiliser une matrice de confusion qui comptabilise les taux de classifications pour chaque classe. Cependant, il peut être trompeur lorsque les classes sont déséquilibrées.

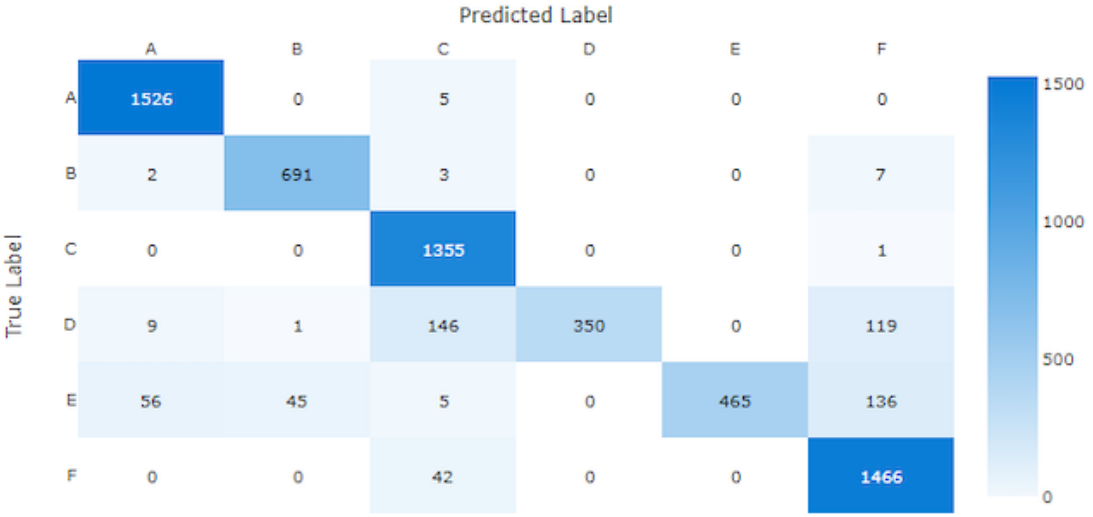


Figure 10 : Exemple de matrice de confusion (Microsoft, 2023)

B) La grandeur du classificateur : le nombre d'éléments utilisés pour modéliser les données d'entraînement. Dans le cas des réseaux de neurones, par exemple, cette valeur serait le nombre de neurones utilisés. La compression peut également être utilisée. Plutôt que d'indiquer directement les ressources utilisées, la compression indique le nombre moyen d'exemples modélisés par chaque de neurones.

Les valeurs optimales des paramètres sont déterminées lorsque l'erreur de classification est au minimum. Avant et après ce moment, l'algorithme est respectivement en sous-apprentissage et en sur-apprentissage. En sous-apprentissage, le classificateur n'est pas optimal alors qu'en sur-apprentissage, les données sont apprises par cœur et le classificateur perd ses capacités de généralisation.

C) Le temps de convergence : Le nombre d'itérations nécessaires pour arrêter l'algorithme d'époques d'apprentissage. Pour les réseaux de neurones, ceci revient à utiliser le nombre d'époques d'apprentissage.

D) Précision/Rappel : La précision est la proportion d'exemples classifiés correctement parmi les exemples classifiés positifs. Le rappel est la proportion d'exemples classifiés correctement parmi les exemples vraiment positifs. Ensuite, on trouve la précision moyenne (AP) en traçant la courbe Précision-Rappel, puis en appliquant une méthode d'interpolation.

IV) Résultats et discussion

Étude analytique

L'algorithme PCA nous permet de réduire la dimensionnalité des données. Il a un impact sur la performance en réduisant le temps et le coût d'entrainement. Nous utiliserons (n_components = 100 pour k-NN, 150 puis 50 pour SVM selon le kernel)

