

1. Introduction

L'extraction des attributs caractéristiques est l'une des étapes les plus importantes pour la réussite d'un système de reconnaissance du visage, car dans cette étape les caractéristiques extraites représentent le mieux les informations portées par un visage ce qui mène à une meilleure reconnaissance. Plusieurs méthodes d'extraction de caractéristiques existent, alors nous allons présenter celles qui sont les plus utilisées dans le domaine de la reconnaissance faciale tel que l'analyse en composant principale (PCA) et l'analyse discriminative linéaire (LDA) avec une approche bidimensionnelle de ces méthodes 2D-PCA et 2D-LDA.

La classification est une étape qui consiste à prédire la classe correspondante à chaque personne, dans ce chapitre nous allons aussi étudier les méthodes de classifications à savoir, le réseau de neurone et le classificateur K plus proche voisins.

2. Méthodes d'extraction des caractéristiques

2.1 Analyse de Composante Principale (PCA)

L'analyse en composantes principales ou "Principal component analysis" (PCA) est une méthode statistique qui permet de réduire la dimension originale de données dans les problèmes de reconnaissance et de compression.

L'approche PCA est également appelée transformée de Karhunen-Loeve. Elle transforme les données initiales de haute dimension en un ensemble de plus petite dimension composé de nouvelles variables qui sont des combinaisons linéaires des variables originales. Pour ce faire, PCA cherche une base de composantes principales à partir des vecteurs propres de la matrice de covariance des données. L'approche PCA a été très utilisée pour la reconnaissance de visages.

Sirovich et Kirby [20] ont été les premiers qui utilisent l'approche PCA pour représenter efficacement des images des visages humains. Il a montré que n'importe quel visage en particulier peut être :

- ✓ Représenter l'image dans un nouvel espace plus réduit par rapport à l'espace original, l'image est représentée sur les coordonnées des visages propres (nouvel espace).
- ✓ Représenter efficacement les images de visages, qui peuvent être approximativement reconstruites à partir d'un petit ensemble de poids et d'une image de visage standard (eigenpicture)

Dans ce contexte, Turk et Pentland [21] ont présenté la méthode bien connue “*les visages propres pour la reconnaissance du visage*” en 1991. Depuis ce temps, la PCA a été largement étudié et est devenue l'une des approches les plus réussies dans la reconnaissance du visage.

2.1.1 Reconnaissance de visage par algorithme PCA

Généralement, l'implémentation de l'approche PCA est divisée en deux phases : l'apprentissage et classification. Les étapes incluses dans cette phase sont présentées de la manière suivante :

Étape 1 : soit S un ensemble d'images des visages disponibles, formé de N images de même taille

$$I_i = \begin{bmatrix} X_{11} & \cdots & X_{1M} \\ \vdots & \ddots & \vdots \\ X_{M1} & \cdots & X_{MM} \end{bmatrix}_{M \times M} \quad \text{Pour } i=1, \dots, N$$

$X_{ij} \ i,j=1, \dots, M$: sont les valeurs des pixels

Étape 2 : Concaténer chaque matrice de visage dans un vecteur (Les lignes et les colonnes). Ce dernier est de taille $(M^2 \times 1)$.

$$\begin{bmatrix} X_{11} & \cdots & X_{1M} \\ \vdots & \ddots & \vdots \\ X_{M1} & \cdots & X_{MM} \end{bmatrix}_{M \times M} \xrightarrow{\text{Concaténation}} \begin{bmatrix} X_{11} \\ X_{12} \\ \vdots \\ X_{MM} \end{bmatrix}$$

Figure 3.1 Passage d'une image vers un vecteur dans un espace vectoriel de grande dimension

Étape 3 : Calculer la moyenne des visages et la représenter sous forme d'un vecteur Ψ

$$\Psi = \frac{1}{N} \sum_{i=1}^N \Gamma_i \quad (3.1)$$

Étape 4 : Les visages sont centrés par rapport à leur moyenne

$$\Phi_i = \Gamma_i - \Psi \quad (3.2)$$

Étape 5 : construire la matrice globale A :

$$A = [\Phi_1 \Phi_2 \dots \Phi_N] \quad (3.3)$$

Étape 6 : La matrice de covariance (matrice de dispersion) est calculée par l'équation suivante :

$$C = \frac{1}{M} \sum_{n=1}^M \Phi_n \Phi_n^T \quad (3.4)$$

Le facteur $(1/M)$ va affecter le scalaire de sortie durant l'analyse des vecteurs propres. On peut alors enlever ce facteur d'échelle dans notre calcul, le résultat est :

$$C \cong A \cdot A^T \quad (3.5)$$

Étape 7 :

Cette étape consiste à calculer les vecteurs propres et les valeurs de cette matrice de covariance C de taille $(N \times N)$, c'est-à-dire de l'ordre de la résolution d'une image. Le problème est que cela peut parfois être très difficile et très long. En effet, si $N > M$ (la résolution est supérieure au nombre d'images), il y aura seulement $M - 1$ vecteurs propres qui contiendront de l'information (les vecteurs propres restants auront des valeurs propres associées nulles). Par exemple, pour 100 images de résolution 640×380 , nous pourrions résoudre une matrice L de 100×100 au lieu d'une matrice de 76800×76800 pour ensuite prendre les combinaisons appropriées linéaires des images Φ_i . Le gain de temps de calcul serait considérable. Typiquement, nous passerions d'une complexité de l'ordre du nombre de pixels d'une image à une complexité de l'ordre du nombre d'images.

Étape 8 : Considérons les vecteurs propres V_i de la matrice $A^T A$:

$$A^T A V_i = u_i V_i \quad (3.6)$$

On multiplie les deux cotés par A on obtient :

$$A A^T A V_i = u_i A V_i \quad (3.7)$$

$$C A V_i = u_i A V_i \quad (3.8)$$

$$C U_i = u_i U_i \quad (3.9)$$

Les valeurs propres de matrice C sont choisies dans l'ordre descendant qui représente la variance de la distribution de l'espace original, pour cela nous choisissons les meilleurs vecteurs propres qui correspondent aux meilleures valeurs propres :

$$u_i = \frac{1}{M} \cdot \sum_{i=1}^M \text{var}(U_i \cdot A_i^T) \quad (3.10)$$

On note par $A V_i$ les vecteurs propres de C , les deux matrices $A A^T$ et $A^T A$ ont les mêmes valeurs et vecteur propres qui sont reliés comme suit :

$$U_i = A V_i \quad (3.11)$$

Étape 8 : obtenir les visages propres

Les visages propres sont des combinaisons linéaires des vecteurs propres de la matrice C et de la matrice globale qui contient les images d'apprentissages L (NxN)

$$U_i = \sum_{k=1}^M V_{IK} \phi_k \quad i=1 \dots M \quad (3.12)$$

Étape 9:

La projection des visages dans le nouvel espace engendré par les visages propres, ce qu'on appelle les bases dominantes qui décrivent et caractérisent la base d'apprentissage.

