Diffusion de labels sur graphe : Application à l'identification de visages

Pierre Buyssens Marinette Revenu

GREYC – Université de Caen, CNRS, ENSICAEN
6 Bld du Maréchal Juin, 14050 CAEN
pierre.buyssens@unicaen.fr

Résumé

Nous présentons une méthode générique de classification fondée sur une diffusion de labels sur graphe. Mimant la méthode de ligne de partage des eaux largement utilisée dans le domaine de la segmentation d'images, nous proposons un algorithme s'appliquant à un graphe de distances de topologie quelconque. Appliquée au problème de l'identification de visages, cette méthode est testée avec de nombreux types de distances. Nous proposons également deux fonctions de pénalité qui, utilisées lors du processus de diffusion, permettent une nette amélioration de la classification.

Mots Clef

Diffusion sur graphe, Identification de visages, Biométrie.

Abstract

We present a generic classification method based on a label diffusion on graph. Inspired by the watershed transform widely used in the image segmentation field, we propose a simple algorithm working on distances graphs of any topology. Applied to the faces identification problem, this method can deal with any type of distances. We propose also two penalty functions that clearly improve the identification results when used within the diffusion process.

Keywords

Diffusion on graph, Face Identification, Biometric.

1 Introduction

La reconnaissance faciale de personnes est un sujet dont l'intérêt n'est plus à démontrer : biométrie, vidéo–surveillance, IHM avancées ou encore indexation d'images/vidéos.

L'identification biométrique consiste à trouver une identité inconnue (l'image requête) parmi un ensemble de personnes connues (la galerie). La plupart des approches proposées dans la littérature sont construites sur le même schéma composé de trois étapes : 1) prétraitement des images, 2) extraction de caractéristiques faciales, 3) classification de ces caractéristiques. Cet article traite de la dernière partie, l'étape de classification.

Prétraitement des images La première étape consiste à localiser un visage dans une image, le redimensionner et éventuellement appliquer des algorithmes permettant d'améliorer la qualité de l'image en vue de la seconde étape.

Extraction des caractéristiques La deuxième étape consiste à calculer certaines caractéristiques du visage. Celles–ci doivent être suffisamment simples à calculer, robustes à d'éventuelles dégradations des images, ainsi que suffisamment discriminantes pour différentier deux personnes différentes. De nombreuses méthodes d'extraction de caractéristiques existent dans la littérature, elles peuvent principalement être groupées en deux parties :

– Les méthodes locales extrayant des caractéristiques locales (comme les yeux, le nez ou la bouche) puis les combinant dans un modèle global permettant de « définir » un visage [4] [16],
– Les méthodes globales prennant l'image dans sa globalité et appliquant des méthodes statistiques telle la réduction de dimension. Citons, parmi les plus populaires, l'utilisation de techniques linéaires comme la PCA [15] ou la LDA [9], ainsi que leurs versions non linéaires Kernel–PCA [12], et Kernel–LDA [11].

Classification La dernière étape consiste à comparer un ou plusieurs vecteurs caractéristiques requêtes à un ensemble de vecteurs caractéristiques issus de personnes dont l'identité est connue. De nombreux classifieurs peuvent ici être utilisés. Notons cependant, que l'utilisation de classifieurs fondés sur l'apprentissage (réseaux de neurones, SVM . . .) n'est en pratique pas applicable. En effet, ce type de classifieur apprend à discriminer les différentes identités contenues dans la galerie. Cependant, si une personne est ajoutée (ou enlevée) à la galerie, alors le modèle doit être revu et l'apprentissage doit de nouveau être effectué.

Les algorithmes de classification couramment utilisés en biométrie comparent directement les vecteurs caractéristiques un à un.

L'algorithme classique du plus proche voisin (*NN*) sélectionne le vecteur caractéristique v_g de la galerie $\mathbf{G}$ le plus proche du vecteur requête v_p selon une certaine distance :

$$v_g = \min_{i \in \mathbf{G}} \|v_{g_i} - v_p\|$$

L'identité correspondant à v_p est alors celle de v_g.