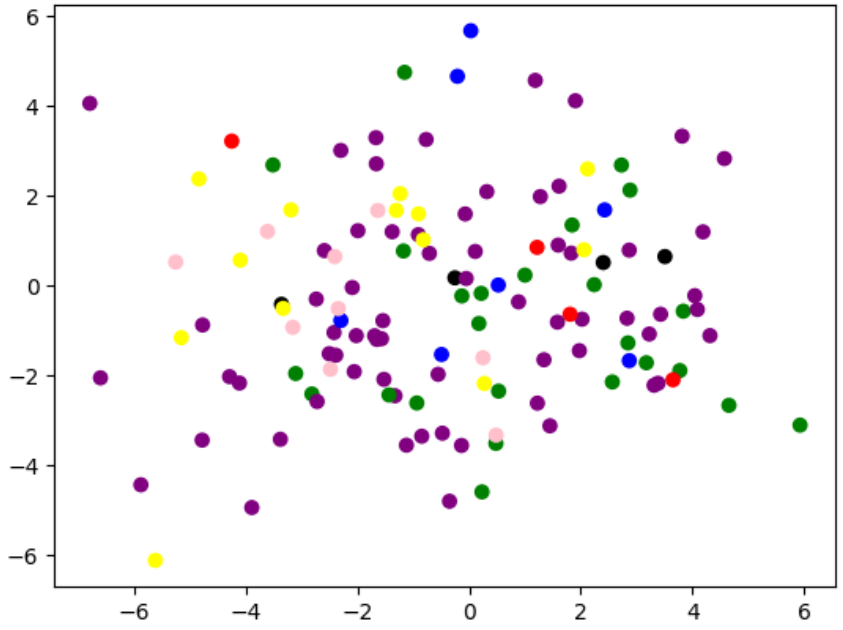


Figure 11 : Graphique montrant la répartition des données avec PCA

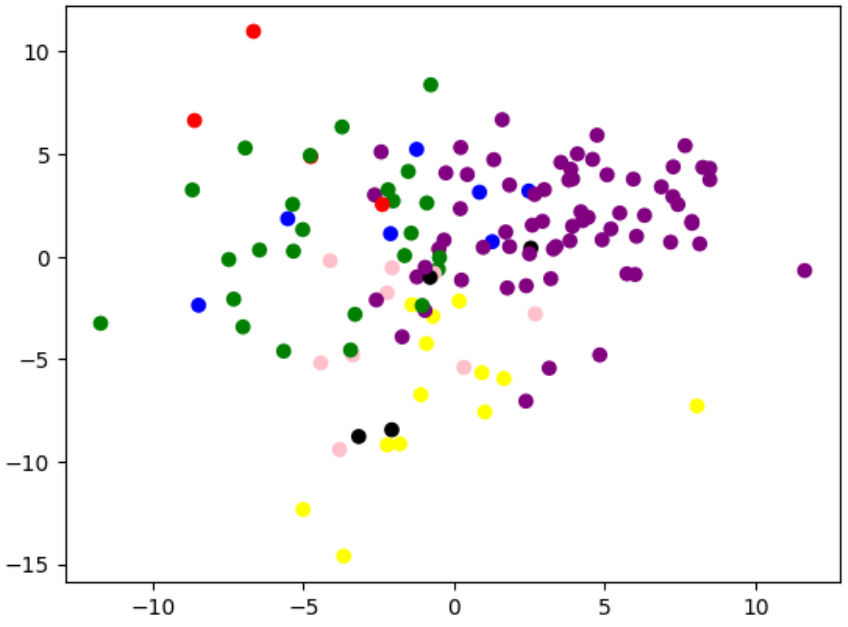


Figure 12 : Graphique montrant la répartition des données avec LDA

On remarque qu'avec l'algorithme LDA les classes sont regroupées en clusters, ce qui est un point indispensable pour effectuer ensuite la classification. En effet, celui-ci sépare les classes en proposant de nouvelles dimensions discriminantes. Nous utiliserons par ailleurs, (n_components = 4 pour k-NN, et 5 pour SVM)

Pour les classificateurs, nous pouvons rechercher les hyperparamètres optimisant les performances. Pour un SVM cela passe par la valeur de C et le type de noyau utilisé (linéaire, polynomial, RBF). On peut visualiser les marges et juger de leur efficacité. Les tests sont faits sur chaque noyau avec différentes valeurs de C (on conserve le maximum de classification). On remarque que le SVM peut être sensible aux outliers, une marge souple peut contrer ce problème.