Les poids de projection des visages originaux sur l'espace visage propre sont décrits comme suit :

$$W_{ik} = U_k^T (\Gamma_i - \psi) \quad (3.13)$$

$$W_{ik} = U_k^T \phi_i \quad \text{Pour } k, i=1, 2 \dots M$$

Le vecteur de poids des caractéristiques des visages :

$$\Omega_i^T = [W_{i1} \ W_{i2} \ \dots \ W_{iM}] \quad (3.14)$$

2.1.2 La reconstruction de visage par les visages propres

Une image du visage peut être approximativement reconstituée (voir figure 2.2) en utilisant l'image moyenne et les visages propres comme suit :

$$\hat{X}_i = \Psi + \sum_{k=1}^N W_{ik} U_k \quad i=1 \dots N \quad (3.15)$$

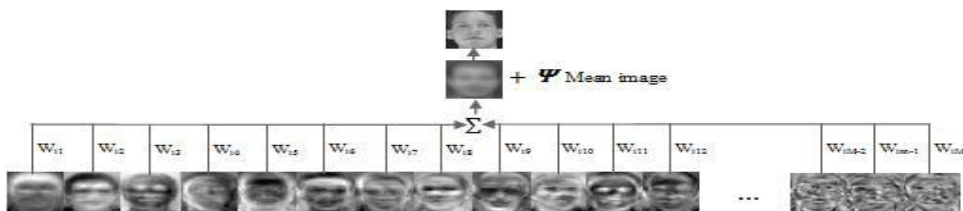


Figure 3.1 La reconstruction de l'image de visage par PCA

Le degré de reconstruction ou le "taux d'erreur de reconstruction (RER)" peut être exprimé au moyen de distance euclidienne entre l'image du visage original et reconstitué comme indiqué:

$$\tau(\%) = \frac{\hat{X} - X}{X} \quad (3.16)$$

\hat{X} : visage reconstruit

X : visage original

2.1.3 Phase de classification

Pour chaque image de teste, les caractéristiques de visage sont tout d'abord extraites puis utilisées indépendamment par la procédure de reconnaissance basée sur "Eigenfaces". Les distances obtenues de l'ensemble des sous-images sont ensuite fusionnées pour générer un résultat global de classification.

Étape 1 : Acquérir un nouveau visage de test pour l'authentification et exprimer comme un vecteur obtenu en concaténant chaque linge en colonne.

$$X_{test} = \begin{bmatrix} X_{11} \\ X_{12} \\ \vdots \\ \vdots \\ X_{MM} \end{bmatrix}$$

Étape 2 : Centrer le visage de test par rapport à la moyenne des visages d'apprentissage :

$$\phi_{test} = \Gamma_{test} - \Psi \quad (3.17)$$

Étape 4 : Projeter l'image de test sur le nouvel espace des visages et extraire le poids de visage de test

$$W_k = U_K(\Gamma_{test} - \Psi) \quad (3.18)$$

Vecteur caractéristique de visage test :

$$\Omega_{test} = [W_1 \ W_2 \ \dots \ W_M] \quad (3.19)$$

Étape 5 : L'étape de classification est la phase dans laquelle le système de reconnaissance de visages assigne un visage test à une classe parmi celles de la base d'apprentissage selon un certain critère bien choisi. Pour cela, nous présentons les distances usuelles les plus utilisées. Calculer la distance entre le vecteur de poids de test et les vecteurs des poids des visages d'apprentissages.

Notons que : Ω_{test} est le vecteur des caractéristiques des visages des tests

Ω_k est le vecteur des caractéristiques des visages d'apprentissages

1) L1 distance (distance Minkowski) :

$$\varepsilon_k(\Omega_{test}; \Omega_k) = |\Omega_{test} - \Omega_k| = \sum_{i=1}^k |(\Omega_{test})_i - (\Omega_k)_i| \quad k=1 \dots N \quad (3.20)$$

2) L2 distance (Euclidian distance):

$$\varepsilon_k(\Omega_{test}; \Omega_k) = \|\Omega_{test} - \Omega_k\| = \sum_{i=1}^k ((\Omega_{test})_i - (\Omega_k)_i)^2 \quad k=1 \dots N \quad (3.21)$$

3) distance MahCosine (angle négative entre les vecteurs des caractéristiques) :

$$(\Omega_{test}; \Omega_k) = -\frac{\Omega_{test} \cdot \Omega_k}{\|\Omega_{test}\| \cdot \|\Omega_k\|} = -\frac{\sum_{i=1}^k (\Omega_{test})_i \cdot (\Omega_k)_i}{\sum_{i=1}^k ((\Omega_{test})_i)^2 \cdot ((\Omega_k)_i)^2} \quad k=1 \dots N \quad (3.22)$$

4) Distance Mahalanobis

$$\varepsilon_k(\Omega_{test}; \Omega_k) = \sqrt{(\Omega_{test} - \Omega_k)C^{-1}(\Omega_{test} - \Omega_k)^T} \quad (3.23)$$

C=matrice de covariance

On simplifie l'équation comme suite :

$$\varepsilon_k(\Omega_{test}; \Omega_k) = -\sum_{i=1}^N (\Omega_{test})_i (\Omega_k)_i Z_i \quad (3.24)$$

$Z = \frac{1}{\lambda_i^2}$, λ_i : les valeurs propres

5) L1 distance +Mahalanobis :

$$\varepsilon_k(\Omega_{test}; \Omega_k) = |\Omega_{test} - \Omega_k| = \sum_{i=1}^k |(\Omega_{test})_i - (\Omega_k)_i| Z_i \quad k=1 \dots N \quad (3.25)$$

6) distance L2 +Mahalanobis :

$$\varepsilon_k(\Omega_{test}; \Omega_k) = \|\Omega_{test} - \Omega_k\| = \sum_{i=1}^k ((\Omega_{test})_i - (\Omega_k)_i)^2 Z_i \quad k=1 \dots N \quad (3.26)$$

7) distance Mahcosine +Mahalanobis:

$$\varepsilon_k(\Omega_{test}; \Omega_k) = -\frac{\Omega_{test} \cdot \Omega_k}{\|\Omega_{test}\| \cdot \|\Omega_k\|} = -\frac{\sum_{i=1}^k (\Omega_{test})_i \cdot (\Omega_k)_i Z_i}{\sum_{i=1}^k ((\Omega_{test})_i)^2 \cdot ((\Omega_k)_i)^2} \quad k=1 \dots N \quad (3.27)$$

$$\varepsilon_k(\Omega_{test}, \Omega_k) = \|\Omega_{test} - \Omega_k\|$$

Étape 10 : calculer le seuil de décision défini comme la moitié de la distance maximale entre toutes les classes :

$$\theta = \frac{1}{2} \max(\|\Omega^i - \Omega^j\|) \quad (3.28)$$

Enfin, on prend la distance minimale qui y est comparée avec un seuil θ bien choisi pour faire la décision comme suit :

If $\min\{\varepsilon_k\} \geq \theta$ ($k=1 \dots M$) c'est un visage connu

If $\min\{\varepsilon_k\} \leq \theta$ ($k=1 \dots M$) c'est un visage inconnu

2.1.4 Les inconvénients de l'approche PCA

PCA représente l'intensité de l'image comme un vecteur, donc la taille du vecteur pourrait être très grande. cette grande taille du vecteur crée un problème de l'estimation des vecteurs propres de la matrice de covariance d'échantillon des données lors de la mise en œuvre de l'algorithme PCA. Pour éviter cet inconvénient, Yang et al. [18] ont proposé la PCA deux dimensions (2DPCA), (ou IMPCA [17]) qui utilise directement l'image en 2D. Généralement, par rapport à le 2DPCA, la méthode PCA a besoin de plus de coefficients pour la représentation d'image. Pour résoudre ce problème, plusieurs solutions ont été proposées. Par exemple la projection bilatérale basée sur 2DPCA (B2DPCA) réduit la

redondance entre les lignes et les colonnes des images des visages par leur projection de gauche et de droite en multipliant les matrices de projection.

La dimension de sous-espace affecte significativement la précision de la reconnaissance de ces méthodes. Si d est petit, les images pourraient perdre des informations discriminantes importantes lorsqu'elles sont projetées sur le sous-espace. Une grande valeur de d pourrait créer le phénomène sur-apprentissage “overfitting” et réduire le taux de la reconnaissance.

Le problème de phénomène “overfitting” pourrait être résolu en utilisant la technique de sous-espace aléatoire (RS) comme a été démontré dans [10].

2.2 Méthodes d'analyse en composantes principales bidimensionnelles 2DPCA

Cette technique est proposée par Zhou [18] pour l'extraction des caractéristiques d'image. Contrairement à l'approche conventionnelle PCA, 2DPCA est représenté les images sous forme des matrices 2D que des vecteurs 1D.

la matrice de covariance d'image peut être construite directement à l'aide des matrices d'image originales. L'approche 2DPCA possède deux avantages importants par rapport au PCA.

- ✓ conserver l'information temporelle et spatiale de l'image.
- ✓ plus rapide que le PCA classique en termes de temps pour calculer la matrice de covariance.