D'autres algorithmes de classification utilisent l'ensemble des vecteurs caractéristiques de la galerie pour obtenir un modèle du vecteur requête. C'est le cas pour le mélange de Gaussiennes où chaque vecteur requête est décomposé en une somme pondérée de Gaussiennes :

$$v_p = \sum_i \pi_i * \mathcal{G}(id)$$

où π_i sont les coefficients du mélange et $\mathcal{G}(id)$ les modèles Gaussiens de chaque individu id de la galerie. L'identité de v_p correspond alors à l'identité dont le coefficient de mélange est le plus grand.
Un autre algorithme de classification utilise l'ensemble de la galerie et réalise une décomposition parcimonieuse d'un vecteur requête sur l'ensemble des vecteurs caractéristiques connus [17]. Ici, la décomposition parcimonieuse est réalisée selon l'énergie :

$$E = \min_{\boldsymbol{x} \in \mathbb{R}^m} (\|\boldsymbol{v_p} - A\boldsymbol{x}\|_2^2 + \lambda \|\boldsymbol{x}\|_1)$$

où $A \in \mathbb{R}^{n \times m}$ est une matrice regroupant en colonne les vecteurs caractéristiques de la galerie, et $\boldsymbol{x} \in \mathbb{R}^m$ est le vecteur contenant les coefficients de la décomposition. L'identité de v_p est alors déduite par :

$$identité(v_p) = identité(\min_i \|v_p - A_i x_i\|_2)$$

Dans cet article, nous proposons un autre schéma pour l'identification : utiliser à la fois les images de la galerie et les images de l'ensemble requête pour la classification. Les relations entre les vecteurs sont modélisées à l'aide d'un graphe pondéré où chaque nœud représente un vecteur caractéristique. Un label (l'identité) est attribué à chaque nœud correspondant aux vecteurs caractéristiques de la galerie. Une diffusion de labels sur le graphe affecte alors un label à chaque nœud, et donc spécialement aux nœuds non labélisés correspondant aux vecteurs requêtes. Étant donné qu'une arête reliant deux nœuds non labélisés peut être moins pertinente qu'une arête reliant un nœud initialement labélisé (appartenant à la galerie) et un nœud non labélisé, nous proposons deux fonctions de pénalité pouvant être utilisées lors de la diffusion pour favoriser le deuxième cas (en pénalisant le premier cas). Ce processus permet d'introduire une sorte de distance à la source, rejoignant dans ce sens l'approche de Dijkstra. La Section 4 revient sur cette similarité à Dijkstra ainsi que sur le comportement différent des deux approches dans le cas de graphes complets. Notons que la littérature est abondante dans ce domaine avec notamment les techniques de marche aléatoire [10], de noyaux de diffusion sur graphes [8] ou encore de plus courts chemins [3].
Un cas d'usage typique de cet algorithme est son utilisation lorsque le système possède plusieurs images à classer de la même personne. Ce cas peut arriver avec l'utilisation par exemple d'une caméra qui prend plusieurs captures d'une personne. Le système possède alors des captures de la même personne avec des variations de luminosité, expressions faciales, pose,...
L'article est organisé comme suit : la section 2 détaille l'algorithme proposé pour la diffusion de labels et les deux fonctions de pénalité proposées pouvant être utilisées lors de la diffusion. Les expériences réalisées et les résultats obtenus sont décrits à la section 3. Dans la section 4 nous discutons de l'algorithme ainsi que ses liens avec d'autres méthodes de diffusion sur graphe. Finalement, nous concluons et présentons de futurs travaux à la section 5.

2 Diffusion de labels sur graphe

Dans cette section, nous rappellons quelques notations liées aux graphes, nous détaillons l'algorithme de diffusion proposé, et nous expliquons les différentes fonctions de pénalité utilisées conjointement à l'algorithme.

2.1 Notations and Définitions

Notions préliminaires sur les Graphes Nous considérons la situation générale où un domaine discret peut être modélisé par un graphe pondéré. Un graphe pondéré $G = (V, E, w)$ est composé d'un ensemble fini $V = \{u_1, \ldots, u_N\}$ de N *nœuds*, un ensemble d'*arêtes* $E \subset V \times V$, et une *fonction de poids* $w : E \to \mathbb{R}^+$. Une arête $(u, v) \in E$ connecte deux nœuds *adjacents* ou *voisins* u et v de V. Dans la suite, une telle arête est notée $u \sim v$. Nous supposons que le graphe G est simple, connecté et non orienté, ce qui implique que la fonction de poids w est symétrique ($w(u, v) = w(v, u)$ si $(u, v) \in E$).
Dans la suite, nous ne traitons que des graphes de distances. Le poids associé à une arête dans un graphe de distances représente simplement la distance entre deux nœuds selon une fonction de distance donnée.