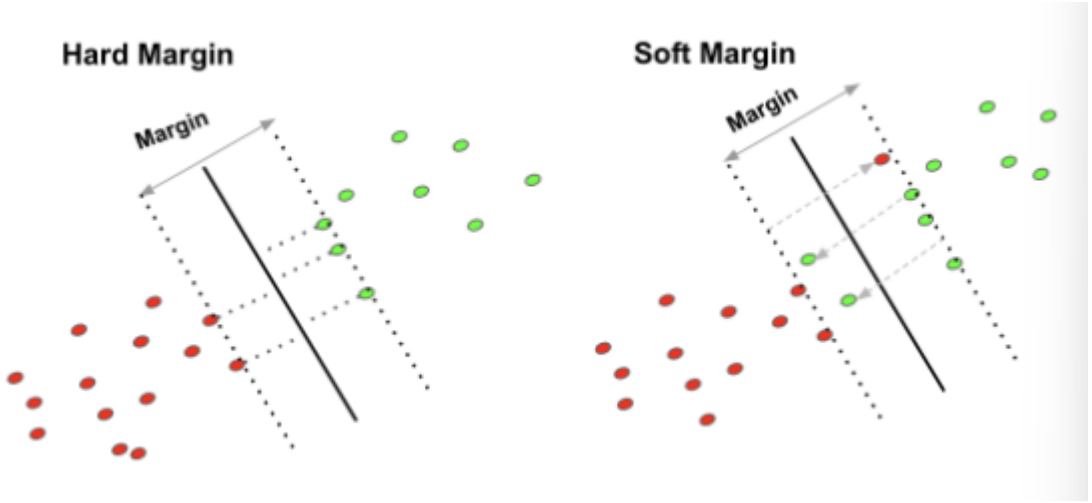


Figure 13 : Hard margin et Soft margin du SVM

En ce qui concerne le k-NN, le choix de valeur de k peut être déterminé en itérant sur celle-ci pour déterminer la plus appropriée. On peut apprécier l'impact du choix de la distance sur ces performances et évaluer la complexité computationnelle qui en découle. Nous conservons k = 12 pour plus de performance.

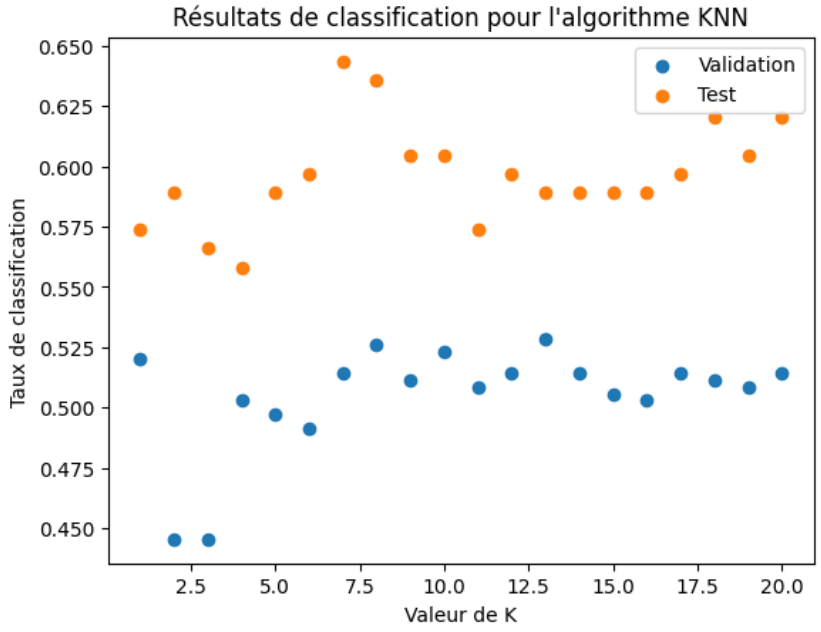


Figure 14 : Classification de K-NN pour différentes valeurs de k

Enfin, les architectures des réseaux CNN peuvent être modifié de façon itérative pour correspondre au mieux au problème (seulement 2 couches de convolutions nécessaires pour notre base de données). On peut également utiliser un réseau préentrainé pour apprentissage par transfert (comme le réseau Resnet 18 que nous utiliserons). Enfin, on peut modifier les hyperparamètres comme le taux d'apprentissage et l'optimiseur et juger de leur impact sur la classification et le coût de l'entrainement.