2.2.1 Les étapes de reconnaissance de visage par 2DPCA

2.2.1.1 Etape d'apprentissage

Pour un ensemble d'apprentissage de M matrices de visages, l'idée de cette technique est de projeter une matrice X ($n \times m$) via une transformation linéaire telle que:

$$Y_i = X \cdot R_i \quad (3.29)$$

Où Y_i est dit vecteur composante principale de dimension ($n, 1$) et R_i est le vecteur de projection de taille ($m \times 1$). Le vecteur optimal R_i de la projection est obtenu en maximisant le critère de variance totale généralisé

$$J(R) = R^T \cdot G_t \cdot R \quad (3.30)$$

Où G_t est la matrice de covariance des images de dimension ($m \times m$) donnée par :

$$G_t = \frac{1}{M} \sum_{j=1}^M (X_j - \bar{X})^T (X_j - \bar{X}) \quad (3.31)$$

Avec X_j : la $j^{\text{ème}}$ image dans la base d'apprentissage

\bar{X} : L'image moyenne totale de toutes les images de la base d'apprentissage.

$$\bar{X} = \frac{1}{M} \sum_{j=1}^M X_j \quad (3.32)$$

En général un seul axe de projection optimal n'est pas suffisant. On doit sélectionner un ensemble d'axes de projection tel que:

$$\begin{aligned} \{R_1, R_2, \dots, R_d\} &= \arg \max J(R) \\ R_i^T \cdot R_j &= 0, i \neq j, i, j = 1, \dots, d \end{aligned} \quad (3.33)$$

Ces axes sont les vecteurs propres de la matrice de covariance G_t correspondant aux « d » plus grandes valeurs propres. L'extraction des caractéristiques d'une image X par l'ACP2D se fait donc selon :

$$Y_k = X \cdot R_k ; k=1, \dots, d \quad (3.34)$$

Où $[R_1, R_2, \dots, R_d]$ est la matrice de projection et $[Y_1, Y_2, \dots, Y_d]$ est la matrice caractéristique de l'image X .

2.2.2 Choix du nombre de vecteurs propres

Le nombre de vecteurs propres associés aux plus grandes valeurs propres à retenir est un grand défaut de cette technique. Pour y remédier, les chercheurs ont adopté différentes solutions:

- Pour un ensemble de 115 images, Sirovitch et Kirby ont trouvé que 40 eigenpictures sont suffisantes pour représenter efficacement cet ensemble.
- Turk et Pentland [21] l'ont choisi heuristiquement, pour leurs tests, sur une base de 16 individus, 7 vecteurs propres ont été retenus.
- Moghaddam [3] a préservé, pour comparer différentes approches de reconnaissance de visages 20 vecteurs propres en justifiant son choix par une erreur de reconstruction raisonnable (0.0012) et un taux de reconnaissance 80% obtenu par eigenfaces sur une base de 1829 images,
- Zhao et al [18]. ont retenu 300 vecteurs propres pour une base de 1038 images après avoir observé que pour un nombre très élevé, les eigenfaces ne représentent pas des visages, donc leur choix était basé sur l'allure des eigenfaces au lieu des valeurs propres.

2.2.3 Etape de classification

Après la transformation par 2DPCA, l'extraction des caractéristiques est obtenue de chaque image. Ensuite, le classifieur le plus proche voisin (KNN) est utilisé pour la classification, la

distance entre deux matrices de caractéristique arbitraires, $B_i = [Y_1^{(i)}, Y_2^{(i)}, \dots, Y_d^{(i)}]$ et $B_j = [Y_1^{(j)}, Y_2^{(j)}, \dots, Y_d^{(j)}]$ est définie par :

$$d(B_i, B_j) = \sum_{k=1}^d \|Y_k^{(i)} - Y_k^{(j)}\|_2 \quad (3.35)$$

On suppose que les images d'apprentissage sont B_1, B_2, \dots, B_N (où N est le nombre total des images d'apprentissage), et chaque image est attribué à une classe P_k .

On prend une image de test B , Si $d(B, B_i) = \min_j d(B, B_j)$ et $B_i \in w_k$, on compare par un seuil bien choisi pour décider la classe de l'image test ($B \in w_k$)

2.2.4 La reconstitution une image de visage par 2DPCA

Dans l'approche "eigenfaces" les composants principaux et les valeurs propres peuvent combiner pour reconstituer le visage d'image par la même manière que 2DPCA.

On pose les vecteurs propres orthonormés qui correspondent aux meilleures valeurs propres de la matrice de covariance de l'image sont X_1, \dots, X_d . Après les échantillons des images sont projetés dans les axes principaux. Les résultats obtenus sont les composants principaux $Y_k = AX_k$ ($k=1, 2, \dots, d$), on prend $V = [Y_1, Y_2, \dots, Y_d]$ et $U = [X_1, X_2, \dots, X_d]$

$$V = AU \quad (3.36)$$

Où les X_1, X_2, \dots, X_d sont orthonormés à partir de l'équation (2.40). Ça permet d'obtenir les échantillons des images reconstituées A

$$\hat{A} = VU^T = \sum_{k=1}^d Y_k X_k^T \quad (3.37)$$

Avec $\hat{A}_k = Y_k X_k^T$ ($k=1, 2, \dots, d$) qui est de même taille que l'image A , et représente l'image reconstruite de A . Autrement dit, l'image A peut-être approximativement reconstitué par l'addition des premiers d'images. En particulier, lorsque le nombre de vecteurs composants principaux choisis $d = n$ (n est le nombre total de vecteurs propres de G_T), nous avons $\hat{A} = A$ i.e., l'image est complètement reconstruit par ses vecteurs composant principal et sans perte d'information. Sinon, si $d < n$, l'image reconstruite A est une approximation de A . Dans le 2DPCA, la dimension d du sous-espace a une grande influence sur l'énergie de l'image construite. Selon Yang et al. [18], l'énergie de l'image construite est concentrée sur un petit nombre des premiers vecteurs composants correspondant aux valeurs propres plus importantes. Ainsi, en choisissant la petite dimension d est suffisante pour obtenir un taux de reconnaissance plus élevé pour le 2DPCA.

2.3 Analyse discriminante linéaire

L'algorithme LDA est né des travaux de Belhumeur et al. De la Yale University (USA), en 1997 [24]. Considérons un ensemble de personnes connues ayant un certain nombre d'images d'apprentissage. Ces personnes sont partitionnées en C classes (groupes), chaque classe correspond à une identité différente. Chacun des individus étant décrit par une image de d caractéristiques et l'on connaît sa classe d'appartenance.

D'un point de vue mathématique, l'analyse discriminante linéaire (LDA) est une technique supervisée, basée sur la maximisation d'un critère de séparabilité. Soit un ensemble de N images d'apprentissage $X_i \in R^d$, appartenant à C classes différentes. Chaque classe X_i contient N_i images d'une personne.

La matrice $W = [w_1, w_2, \dots, w_l]$ contenant les axes les plus discriminants définit la matrice de Projection sur le sous-espace F . La projection Y_i de X_i sur W est donnée par :

$$Y_i = W^T X_i \quad (3.38)$$

Le vecteur Y_i ainsi obtenu, de longueur l , définit la nouvelle représentation compacte associée au vecteur-image X_i .

En appliquant la méthode LDA, nous trouvons les directions de projection que d'une part d'optimiser la distance entre les images de visage de différentes classes et d'autre part de minimiser la distance entre les images de visage de la même classe, en autres termes, la maximisation de la matrice de dispersion Inter-Classe tout en minimisant la matrice de dispersion intra-classe dans le sous-espace projectif. Figure 3.3 montre la meilleure et mauvais séparation de la classe.