La diffusion de labels La *ligne de partage des eaux* (LPE) est une transformation bien connue dans le domaine de la segmentation d'images. Intuitivement, la transformée LPE d'une fonction (vue comme une carte topographique) est composée de régions ou bassins reliés à un minimum local [6].
Largement utilisés en segmentation d'images, les algorithmes classiques utilisent des germes (calculés automatiquement ou fournis par l'utilisateur) comme pixels de départ. L'image de gradient est souvent utilisée comme image de potentiel, vue comme une carte de relief. Après application de l'algorithme, les pixels de l'image appartiennent tous à un bassin dérivé d'un germe initial, et sont donc localement groupés.
Cette transformée est donc une outil de *clustering* qui affecte à chaque pixel (dans le cas de la segmentation d'images) le label d'un germe initial auquel il est rattaché. Mimant cette transformée, nous proposons un algorithme similaire pour un graphe pondéré. Dans la suite, les données attachées à chaque nœud v correspondent à un vecteur correspondant aux caractéristiques faciales extraites d'un visage.

2.2 Algorithme proposé

Étant donné un graphe pondéré de distances G, nous proposons un algorithme simple (Algorithme 1) qui effectue une diffusion de labels sur graphe.
L'algorithme est initialisé avec un ensemble S de nœuds labélisés. Une fonction de label f est définie pour chaque nœud :

$$\begin{cases} f(u) > 0 & \forall u \in S \\ f(u) = 0 & \forall u \notin S \end{cases}$$

Algorithme 1 : Diffusion de labels sur graphe de distances.

Données : un graphe de distances G,
un ensemble S de nœuds labélisés $u, (f(u) > 0)$
Résultat : S
initialiser $f(u) = 0, \forall u \notin S$;
initialiser une liste L d'arêtes $e_{u\sim v}, \forall u \in S$;
tant que $L \neq \varnothing$ **faire**
  $e_{u\sim v} = \min_w L$;
  si $f(v) = 0$ **alors**
    $f(v) = f(u)$;
    $S = S \cup \{v\}$;
    $L = L \cup \{e_{k\sim v}\}$ $\forall k \sim v$ tel que $f(k) = 0$;
  $L = L - \{e_{u\sim v}\}$;

Toutes les arêtes contenant un élément de S sont ajoutées à la liste L des arêtes. À chaque étape, l'algorithme sélectionne l'arête $e_{u\sim v}$ de L dont le poids est minimal et qui possède un et un seul nœud non labélisé. Ce nœud non labélisé v connecté au nœud labélisé u prend alors le label de u : $f(v) = f(u)$. Les arêtes $e_{v\sim k}$ connectant les voisins non labélisés de v sont ensuite ajoutées à L.
La complexité de l'algorithme dépend essentiellement de la recherche de l'arête minimale. Cette partie peut efficacement être implémentée à l'aide d'un tas binaire. La complexité de la recherche de l'arête de poids minimal est alors en $\mathcal{O}(n \log(n))$ où n est le nombre d'arêtes contenues dans L.
Une représentation schématique du déroulement de l'algorithme est présentée à la figure 1. Dans cet exemple, les germes initiaux (couleurs rouge et bleu) sont associés aux nœuds a et b. En supposant que $w_1 < w_2 < w_3$, les nœuds b et c prennent tour à tour le label rouge de a.

Un exemple de segmentation obtenue pour une image de cytologie composée de plusieurs cellules est présentée à la figure 2. Dans cet exemple, chaque pixel (dans l'espace couleur RVB) est représenté par un nœud dans le graphe $u \in \mathbb{R}^3$, et la fonction de poids est donnée par $w(u, v) = \|u - v\|_2$. Le graphe construit pour cet exemple est le graphe des k–plus proches voisins (avec $k = 10$) qui connecte chaque nœud à ses k plus proches voisins selon la fonction de poids utilisée, et ce indépendamment de leur position dans la grille. Pour s'assurer qu'il n'existe pas de sous–graphes disjoints, chaque pixel de la grille a de plus été connecté à ses 4-voisins. L'algorithme détecte les trois cellules bien qu'une seule ait été initialement marquée.

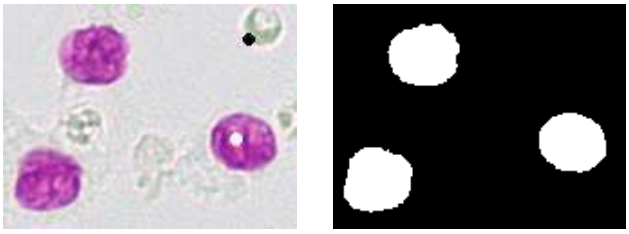

FIGURE 2 – Gauche : Image originale, les deux germes sont représentés par les points blanc et noir. Droite : Le résultat du clustering. Les trois cellules sont bien détectées.