Benchmark des méthodes

Pour le benchmark quantitatif, nous allons former chaque combinaison d'extracteur de caractéristiques et de classificateur sur un jeu d'entraînement, puis évaluer leurs performances sur un jeu de test, toujours sur la base de données lfw_people.

Figure 15 : Benchmark des différentes méthodes

Approche de classification	**Taux de bonnes classifications**
Classification brute LDA	72.41%
Combinaison PCA puis LDA	**82.18%**
K-nearest neighbors (RAW)	51.44%
K-nearest neighbors (PCA)	53.45%
K-nearest neighbors (LDA)	69.83%
K-nearest neighbors (PCA + LDA)	53.74%
SVM (linear) (PCA)	**82.74%**
SVM (linear) (LDA)	69.11%
Best SVM (linear)	82.74% (C=0.1)
Best SVM (kernel poly) (PCA)	72.29% (C=10)
Best SVM (kernel poly) (LDA)	70.74% (C=1)
Best SVM (kernel RBF) (PCA)	**83.72%** (C=5)
Best SVM (kernel RBF) (LDA)	72.87% (C=0.1)
CNN (2 couches)	**91,25%**
ResNet18	**95.64%**

On remarque alors que certaines combinaisons d'extracteurs et classificateurs sont particulièrement performantes et peu coûteuses comme le SVM linéaire avec PCA à 82.74 % ou encore le SVM RBF et PCA avec 83.72%. Enfin, on remarque que les CNN sont les plus efficaces bien que leurs entrainements soient plus longs. En effet, cela peut s'expliquer par leur performance remarquable pour la vision par ordinateur.

Approfondissements : Réseaux Siamois

Les **réseaux siamois** sont un type particulier d'architecture de réseau de neurones utilisée principalement pour des tâches de comparaison et de similarité entre des paires de données. Ils tirent leur nom de la notion de jumeaux siamois, car ils utilisent deux branches du réseau qui partagent les mêmes poids et architectures, comme des jumeaux. L'idée centrale des réseaux siamois est d'apprendre des représentations de haute qualité pour des objets similaires, en les rapprochant dans un espace de caractéristiques commun.

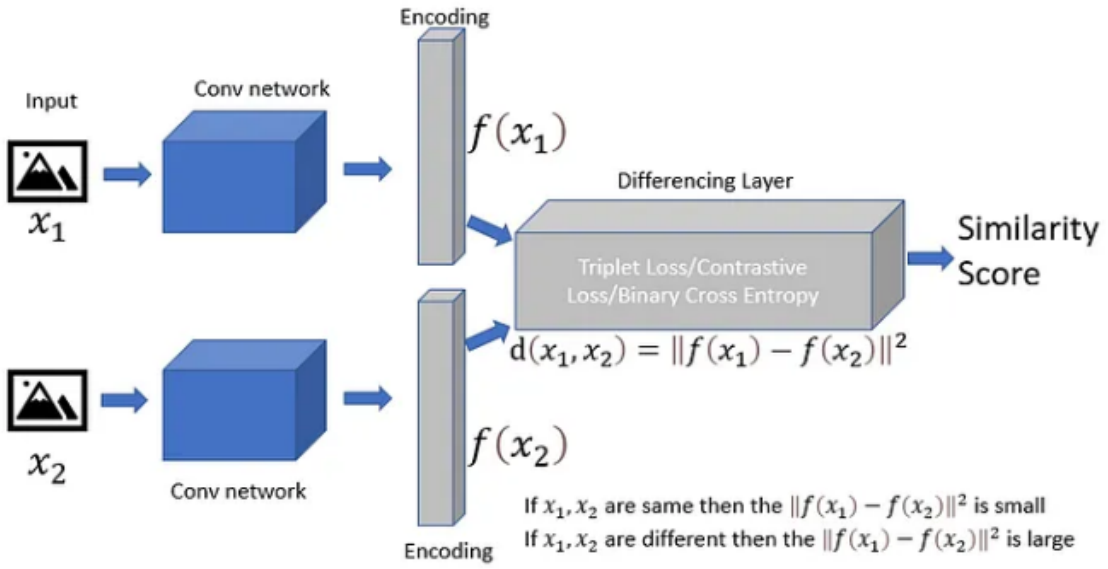


Figure 16 : Réseau siamois (Ren Khandelwal, 2021)