Figure 3.2 (a) la bonne séparation de classe (b) la mauvaise séparation de classe

Le Fisher discriminant linéaire (LDA) surmonte les limitations de la méthode PCA en appliquant un nouveau critère. Ce critère cherche à maximiser le rapport de projection entre la matrice intra-classe et la matrice-interclasse.

$$J_{FLD}(W_{opt}) = \arg \max_W \frac{|W^T S_b W|}{|W^T S_w W|} \quad (3.39)$$

S_b : Matrice de dispersion inter-classe

S_w : Matrice de dispersion intra-classe

➤ **La matrice de dispersion intra-classe:**

La matrice de dispersion de même classe S_w , appelée aussi intra-personnel, représente les variations dans l'apparence d'un même individu en raison d'un éclairage différent et l'expression du visage et elle représente la distribution des images intra –classe par rapport au leur moyenne.

➤ **La matrice de dispersion entre les classes:**

La matrice de dispersion entre les classes S_b , appelée aussi extra-personnel, représente les variations d'apparence dues à la différence entre l'identité. Elle décrit comment les classes sont séparées les unes des autres.

2.3.1 Le problème “SSS” (small sample size problem) de LDA

L'algorithme LDA souffre d'un problème majeur, c'est ce qu'on appelle la singularité de la matrice intra classe $S_w \in R^{n \times n}$ est toujours singulière car la taille de l'image est plus grande par rapport au nombre d'individus de la même classe ($M \ll n \times m$) donc il sera difficile de calculer. Et on ne peut plus déterminer directement la matrice W , c'est le problème de la singularité.

Pour remédier à ce problème, plusieurs travaux de recherche ont été réalisés, utilisant des techniques telles que LDA Fisher, R-LDA, D-LDA, LDA/QR, inverse-LDA et kernel LDA [29, 9, 25, 14, 15, 16]

Dans ce chapitre, nous présentons une implémentations détaillée de l'approche LDA Fisher sur la reconnaissance de visage, on utilisera la PCA comme une étape de prétraitement pour réduire les dimensions des données originales. Ensuite nous présenterons des techniques liés à l'approche LDA.

2.3.2 Reconnaissance des visages en utilisant FLD (méthode Fisherfaces)

La méthode des Fisher faces est sans doute la plus connue des approches utilisant l'analyse discriminante linéaire dans le contexte de la reconnaissance automatique de visages. Elle a été introduite par Belhumeur et al. [24] pour résoudre le problème de la singularité de la matrice de dispersion intra-classe S_w .

Les étapes de base de la méthode LDA Fisher sont les suivants :

Étape 1 : Nous avons besoin d'un ensemble d'apprentissage composé d'un groupe relativement important d'exemples avec diverses caractéristiques faciales. La base de données doit contenir plusieurs exemples d'images de visages pour chaque exemple dans le ensemble d'apprentissage et au moins un exemple dans l'ensemble de test.

Étape 2 : en représentant chaque image par une matrice à deux dimension $N \times N$ de valeurs d'intensité, nous construisons l'extension vectorielle lexicographique $I \in R^{N \times N}$. ce vecteur correspond à la représentation initiale du visage. Ainsi, l'ensemble de tous les visages dans l'espace caractéristique est considérée comme un espace vectoriel de dimension élevée.

Étape 3 : Définir toutes les instances du même visage des personnes comme étant dans une classe et les visages des différents sujets comme étant dans des classes différentes pour tous les sujets dans l'ensemble d'apprentissage .

Étape 4 : dans le but de remédier au problème de singularité de approche LDA, nous ajoutons un étage de prétraitement lequel utilise l'approche PCA pour réduire l'espace de dimension $M - c$, puis on appliquant l'approche LDA afin réduire l'espace de dimension qui est connu sous le nom d'espace de Fisher (c-1).

Étape 5 : Calcul la moyenne de visage de chaque classe comme suit :

$$u_j = \frac{1}{n_j} \sum_{i=1}^{n_j} \Omega_{PCA_i^j} \quad (3.40)$$

u_j :Un vecteur de colonne de dimension (M-cX1)

n_j : Nombre des individus de chaque classe

$\Omega_{PCA_i^j}$: représente les vecteurs caractéristiques de classe j.

Ω_{PCA} : La matrice de projection de taille (M-c)XM M : le nombre total des images d'apprentissages

Étape 6: calcul la moyenne totale des vecteurs caractéristiques de la classe d'images d'apprentissages

$$\mu = \frac{1}{M} \sum_{j=1}^c \Omega_{PCA_i^j} \quad (3.41)$$

μ : Vecteur de colonne $(M - c) \times 1$

Étape 7 : Centrer tous les classes par rapport à leur moyenne

$$\phi_i^j = \Omega_{PCA_i^j} - u_j ; i=1 \dots n_j, j=1 \dots c \quad (3.42)$$

ϕ_i^j : Vecteur de colonne $(M - c) \times 1$

Étape 8: Construire les matrices de dispersion pour chaque classe $S_1, S_2 \dots S_c$.

$$S_j = \sum_{i=1}^{n_j} \phi_i^j \phi_i^{jT} \quad (3.43)$$

S : matrice de dispersion (covariance) de taille $(M-c) \times (M-c)$

Étape 8 : Calculer la matrice intra-classe de dispersion.

$$S_W = \sum_{j=1}^c \sum_{i=1}^{n_j} (\Omega_{PCA_i^j} - u_j)(\Omega_{PCA_i^j} - u_j)^T \quad (3.44)$$

S_W : Matrice de taille $(M-c) \times (M-c)$

Étape 9 : Calculer la matrice Inter-Classe de dispersion

$$S_b = \sum_{j=1}^c n_j (u_j - u)(u_j - u)^T \quad (3.45)$$

S_b : Matrice de taille $(M-c) \times (M-c)$

Étape 10 : La projection optimale est choisie comme une matrice avec des colonnes orthonormées qui maximisent le rapport du déterminant de la matrice de dispersion des échantillons projetés Inter-Classe au déterminant de la matrice de dispersion des échantillons projetés intra-classe, à savoir :

$$W_{opt} = \arg \max_W \frac{|W^T S_b W|}{|W^T S_W W|} \quad (3.46)$$

Ce rapport est maximisé lorsque les vecteurs des colonnes de la matrice de projection w sont les vecteurs propres de $S_b S_W^{-1}$:

$$W = eig(S_b S_W^{-1}) \quad (3.47)$$

Les Colonnes de W sont les vecteurs propres satisfaisants :

$$S_b W_i = \lambda_i S_W W_i \quad i=1, 2, 3 \dots m \quad (m=c-1) \quad (3.48)$$

Les valeurs propres généralisées sont les racines du polynôme caractéristique:

$$|S_b - \lambda S_W| = 0 \quad (3.49)$$

La solution optimale de cette équation est constituée par les vecteurs propres de la matrice suivante:

$$(S_b - \lambda S_W) W_i = 0 \quad (3.50)$$

Les vecteurs propres de matrice $(S_b S_W^{-1})$ sont appelées Fisherfaces :

$$W_{LDA\ opt} = [W_{i1} \ W_{i2} \ \dots \ W_{i(c-1)}] \quad (3.51)$$

Étape 11 : la projection des visages sur l'espace de Fisher

La matrice de projection optimale est calculée par :

$$\Omega_{LDA} = W_{LDA}^T \Omega_{PCA} \quad (3.52)$$

Ω_{LDA} : est la matrice de projection de Fisher de taille $c - 1 \times M$

Ω_{PCA} : est la matrice de projection de l'espace original de l'image vers le sous-espace PCA de taille $M - c \times M$

W_{LDA}^T : est la matrice de projection du sous-espace PCA vers le sous-espace LDA de taille $(M - c) \times (c - 1)$

2.3.3 Partie de test (classification)

Le visage de test au début est projetée dans un sous espace PCA pour réduire la dimension, puis ré-projeté à nouveau dans l'espace LDA pour extraire les vecteurs de caractéristiques des visages.

$$W_k = U_K(\Gamma_{test} - \Psi) \quad \text{Pour } k=1 \dots \dots \dots M-c \quad (3.53)$$

$$\Omega_{test PCA} = [W_1 \ W_2 \ \dots \dots W_{M-c}] \quad (3.54)$$

Les poids de visage de test qui est obtenu à partir de la projection de l'espace LDA est utilisé pour mesurer la similarité.

$$\Omega_{test LDA} = W_{LDA}^T \Omega_{test PCA} \quad (3.55)$$

A chaque visage d'apprentissage est associé une identité sous la forme d'étiquette $im^{test}(j) = id(C)$

Nous cherchons dans cette partie à vérifier un visage de test à partir des classes des visages d'apprentissages.