2.3 Fonctions de pénalisation

L'algorithme proposé agit de manière locale sur les nœuds. Étant donné qu'un label propagé peut être moins pertinent qu'un label germe, nous proposons deux fonctions de pénalisation pour refléter la pertinence d'un label. Ces fonctions agissent sur les arêtes du graphe, et introduisent la notion de globalité dans le processus. Intuitivement, ces fonctions pénalisent les arêtes connectant deux nœuds initialement non labélisés, avec l'idée que plus une arête est « loin » d'un germe, plus elle sera pénalisée.
Nous commencons par introduire la notion de *pas* (noté $p_{u\to v} \in \mathbb{N}^*$) représentant pour un nœud donné u le nombre d'arêtes à traverser pour atteindre un nœud v. Ces pas peut facilement être mis à jour lors du déroulement de l'algorithme, lorsque les arêtes sont ajoutées à L. Les nœuds $u \in S$ ont un pas fixe $p_u = 0$. Dans l'exemple de la figure 1, $p_a = p_d = 0$, $p_{b\to a} = p_{c\to d} = 1$, et $p_{c\to a} = p_{b\to d} = 2$.
Les deux fonctions de pénalisation Ψ^1 et Ψ^2 proposées sont :

$$\begin{aligned} \Psi^1_{C,\gamma}(p) &= 1 + C(1 - \tfrac{1}{p^\gamma}) \\ \Psi^2_{C,\gamma}(p) &= 1 + C(1 - \exp(-\tfrac{1}{2}\tfrac{(p-1)^2}{\gamma^2})) \end{aligned}$$

Ces fonctions sont strictement croissantes selon p si $C > 0$, et sont définies sur $[1, +\infty[$ dans $[1, C + 1]$ (figure 3). Étant donné que ces fonctions ne sont utilisées que pour les nœuds u non labélisés, nous avons toujours $p_u > 0$. À noter que pour $C = 0$, nous avons $\Psi^1 = \Psi^2 = 1$, ce qui implique qu'aucune pénalisation n'est appliquée aux arêtes.
La pénalisation d'une arête $e_{u\sim v}$ ajoutée à L est ensuite calculée par :

$$w(u, v) \leftarrow \Psi^i_{C,\gamma}(p) \cdot w(u, v) \text{ avec } i \in \{1, 2\}$$

Ces fonctions permettent de contrôler la vitesse de diffusion. Lorsque $C = 0$, le processus est exactement celui décrit par l'algorithme 1. Si, pour $p > 1$), $\gamma \to +\infty$ pour Ψ^1 (ou $\gamma \to 0$ pour Ψ^2), les poids tendent alors vers $(C + 1) \cdot w(u, v)$.

3 Expérimentations et résultats

Dans cette section, nous détaillons la base de données utilisée comme *benchmark*, le prétraitement appliqué aux

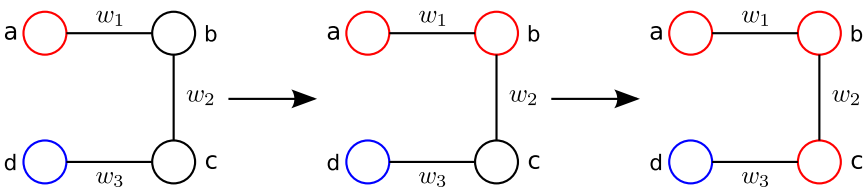

FIGURE 1 – Déroulement de l'algorithme pour un graphe où $w_1 < w_2 < w_3$.