Ainsi, nous sommes face à une architecture de réseau de neurones qui prend deux entrées différentes transmises par deux sous-réseaux identiques, miroirs l'un de l'autre, avec la même architecture, paramètres et poids. Chaque branche traite une des deux entrées (paires d'images, de séquences, etc.) et génère une représentation de ces données. L'objectif du réseau siamois est de classer si les deux entrées sont identiques ou différentes en utilisant un score de similarité, qui peut être calculé à l'aide de **l'entropie croisée binaire**, de la **fonction contrastive** ou de la **perte de triplet**.

Pour former le réseau siamois, on charge un jeu de données contenant différentes classes, puis on crée des paires de données positives (identiques) et négatives (différentes). On construit ensuite le réseau neuronal convolutif (CNN) avec la même architecture pour les deux sous-réseaux, en utilisant une couche entièrement connectée pour encoder les caractéristiques. Une fois que les deux branches ont généré leurs représentations respectives, la distance entre ces représentations est calculée pour obtenir un score de similarité à l'aide d'une couche entièrement connectée avec activation sigmoïde.

Les deux branches du réseau partagent exactement les mêmes poids et biais, ce qui signifie qu'elles sont entraînées ensemble pour extraire des caractéristiques similaires à partir de chaque entrée.

Le réseau siamois est généralement formé en utilisant une fonction de perte qui encourage les paires d'entrées similaires à avoir des représentations rapprochées, tandis que les paires d'entrées non similaires ont des représentations éloignées.
Les fonctions de perte utilisées incluent l'entropie croisée binaire, la fonction contrastive et la perte de triplet. La perte contrastive différencie les images similaires et dissemblables en comparant leurs distances. La perte de triplet utilise des triplets de données pour minimiser la distance entre l'ancre et l'échantillon positif, tout en

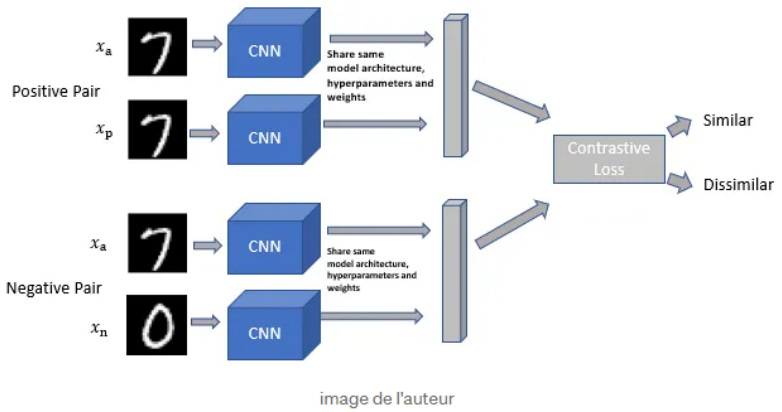


Figure 17 : Constrastive Loss (Ren Khandelwal, 2021)

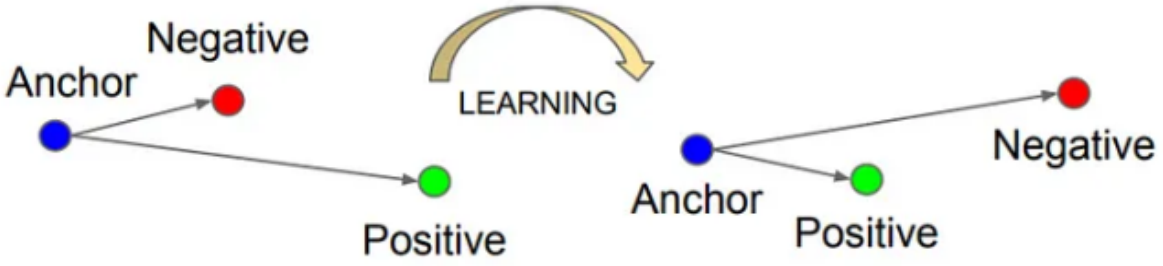
$$Contrastive\ Loss = (1-Y)\frac{1}{2}D_w^2 + (Y)\frac{1}{2}\{\max(0, m - D_\omega^2)\}$$

Les avantages sont qu'il est robuste au déséquilibre de classe et qu'il nécessite peu d'informations. Son apprentissage ponctuel lui permet de prédire la similitude ou la différence entre deux entrées, même avec peu d'exemples d'entraînement.