La méthode de classification de visage est la même utilisée dans l'approches PCA, on mesure la distance de similarité par méthode euclidienne puis en compare avec un seuil pour vérifier ce visage, s'il est connu ou inconnu.

2.4 La méthode 2DLDA

En 2006 S. Nourath [30] ont proposé une nouvelle approche LDA bidimensionnelle. La différence principale entre 2DLDA et LDA classique se trouve dans le modèle de représentation des données. LDA classique travaille avec les représentations vectorielles de données, tandis que l'algorithme 2DLDA utilise les données sous forme matricielle. Dans cette méthode, 2DLDA utilise directement la matrice d'image pour calculer la matrice de dispersion entre la classe et matrice de dispersion intra-classe.

2.4.1 Reconnaissance du visage à l'aide algorithme 2D LDA

Soit X désigne un vecteur des colonnes unitaires à n -dimensions.

L'idée principale de cette approche est de projeter l'image A matrice aléatoire, de $m \times n$, sur X par une transformation linéaire suivante :

$$Y = AX \quad (3.56)$$

Ainsi, on obtient un vecteur de projection m-dimensionnelle Y , qui est appelé le vecteur caractéristique de l'image projetée A .

Supposons qu'ils y sont des L classes connus, et M dénote le nombre total des échantillons de toutes les classes.

L'image d'apprentissage est représentée par une matrice $m \times n$ $A_j (j = 1, \dots, M)$ et \bar{A}_i ($i=1, \dots, L$) désigne l'image moyenne de class T_i , et N_i est le nombre d'échantillons dans la classe T_i , la classe projetée est P_i . Après la projection de l'image d'apprentissage sur X , on obtient la matrice de la caractéristique projetée :

$$Y_i = A_j X, j = 1, 2, \dots, M \quad (3.57)$$

Objectif de cette équation est de trouver une meilleur projection, en effet, la dispersion totale des échantillons projetés peut être caractérisé par le tracé de la matrice de covariance de projection des vecteurs de caractéristiques [24]. De ce point de vue, nous avons introduit un critère;

$$J_{FLD}(W_{opt}) = \arg \max_W \frac{|W^T S_b W|}{|W^T S_w W|} \quad (3.58)$$

$$P_b = \text{trace}(S_b)$$

$$P_w = \text{trace}(S_w)$$

Où, S_b représente la matrice de dispersion inter-class de vecteurs de caractéristiques de projection d'images d'apprentissage, et S_w représente la matrice de dispersion intra-classe de vecteurs de caractéristiques de projection d'images d'apprentissage. Si :

$$S_b = \sum_{i=1}^N N_i (\bar{Y}_i - \bar{Y})(\bar{Y}_i - \bar{Y})^T$$

$$S_b = \sum_{i=1}^L N_i [(\bar{A}_i - \bar{A})X][(\bar{A}_i - \bar{A})X]^T \quad (3.59)$$

$$S_w = \sum_{i=1}^L \sum_{y_k \in P_i} (\bar{Y}_k - \bar{Y})(\bar{Y}_k - \bar{Y})^T$$

$$S_w = \sum_{i=1}^L \sum_{y_k \in P_i} [(\bar{A}_k - \bar{A})X][(\bar{A}_k - \bar{A})X]^T \quad (3.60)$$

C'est dire :

$$\text{trace}(S_b) = X^T \left[\sum_{i=1}^L N_i (\bar{A}_i - \bar{A})^T (\bar{A}_i - \bar{A}) \right] X$$

$$= X^T S_b X \quad (3.61)$$

$$\text{trace}(S_w) = X^T \left[\sum_{i=1}^L \sum_{y_k \in P_i} (\bar{A}_i - \bar{A})^T (\bar{A}_i - \bar{A}) \right] X$$

$$= X^T S_W X \quad (3.62)$$

Nous pourrions évaluer et utiliser directement des échantillons des images d'apprentissage. Ainsi, le critère peut être exprimé par :

$$J(X) = \frac{X^T S_W X}{X^T S_b X} \quad (3.63)$$

Où X : vecteur colonne unitaire.

Le vecteur unitaire X qui maximise $J(X)$ est appelé l'axe de projection optimal. La projection optimale est choisie lorsque X_{OPT} maximise le critère, comme l'équation suivante :

$$X_{OPT} = \underset{X}{\operatorname{argmax}} J(X) \quad (3.64)$$

Si S_W est inversible, la solution d'optimisation est de résoudre le problème aux valeurs propres généralisé:

$$S_b X_{opt} = \lambda S_W X_{opt} \quad (3.65)$$

λ est la valeur propre maximale de $S_W^{-1} S_b$

En général, ce n'est pas suffisant d'avoir un seul axe de projection optimale. Nous avons besoin de sélectionner un ensemble des axes de projection, x_1, x_2, \dots, x_d avec sous contraintes qui sont les suivantes :

$$\begin{aligned} \{x_1, x_2, \dots, x_d\} &= \underset{X}{\operatorname{argmax}} J(X) \\ X_i^T X_j &= 0, i \neq j, i, j = 1, 2, \dots, d \end{aligned}$$

En effet, les axes de projection optimale, x_1, x_2, \dots, x_d sont les vecteurs propres orthonormés de $S_W^{-1} S_b$ correspondant aux d premières meilleures valeurs propres.

L'utilisation de ces axes de projection permet de créer une nouvelle matrice de projection X , qui est une matrice de $n \times d$:

$$X = [x_1, x_2, \dots, x_d] \quad (3.66)$$

2.4.2 Extraction des caractéristiques

Nous allons utiliser les vecteurs de projection optimale de 2DLDA x_1, x_2, \dots, x_d pour extraire les caractéristiques d'image, on utilise l'équation suivante :

$$y_k = A X_k, k = 1, 2, \dots, d \quad (3.67)$$

Puis, nous avons une famille de vecteurs caractéristiques y_1, y_2, \dots, y_d qui forme la matrice $y = [y_1, y_2, \dots, y_d]$ de taille $M \times d$ appelée matrice de caractéristiques de l'image A.

2.4.3 Reconstruction d'image par 2DLDA

Dans la méthode 2DLDA, nous pouvons utiliser les matrices caractéristiques et les axes de projections optimales pour reconstruire l'image de visage.

Les caractéristiques d'image A sont obtenues par l'équation (3.68)

$$Y = AX$$

Par ailleurs les vecteurs x_1, x_2, \dots, x_d sont orthonormaux, il est facile d'obtenir l'image reconstruite de A:

$$\tilde{A} = YX^T = \sum_{k=1}^d y_k x_k^T \quad (3.69)$$

Nous avons appelé $\tilde{A} = y_k x_k^T$ comme une image reconstituée de A, qui a la même taille que l'image A. Si nous choisissons $d = n$, alors nous pouvons reconstruire complètement les images dans l'ensemble de l'apprentissage : $\tilde{A} = A$. Si $d < n$ l'image construite \tilde{A} est une approximation de A.

2.4.4 Classification

Etant données deux images de visages A1, A2 représentée par la matrice de caractéristiques $2DLDA_v = [y_1^1, y_2^1, \dots, y_d^1]$ et $Y_2 = [y_1^2, y_2^2, \dots, y_d^2]$. Ainsi, la similarité $d(Y_1, Y_2)$ est défini comme :

$$d(Y_1, Y_2) = \sum_{k=1}^d \|y_1^k - y_2^k\|_2 \quad (3.70)$$

Où $\|y_1^k - y_2^k\|$: dénote la distance euclidienne entre les deux vecteurs de caractéristiques y_1^k et y_2^k . Si la matrice de caractéristiques d'images d'apprentissage sont y_1, y_2, \dots, y_M (M est le nombre total d'images d'apprentissage), et chaque image est assignée à une classe C. Puis, pour une image de test donnée Y, Si $d(Y, Y_i) = \min_j d(Y, Y_j)$ et $Y_j \in T_i$, alors la décision résultante est $Y \in C_i$

Il est à noter que toutes les méthodes présentées jusqu'ici sont des méthodes linéaires.