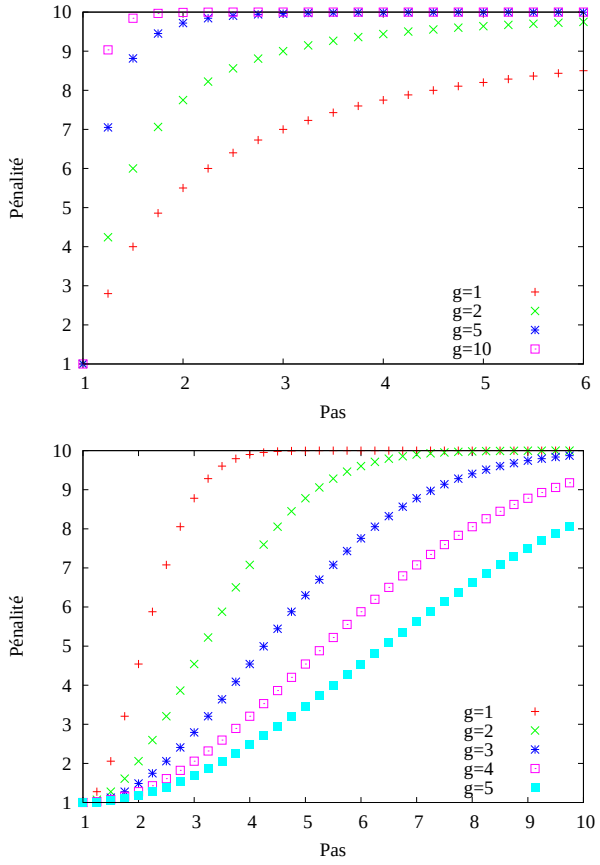

FIGURE 3 – Fonctions de pénalité Ψ^1 (haut) et Ψ^2 (bas) pour différentes valeurs de γ (noté g) et avec $C = 9$.

images, l'étape d'extraction des caractéristiques faciales, ainsi que les expériences menées.

3.1 Détails de la base de données

La base de données *Notre–Dame* [5] (voir la figure 4 pour quelques échantillons) est une base de données de référence disponible publiquement. Un protocole de test est livré, contenant des listes d'images à utiliser pour l'apprentissage/l'enrôlement/les tests. Deux expériences principales sont disponibles : *Same–session* et *Time–lapse*. Dans cet article, nous nous sommes concentrés sur l'expérience *Time–lapse* présentant des images prises à des intervalles de plusieurs semaines/mois. Cette expérience nous paraît plus pertinente que l'expérience *Same–session* présentant moins de difficultés.

L'expérience *Time–lapse* est composée de 16 sous–expériences, différant selon l'illumination et les expressions faciales des ensembles requêtes et des galeries (voir [5] pour plus de détails).

Chaque sous–expérience consiste à trouver l'identité de 431 images requêtes parmi une galerie composée de 63 images de 63 personnes.

Dans le reste de cet article, et pour des raisons de clarté, nous ne présentons qu'une moyenne des 16 taux d'identification au rang 1, bien que chaque sous–expérience ait été menée indépendamment.

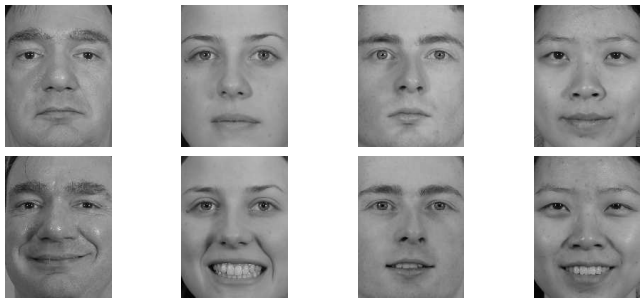

FIGURE 4 – Échantillons de la base de données *Notre–Dame*. Haut : quelques images de la galerie. Bas : quelques images requêtes.

3.2 Prétraitement

Chaque image de la base est normalisée avant l'extraction des caractéristiques. Les images sont normalisées géométriquement de sorte que les yeux se trouvent à la même position dans chaque image (voir la figure 5). Les images sont redimensionnnées à la taille 150×200, et les valeurs des pixels sont normalisées de sorte que leur moyenne soit nulle ($\mu = 0$) et leur écart–type unitaire ($\sigma = 1$).

3.3 Extraction des caractéristiques faciales

L'extraction des caractéristiques faciales consiste en une Analyse en Composantes Principales (méthode des *Eigenfaces* [15]). Chaque image est caractérisée par un vecteur $\Phi \in \mathbb{R}^n$ contenant les coefficients issus de la décomposition sur la base définie par les vecteurs propres retenus. Dans nos tests, les vecteurs propres représentant 95% de l'énergie totale induite par les valeurs propres issues de la décomposition sont retenus, ce qui correspond à des vecteurs caractéristiques de taille 97.

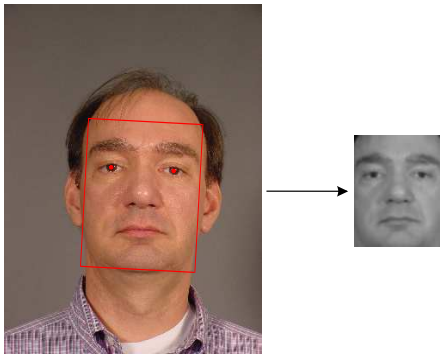

FIGURE 5 – Prétraitement des images.