L'objectif de la fonction de perte contrastive est d'avoir une petite distance pour les paires positives et une plus grande distance pour les paires négatives.

Pour ce projet, nous avons utilisé la fonction de perte de triplet. Son objectif est d'apprendre des embeddings (encodages) de telle sorte que les échantillons similaires soient rapprochés dans l'espace des embeddings, tandis que les échantillons dissemblables soient éloignés les uns des autres. Cela permet de créer des représentations d'échantillons qui sont plus discriminantes pour les tâches de comparaison.

La fonction de perte de triplet prend généralement trois échantillons à la fois : l'ancre (A), un échantillon positif (P) qui est similaire à l'ancre, et un échantillon négatif (N) qui est dissemblable de l'ancre. L'objectif est de minimiser la distance entre l'ancre et l'échantillon positif, tout en maximisant la distance entre l'ancre et l'échantillon négatif, en respectant une certaine marge (margin).



$$L = \max(\mathrm{d}(a,p) - \mathrm{d}(a,n) + margin, 0)$$

Figure 18 : Triplet loss (FaceNet, 2015)

Expérimentation

Nous avons pu, grâce au squelette de code proposé, découvrir un modèle de réseau siamois et l'entrainer nous-mêmes avec des données de la base lfw_people. Après avoir codé les fonctions de manipulation de données prepare_dataset, prepare_data_idset et extract_fold_subset, nous avons extrait 15 images par identités représentant à 96 identités et à 3595 images au total.

On met en place une validation croisée à 5 folds et on réserve 21 classes pour réaliser les tests. Ainsi, les classes rencontrées lors de la phase de test n'ont pas été rencontrées par le modèle pendant la phase d'entraînement. Enfin, on définit ensuite la fonction de perte (triplet loss) et l'optimiseur Adam, tous deux couramment utilisés pour l'entrainement de réseaux siamois (lr = 0.0001).

Enfin, on peut comparer dans la fonction train les outputs du réseau et calculer la distance entre les paires d'images. On compose nos batchs de données avec un certain nombre d'ids identiques pour confronter des paires similaires et différentes.

La fonction de perte de triplet permet d'éloigner les paires différentes et rapprocher les paires identiques.

Pour valider ce modèle, on utilise souvent une mesure de mAP, rank-1 et rank-5. On peut ainsi choisir notre mesure de similarité (distance euclidienne ou **distance cosinus**) et produire la matrice de similarité correspondante pour enfin évaluer notre performance. On peut ainsi définir le meilleur modèle pour chaque fold et analyser les métriques afin de déterminer le modèle le plus performant pour notre application.

```
fc : Rank-1: 100.00% | Rank-5: 100.00% | mAP: 62.24%
att : Rank-1: 100.00% | Rank-5: 100.00% | mAP: 61.42%
Epoch : 73 Iteration :  0  Loss =  0.04022646322846413
Epoch : 73 Iteration :  5  Loss =  0.06699022650718689
Epoch : 73 Iteration :  10  Loss =  0.04742693901062012
Epoch : 73 Iteration :  15  Loss =  0.05884022265672684
Epoch : 73 Iteration :  20  Loss =  0.07937929034233093
Epoch : 73 Iteration :  25  Loss =  0.07835343480110168
Epoch : 74 Iteration :  0  Loss =  0.012500420212745667
Epoch : 74 Iteration :  5  Loss =  0.05534464120864868
Epoch : 74 Iteration :  10  Loss =  0.04192022234201431
Epoch : 74 Iteration :  15  Loss =  0.0721636414527893
Epoch : 74 Iteration :  20  Loss =  0.06942557543516159
Epoch : 74 Iteration :  25  Loss =  0.0
Extracting Gallery Feature...
fc : Rank-1: 100.00% | Rank-5: 100.00% | mAP: 64.03%
att : Rank-1: 100.00% | Rank-5: 100.00% | mAP: 63.59%
Epoch : 75 Iteration :  0  Loss =  0.0
Epoch : 75 Iteration :  5  Loss =  0.0016223639249801636
```