Cependant, les variations dans les images sont de nature non linéaire. Afin de pouvoir traiter ce problème de non-linéarité en reconnaissance faciale, dans ce mémoire nous nous sommes penchés aux approches neuronales qui ont montré leur succès dans différentes applications de vision par ordinateur dont la tâche de classification et la discrimination des données.

3. Méthodes de Classification

3.1 Réseaux de Neurones

Le domaine des réseaux de neurones est comme tout autre domaine de la science, une longue histoire de développement avec de nombreux hauts et des bas, comme nous le verrons bientôt.

L'histoire des réseaux de neurones commence au début des années 1940 et donc presque simultanément avec l'historique des ordinateurs électroniques programmables.

3.1.1 Neurones biologique

On pense que le système nerveux compte plus de 1000 milliards de neurones interconnectés. Bien que les neurones ne soient pas tous identiques, leur forme et certaines caractéristiques permettent de les répartir en quelques grandes classes. En effet, il est aussi important de savoir, que les neurones n'ont pas tous un comportement similaire en fonction de leur position dans le cerveau. Les neurones sont reliés entre eux par l'intermédiaire d'axones et de dendrites. En première approche, on peut considérer que ces sortes de filaments sont conducteurs d'électricité et peuvent ainsi véhiculer des messages depuis un neurone vers un autre. Les dendrites représentent les entrées du neurone et son axone sa sortie.

Un neurone émet un signal en fonction des signaux qui lui proviennent des autres neurones. On observe en fait au niveau d'un neurone, une *intégration* des signaux reçus au cours du temps, c'est à dire une sorte de sommations des signaux. En général, quand la somme dépasse un certain seuil, le neurone émet à son tour un signal électrique.

La notion de *synapse* explique la transmission des signaux entre un axone et une dendrite. Au niveau de la jonction (c'est à dire de la synapse), il existe un espace vide à travers lequel le signal électrique ne peut pas se propager. La transmission se fait alors par l'intermédiaire de substances chimiques, les *neuro-médiateurs*. Quand un signal arrive au niveau de la synapse, il provoque l'émission de neuro-médiateurs qui vont se fixer sur des récepteurs de l'autre côté de l'espace inter-synaptique. Quand suffisamment de molécules se sont fixées, un signal électrique est émis de l'autre côté et on a donc une transmission. En fait, suivant le type de la synapse, l'activité d'un neurone peut renforcer ou diminuer l'activité de ces voisins. On parle ainsi de synapse excitatrice ou inhibitrice. [37]

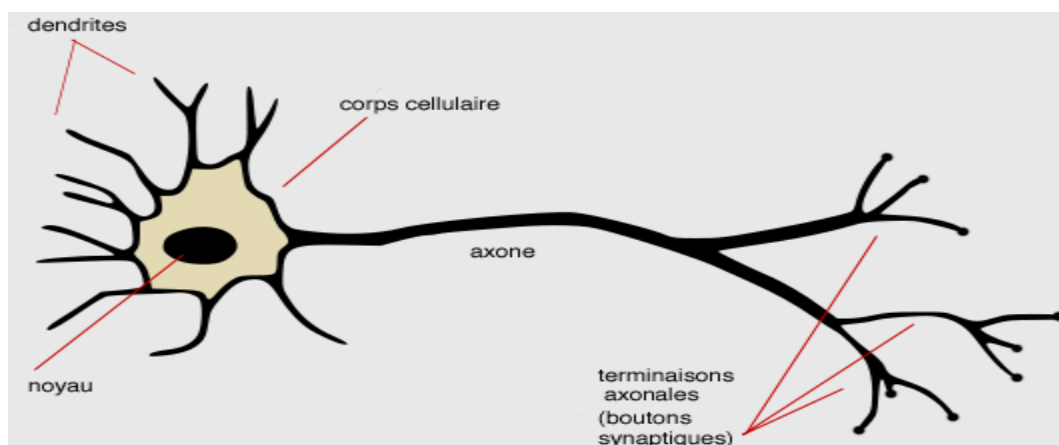


Figure 3.3 Neurone biologique

3.1.2 Réseaux de neurones formels

Les réseaux de neurones formels sont à l'origine une tentative de modélisation mathématique du cerveau humain. Les premiers travaux datent de 1943 et sont l'œuvre de MM. *Mac Culloch et Pitts*. Ils présentent un modèle assez simple pour les neurones et explorent les possibilités de ce modèle.

Le modèle de neurone formel présenté ici, du à Mac Culloch et Pitts, est un modèle mathématique très simple dérivé d'une analyse (elle aussi assez simple) de la réalité biologique. On constate tout d'abord que le modèle biologique fait intervenir une notion temporelle qui est difficile à intégrer dans un modèle simple. On oublie donc cette notion et de ce fait on remplace l'intégration temporelle par une simple sommation des signaux arrivant au neurone (ces signaux sont communément appelés les *entrées* du neurone). On compare ensuite la somme obtenue à un seuil et on déduit de la comparaison la sortie du neurone. Cette sortie sera par exemple égale à 1 si la somme est supérieure au seuil et à 0 dans le cas contraire. Plus formellement encore, il suffit pour obtenir ce comportement de soustraire le seuil considéré à la somme des entrées, et de faire passer le résultat par la *fonction de transfert* du neurone qui est ici la fonction de heaviside. Le résultat après transfert est alors la sortie du neurone. Cette enchaînement "sommation" puis "non-linéarité" représente finalement les propriétés "physiques" du neurone.

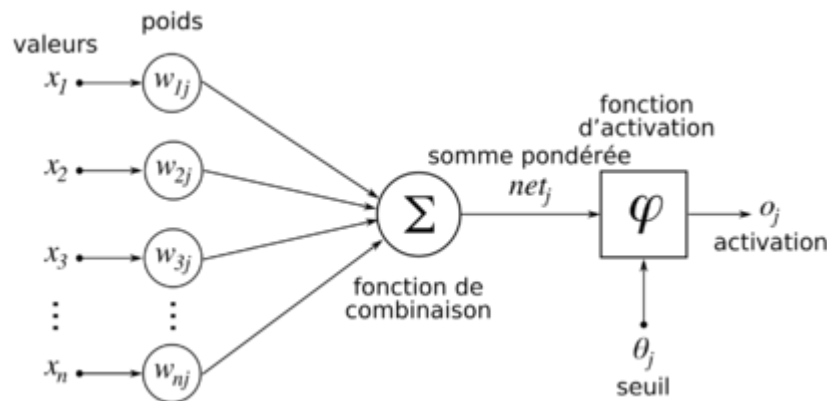


Figure 3.4 Neurone Formel

La modélisation des synapses est assez simpliste en première approche. On se contente en effet d'introduire la notion de synapse excitatrice et de synapse inhibitrice en multipliant la sortie d'un neurone par 1 ou -1 avant de la transmettre aux autres neurones. Afin de donner plus de puissance au modèle, on généralise ce modèle en introduisant ensuite une *connexion synaptique* qui est une valeur réelle. La sortie d'un neurone est alors multipliée par la valeur

de la connexion synaptique avant d'être transmise à un autre neurone. On modélise ainsi la connexion entre deux neurones. [37]

3.1.3 Composantes des réseaux de neurones

Un réseau de neurone technique se compose d'unités de traitement simples, les neurones et dirigés, connexions pondérées entre ces neurones. Ici, la force d'une connexion (Ou le poids de connexion) entre deux neurones i et j est appelé $W_{i,j}$