3.4 Résultats de l'identification

Les tests d'identification menés consistent en la comparaison de la méthode de diffusion de labels au classifieur du plus proche voisin (*NN*) ainsi qu'à la méthode de classification fondée sur la parcimonie (*Sparse Representation–based Classification*, noté *SRC*) proposée dans [17].

Distances utilisées Quelques mesures de distance ont été testées pour la construction du graphe ainsi que pour le classifieur du plus proche voisin. Pour les distances relatives à l'espace de Mahalanobis (Équations 7, 8 et 9), les vecteurs caractéristiques u et v doivent être transformés de sorte que la variance selon chaque dimension soit unitaire (voir [2]). Soient m et n les vecteurs dans l'espace de Mahalanobis correspondant à u et v, ils sont définis par :

$$m_i = \frac{u_i}{\sigma_i} \quad (1)$$

$$n_i = \frac{v_i}{\sigma_i} \quad (2)$$

où σ_i représente l'écart–type selon chaque dimension.
Le lecteur intéressé pourra trouver de plus amples détails sur les équations 3 à 9 dans [2]. Les équations 10 et 11 sont détaillées respectivement dans [14] et [1].

$$D_{CityBlock}(u,v) = \sum_i |u_i - v_i| \quad (3)$$

$$D_{Euclidienne}(u,v) = \sqrt{\sum_i (u_i - v_i)^2} \quad (4)$$

$$D_{Corrélation}(u,v) = \frac{\sum_i (u_i - \bar{u})(v_i - \bar{v})}{(N-1)\sqrt{\frac{\sum_i (u_i-\bar{u})^2}{N-1}}\sqrt{\frac{\sum_i (v_i-\bar{v})^2}{N-1}}} \quad (5)$$

$$D_{Covariance}(u,v) = \frac{\sum_i u_i v_i}{\sqrt{\sum_i u_i^2}\sqrt{\sum_i v_i^2}} \quad (6)$$

$$D_{MahL1}(u,v) = \sum_i |m_i - n_i| \quad (7)$$

$$D_{MahL2}(u,v) = \sqrt{\sum_i (m_i - n_i)^2} \quad (8)$$

$$D_{MahCosine}(u,v) = \frac{m \cdot n}{|m||n|} \quad (9)$$

$$D_{Hellinger}(u,v) = \sqrt{\sum_i \left(\sqrt{|u_i|} - \sqrt{|v_i|}\right)^2} \quad (10)$$

$$D_{Canberra}(u,v) = \sum_i \frac{|u_i - v_i|}{|u_i + v_i|} \quad (11)$$

Résultats Le tableau 1 présente les principaux résultats obtenus. Pour chaque distance, les moyennes des taux d'identification au rang 1 (ainsi que l'écart–type associé) sont présentées. Les résultats pour la méthode de diffusion de labels sur graphe (*Diff. Lab.* dans le tableau) sont décomposés en deux selon la fonction de pénalisation utilisée (Section 2.3).

Étant donné que la méthode *SRC* fondée sur la parcimonie ne peut être déclinée selon les distances considérées, ses résultats n'ont pas été inclus dans le tableau. Pour cette méthode, la moyenne des taux de reconnaissance au rang 1 est de 60.70% (avec un écart–type de 9.81).

Pour la méthode de diffusion de labels sur graphe, une recherche par grille a été effectuée sur les deux paramètres C et γ pour Ψ^1 et Ψ^2. Le meilleur jeu de paramètres (celui donnant le meilleur taux d'identification) est noté ϕ.

Du tableau 1, nous pouvons constater que la méthode de diffusion donne toujours de meilleurs résultats que la classification par plus proche voisin, et ce, quelle que soit la distance utilisée.

Lorsqu'aucune pénalisation n'est utilisée lors de la diffusion (cas $C = 0$), les taux d'identification sont les pires. Cela montre clairement que les visages de différentes personnes peuvent être proches dans l'espace de projection défini par les vecteurs propres issus de l'ACP, ce qui indique que la méthode des *eigenfaces* n'est pas optimale pour la séparation des classes de visages.

4 Discussion

Les graphes construits dans toutes nos expériences sont complètement connectés : chaque nœud est connecté à tous les autres. Nous avons préféré ce type de graphe étant donné que nous n'avons fait aucune supposition sur la distribution des vecteurs caractéristiques dans l'espace propre. D'autres types de graphes peuvent néanmoins être considérés comme le graphe des k plus proches voisins (similaire à celui utilisé dans l'exemple présenté à la figure 2). Pour de tels graphes, il faut cependant faire attention à ce qu'il n'existe pas de sous–graphes disjoints.