Figure 19: Résultats du modèle siamois

On récupère ainsi la meilleure performance pour les folds données (5) et les epochs souhaitées (100). On remarque une performance maximale de 67%, pour le fold 2. Il serait possible de tuner encore les hyperparamètres pour obtenir un meilleur résultat même si le délai ne nous le permet pas.

En théorie, les réseaux siamois sont particulièrement utiles pour comparer et identifier des similarités entre différentes images. Ils sont intéressants lorsque les données labellisées sont limitées et que l'on souhaite que le modèle généralise facilement (one shot). Enfin, il reste plus robuste au déséquilibre de classe.

V) Conclusion

Ainsi, ce projet nous a beaucoup appris sur l'identification de visages en utilisant une combinaison de différentes approches, notamment les réseaux de neurones convolutifs (CNN), les machines à vecteurs de support (SVM) et les k-plus proches voisins (k-NN). Différentes méthodes d'extraction de caractéristiques, telles que PCA et LDA, ont également été étudiés ainsi que leur impact sur l'amélioration

des performances des classificateurs. Enfin, le réseau siamois, une architecture unique pour classer la similitude ou la dissemblance entre les entrées, a également été exploré.

Les performances des différents classificateurs ont été évaluées et comparées. Les résultats ont montré que la combinaison de différentes approches, en particulier PCA et LDA, peut conduire à de meilleurs taux de classification.

Recommandations

Il serait intéressant d'explorer d'autres architectures de réseaux neuronaux, telles que les réseaux de neurones récurrents (RNN) ou les transformers, pour voir comment elles pourraient améliorer les performances de l'identification de visages.

plus, une augmentation des données peut aider à améliorer la généralisation du modèle et à éviter le surapprentissage. En effet, c'est une technique qui consiste à générer de nouvelles données à partir des données existantes en effectuant des transformations telles que la rotation, le zoom, ou le changement de luminosité.

Améliorations possibles

L'utilisation de bases de données plus grandes et plus diversifiées peut aider à améliorer la capacité du modèle à généraliser à différentes variations de visages.

En plus de PCA et LDA, d'autres méthodes d'extraction de caractéristiques peuvent être explorées pour voir si elles conduisent à de meilleures performances.

En résumé, le projet a démontré l'efficacité de différentes approches pour l'identification de visages. Cependant, il y a encore de nombreuses possibilités d'amélioration et de recherche pour pousser les performances encore plus loin. L'identification de visages est un domaine en constante évolution, et l'utilisation de techniques d'apprentissage en profondeur offre de nombreuses opportunités pour des avancées futures.

VI) Références

- Gérard Dubey, (2008), *L'identification biométrique : vers un nouveau contrôle social ? :* https://journals.openedition.org/rsa/352

- Rafael M. O. Cruz, (2023), *SYS828 Systèmes biométriques :* https://ena.etsmtl.ca/course/view.php?id=18789

- Bilal Alchalabi, (2023), *Laboratoires d'évaluation d'algorithmes pour l'identification de visages statiques - SYS828 :* https://ena.etsmtl.ca/course/view.php?id=18789#section-1

- Kaggle, (2019), *LFW - People (Face Recognition)* : https://www.kaggle.com/datasets/atulanandjha/lfwpeople

- Renu Khandelwal, (2021), *One-Shot Learning With Siamese Network* : https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda#:~:text=Loss%20functions%20used%20in%20Siamese%20Network&text=between%20its%20inputs.-,The%20Similarity%20score%20can%20be%20calculated%20using%20Binary%20cross%2Dentropy,function%20is%20the%20default%20choice.

- Farah F. Alkhalid, (2022), *The Effect of Optimizers on Siamese Neural Network Performance* : https://ieomsociety.org/proceedings/2022istanbul/1019.pdf