Un réseau de neurone est un triangle trié (N, V, w) avec deux ensembles N, V et une fonction w , où N est l'ensemble des neurones et V un ensemble des connexions entre le neurone i et le neurone j . La fonction $w : V \rightarrow \mathbb{R}$ définit les poids, où $w((i, j))$, le poids de la connexion entre le neurone i et le neurone j est raccourcie à $W_{i,j}$. Cela dépend de point de vue est soit indéfini, soit 0 pour les connexions qui n'existent pas dans le réseau ainsi, les poids peuvent être mis en œuvre dans une matrice de poids carré W ou, éventuellement, dans un vecteur de poids W avec le numéro de ligne de la matrice indiquant la connexion commence, et le nombre de colonne de la matrice indiquant, quel neurone est la cible. [37]

3.1.4 Modélisation du neurone

Les réseaux de neurones sont des modèles, à ce titre ils peuvent être décrit par leurs composants, leurs variables descriptives et les interactions des composants.

a) Structure

Chaque neurone artificiel est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones amont. A chacune de ces entrées est associée un poids w (abréviations de Wight (<< poids >> en Anglais) représentatif de la force de la connexion. Chaque processeur élémentaire est doté d'une sortie unique, qui se ramifie ensuite pour alimenter un nombre variable de neurones avals. A chaque connexion est associée un poids.

b) Comportement

On distingue deux phases. La première est habituellement le calcul de la somme pondérée des entrées (a) selon l'expression suivante : $a = \sum (w_i * e_i)$.

A partir de cette valeur, une fonction de transfert calcule la valeur de l'état du neurone. C'est cette valeur qui sera transmise aux neurones avals. Il existe de nombreuses formes possibles pour la fonction de transfert, les plus courantes sont présentées sur la figure

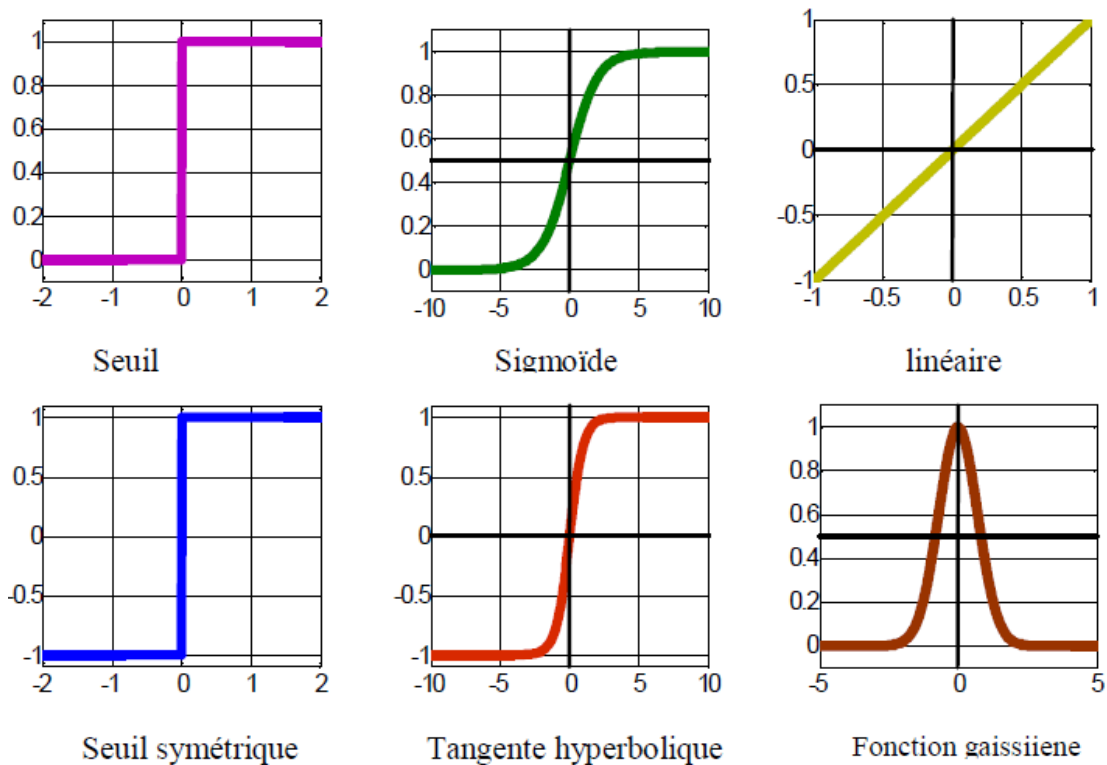


Figure 3.5 Différents types de fonction de transfert.

c) Variables descriptives

Ces variables décrivent l'état du système. Dans le cas des réseaux de neurones qui sont des systèmes non autonomes, un sous-ensemble des variables descriptives est constitué par les variables d'entrée, variables dont la valeur est déterminée extérieurement au modèle.

d) Structure d'interconnexion

Les connexions entre les neurones qui composent le réseau décrivent la topologie du modèle. Elle peut être quelconque, mais le plus souvent il est possible de distinguer une certaine régularité. [33]

3.1.5 Architecture des réseaux neuronaux

3.1.5.1 Réseaux non bouclée (Statique)

Réseaux d'avance permettent aux signaux de voyager d'une manière unique; De l'entrée à la sortie. Il n'y a pas de commentaires (boucles), c'est-à-dire que la sortie de n'importe quelle couche n'affecte pas cette même couche. Les réseaux d'avance ont tendance à être des réseaux directs qui associent les entrées aux résultats.

Ils sont largement utilisés dans la reconnaissance de motifs. Ce type d'organisation est également appelé Bas - En haut ou Haut - vers le bas.

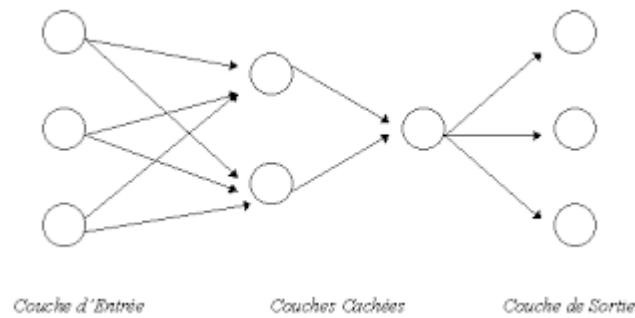


Figure 3.6 Réseaux non bouclée (Statique)

3.1.5.2 Réseaux bouclée (Dynamique)

Les réseaux de rétroaction peuvent transmettre des signaux dans les deux sens en introduisant des boucles dans le réseau. Les réseaux de rétroaction sont très puissants et peuvent devenir extrêmement compliqués. Les réseaux de rétroaction sont dynamiques; Leur «état» change continuellement jusqu'à ce qu'ils atteignent un point d'équilibre. Ils restent au point d'équilibre jusqu'à ce que l'entrée change et qu'un nouvel équilibre soit nécessaire.

Les architectures de rétroaction sont également appelées interactives ou récurrentes, bien que ce dernier terme soit souvent utilisé pour désigner des connexions de rétroaction dans des organisations à une seule couche. [36]

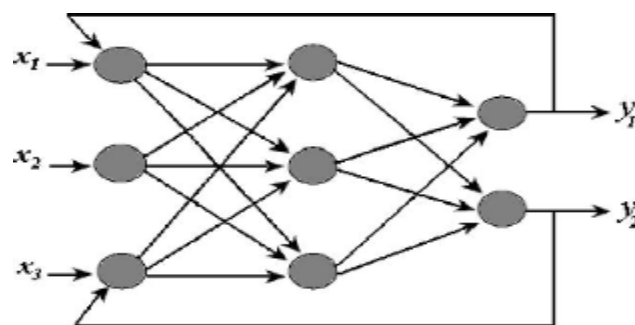


Figure 3.7 Réseaux bouclée (Dynamique)

3.1.6 Apprentissage des Réseaux de Neurones

L'apprentissage est considéré comme une tâche de construction de nouvelles connaissances ou amélioration des connaissances existantes dont le rôle est de définir les poids de chaque connexion. Le but de cette phase est d'améliorer les performances du

système en tenant compte des ressources et des compétences dont il dispose. L'apprentissage et l'adaptation constituent deux caractéristiques essentielles des réseaux de neurones.