La méthode de diffusion de labels présente des liens avec les méthodes sur les graphes comme l'algorithme bien connu de Dijkstra [7] ou encore la méthode plus générale fondée sur l'équation eikonale [13]. Une différence notable

	NN	*Diff. Lab.* Ψ^1	*Diff. Lab.* Ψ^2
Eq.3	69.76% (10.44)	74.59% (9.24) $\phi = (1,1)$	**75.20%** (9.53) $\phi = (4,3)$
		$C = 0$: 52.43% (17.71)	
Eq.4	59.09% (8.17)	**65.74%** (8.24) $\phi = (1,1)$	65.61% (8.72) $\phi = (1,1)$
		$C = 0$: 42.24% (15.16)	
Eq.5	58.06% (8.27)	**65.84%** (8.83) $\phi = (2,1)$	65.83% (9.31) $\phi = (2,1)$
		$C = 0$: 41.79% (14.99)	
Eq.6	58.55% (8.22)	65.71% (8.58) $\phi = (2,1)$	**65.82%** (8.84) $\phi = (2,1)$
		$C = 0$: 42.32% (15.04)	
Eq.7	68.14% (11.89)	73.53% (11.05) $\phi = (1,1)$	**75.50%** (11.82) $\phi = (5,4)$
		$C = 0$: 59.61% (20.24)	
Eq.8	71.70% (11.65)	76.42% (10.38) $\phi = (1,1)$	**78.19%** (10.80) $\phi = (10,5)$
		$C = 0$: 58.91% (21.33)	
Eq.9	74.23% (10.57)	**80.48%** (8.90) $\phi = (1,2)$	80.32% (9.62) $\phi = (3,2)$
		$C = 0$: 60.78% (21.91)	
Eq.10	54.81% (11.77)	58.40% (11.90) $\phi = (1,2)$	**62.83%** (11.46) $\phi = (3,3)$
		$C = 0$: 43.31% (18.81)	
Eq.11	41.83% (12.14)	42.57% (12.33) $\phi = (1,1)$	**49.20%** (16.06) $\phi = (1,3)$
		$C = 0$: 34.71% (16.37)	

TABLE 1 – Taux moyens d'identification au rang 1, écarts–types entre parenthèses. ϕ : paramètres (C, γ) utilisés avec les fonctions de pénalisation Ψ^1 et Ψ^2. $C = 0$: Taux d'identification obtenus sans pénalisation. Meilleur taux par distance en gras. Taux moyen obtenu avec la méthode *SRC* : 60.70% (écart–type : 9.81).

réside dans le fait qu'elle ne calcule pas explicitement de distance entre deux nœuds.

L'algorithme de Dijkstra calcule la somme de poids minimale permettant de relier un nœud non labélisé à un germe. Dans le cas particulier d'un graphe complètement connecté, cela revient à la classification par plus proche voisin (pour l'exemple du graphe de la figure 1, nous avons en effet $w_{a\to c} \leq w_{a\to b} + w_{b\to c}$). La méthode de classification présentée diffère également de la méthode fondée sur l'équation eikonale par le fait qu'elle ne traite que les arêtes du graphe, et ne calcule pas de gradient sur les nœuds par exemple. De plus, l'algorithme proposé dans [13] résolvant l'équation eikonale nécessite de nombreuses itérations pour converger, ce qui peut être lent en pratique.

Les paramètres C et γ des fonctions de pénalisation jouent un rôle significatif pour les taux finaux d'identification. Bien que les meilleurs jeux de paramètres pour les résultats du tableau 1 aient été trouvé via une recherche par grille, d'autres combinaisons permettent également d'obtenir de bons résultats. Les tableaux 2 et 3 montrent les taux d'identification obtenus pour la distance relative à l'équation 9 pour différentes valeurs de C et γ selon respectivement les fonctions de pénalisation Ψ^1 et Ψ^2. Ces tableaux montrent qu'il existe quelques couples « efficaces » permettant d'obtenir de bons taux d'identification. Ils permettent de plus d'apprécier l'évolution des taux finaux d'identification selon l'évolution des paramètres C et γ.

Le schéma de classification proposé peut être adapté facilement aux graphes de similarité. Pour un tel graphe, le poids ($w \in [0, 1]$) d'une arête représente le degré de similarité des nœuds qu'elle relie. Dans l'algorithme, l'arête à extraire devient celle dont le poids est maximal ($e_{u\sim v} = \max_w L$). Les fonctions de pénalisation Ψ^1 et Ψ^2 doivent également être adaptées de sorte qu'elles soient décroissantes selon p.