Lorsque la phase d'apprentissage est achevée, le réseau doit être capable de faire les bonnes associations pour les vecteurs d'entrées qu'il n'aura pas appris. C'est l'une des propriétés importante dans les réseaux de neurones. Des différents algorithmes d'apprentissage et différents types d'apprentissage ont été développés dans la littérature dans le but de réaliser l'adaptation et l'optimisation des poids d'un réseau de neurone. [31]

3.1.6.1 Apprentissage supervisé

Un superviseur, ou professeur, fournit au réseau des couples d'entrées sorties. Il fait apprendre au réseau l'ensemble de ces couples, par une méthode d'apprentissage. Dans le cas de la rétro-propagation du gradient de l'erreur, en comparant pour chacun d'entre eux la sortie effective du réseau et la sortie désirée. L'apprentissage est terminé lorsque tous les couples entrées-sorties sont reconnus par le réseau. [31]

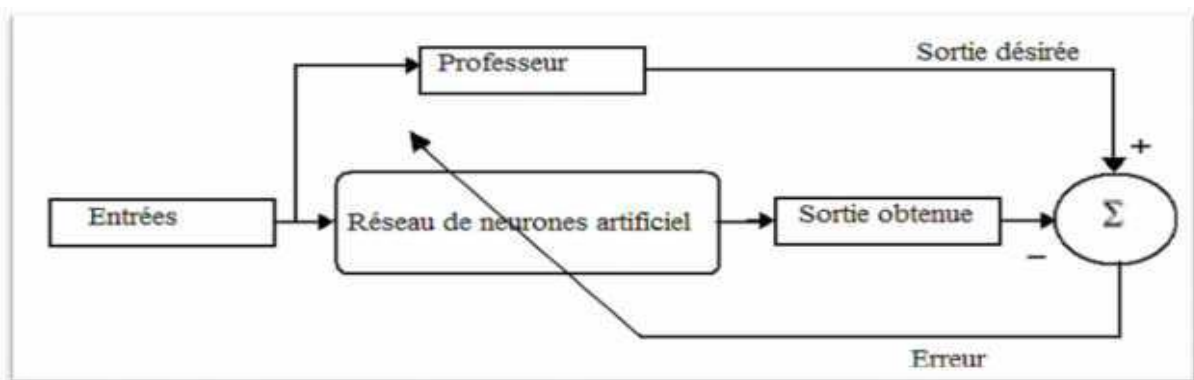


Figure 3.8 Apprentissage supervisé

3.1.6.2 Apprentissage non supervisé

Cet apprentissage consiste à détecter automatiquement des régularités qui figurent dans les exemples présentés et à modifier les poids des connexions pour que les exemples ayant les mêmes caractéristiques de régularité provoquent la même sortie. Les réseaux auto-organiseurs de Kohonen sont les réseaux à apprentissage non supervisé les plus connus. [31]

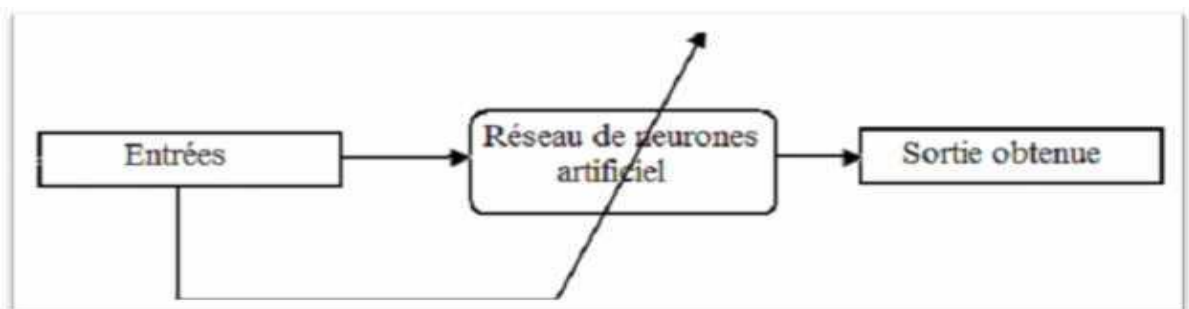


Figure 3.9 Apprentissage non supervisé

3.1.7 Le réseau de neurones probabiliste (PNN)

Le réseau de neurones probabiliste introduit par Donald Specht en 1988, est un réseau feed-forward à 3 couches utilisé pour la classification des données [D. F. Specht, 1990]. À l'inverse des autres réseaux de neurones qui sont basés sur la rétro-propagation, le réseau de neurones probabiliste est basé sur des principes statistiques dérivés de la stratégie de décision de Bayes et des fonctions estimateurs de la densité de probabilité. [32] Pour l'apprentissage d'un réseau probabiliste, il faut d'abord que chaque vecteur \mathbf{x} en entrée soit normalisé de façon que : $\sum_{i=1}^n x_i^2 = 1$. Le nombre de neurones d'entrée est égal au nombre des paramètres (variables) de la forme à classifier. Pour chaque échantillon de la base d'apprentissage, on crée un neurone dans la couche cachée (couche de motifs) avec les connections correspondantes avec les neurones d'entrée de façon que $w_k = x_k$ pour $k=1, 2, \dots, n$. Ensuite une seule connexion est créée vers le neurone de la couche de classification correspondant à la classe de l'échantillon. [1] Ce processus est illustré dans l'algorithme suivant :

Début

Initialisation

$j=0$; n = premier échantillon d'apprentissage ;

Faire

$j \leftarrow j+1$;

Normalisation :

$$x_{jk} \leftarrow \frac{x_{jk}}{\sqrt{\sum_i^d x_{jk}^2}}$$

Apprentissage des poids : $w_{jk} \leftarrow x_{jk}$;

Si $x \in w_j$ **alors** $a_{ic} \leftarrow 1$;

Jusqu'à $j = n$;

Fin

Figure3.11 L'algorithme PNN

3.2 Le classificateur du K plus proches voisins (KNN)

C'est un classificateur simple basé sur le calcul de distance entre les exemples d'apprentissage et les exemples de tests, généralement la norme euclidienne est souvent employée comme mesure de distance, dans chaque étape de l'apprentissage, l'algorithme mémorise les k meilleurs exemples de l'ensemble d'apprentissage ($kppv(x)$) qui sont proches à l'exemple de test x . Cet algorithme est souvent performant s'il y a suffisamment d'exemples d'apprentissage, mais demande un temps de prédiction très long pour passer tous les exemples afin de trouver les K meilleures solutions. ce processus est illustré dans l'algorithme suivant [32]

L'algorithme

Soit $L = \{(x', c) | x' \in R_d, c \in C\}$ est un ensemble d'apprentissage.

(x' est un exemple d'apprentissage et c sa classe qui lui est associée.

Soit x l'exemple de test dont on souhaite déterminer sa classe

Début

Pour chaque (*exemple* $(x', c) \in L$) **faire**

Calculer la distance $D(x, x')$

Fin

Pour chaque $\{x' \in kppv(x)\}$ **faire**

Compter le nombre d'occurrences de chaque classe

Fin

Attribuer à x la classe la plus fréquente

Fin

Figure 3.11 L'algorithme KNN

Conclusion :

L'extraction de caractéristiques est une étape primordiale qui vise à présenter les données dans un espace réduit et discriminant. Les techniques d'apparence globales (2DPCA, 2DLDA,...etc) sont toujours sensibles aux variations intrinsèque et extrinsèque de l'application de reconnaissance du visage (éclairage, occultation, expression faciales, pose de caméra) et qui ont sans doute un challenge pour l'étape d'extraction des caractéristiques dans un système biométrie. Les méthodes de classification sont arrivées pour pallier les lacunes de l'étape précédente. En revanche, dans ce mémoire nous avons introduit le réseau neurone probabiliste dans notre application pour classer ou discriminer les personnes selon leurs caractéristiques extraites par les deux approches à savoir, 2DPCA et 2DLDA.