5 Conclusion

Nous avons présenté une méthode générique de classification appliquée au problème de l'identification de visages. Inspirée de la ligne de partage des eaux, cette méthode est basée sur une diffusion de labels sur graphe. Les échantillons de la galerie sont considérés comme les germes initiaux, l'algorithme affecte alors à chaque vecteur requête un label correspondant à une identité. Testé sur la base de données Notre–Dame, l'algorithme utilisé conjointement aux fonctions de pénalisation proposées permet d'obtenir de meilleurs taux d'identification que ceux obtenus via les autres méthodes de classification ayant été utilisées comme bases de comparaison. Par souci de clarté, nous n'avons cependant testé cette méthode qu'avec des vecteurs caractéristiques faciaux issus d'une ACP, ce qui n'est pas optimal pour la reconnaissance faciale. De futures expérimentations seront menées sur d'autres types de caractéristiques, avec d'autres topologies pour la construction du graphe, ainsi que sur d'autres modalités biométriques et d'autres données.

Références

[1] D. Androutsos, K. Plataniotis, and A. Venetsanopoulos. Distance measures for color image retrieval. *International Conference on Image Processing*, pages 770–774, 1998.

[2] J. R. Beveridge, D. S. Bolme, B. A. Draper, and M. Teixeira. The CSU face identification evaluation system : Its purpose, features, and structure. *Machine Vision and Applications*, 16(2) :128–138, February 2005.

[3] Karsten M. Borgwardt and Hans-Peter Kriegel. Shortest-path kernels on graphs. In *ICDM*, pages 74–81. IEEE Computer Society, 2005.

[4] R. Brunelli and T. Poggio. Face recognition : Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10) :1042–1052, 1993.

[5] X. Chen, P. J. Flynn, and K. W. Bowyer. IR and visible light face recognition. *Computer Vision and Image Understanding*, 99(3) :332–358, September 2005.

[6] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed cuts : Minimum spanning forests and the drop of

γ \ C	1	2	3	4	5	10
1	**79.88%**	79.71%	78.61%	77.71%	76.87%	74.85%
2	**80.48%**	78.95%	77.43%	76.46%	75.78%	74.28%
5	**80.24%**	78.24%	76.81%	75.91%	75.10%	74.28%
10	**80.27%**	78.13%	76.87%	75.65%	75.02%	74.28%

TABLE 2 – Taux d'identification au rang 1 obtenus pour la distance de l'équation 9 pour différentes valeurs de C et γ pour la fonction de pénalisation Ψ^1. Meilleur taux par ligne en gras.

γ \ C	1	2	3	4	5	10
1	**80.14%**	**79.81%**	78.91%	77.91%	76.87%	75.07%
2	74.85%	78.34%	**80.32%**	**80.16%**	**80.06%**	78.55%
3	72.02%	74.60%	76.39%	78.58%	78.92%	**80.08%**
4	69.76%	72.72%	74.21%	75.23%	76.18%	79.71%
5	68.88%	70.70%	72.70%	73.79%	74.55%	78.04%
6	67.92%	69.69%	70.95%	72.50%	73.33%	75.89%
7	67.32%	68.83%	70.20%	70.93%	72.23%	74.72%
8	66.41%	68.35%	69.17%	70.21%	70.76%	73.83%
9	65.45%	67.76%	68.76%	69.18%	70.21%	72.78%
10	64.96%	67.40%	68.34%	68.90%	69.41%	72.14%

TABLE 3 – Taux d'identification au rang 1 obtenus pour la distance de l'équation 9 pour différentes valeurs de C et γ pour la fonction de pénalisation Ψ^2. Meilleur taux par colonne en gras.

water principle. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(8) :1362–1374, August 2009.

[7] D. B. Johnson. A note on Dijkstra's shortest path algorithm. *Jrnl A.C.M.*, 20(3) :385–388, July 1973.

[8] Risi Imre Kondor and John Lafferty. Diffusion kernels on graphs and other discrete input spaces, April 21 2002.

[9] D. J. Kriegman, J. P. Hespanha, and P. N. Belhumeur. Eigenfaces vs. fisherfaces : Recognition using class-specific linear projection. In *European Conference on Computer Vision*, pages I :43–58, 1996.

[10] László Lovász. Random walks on graphs : A survey, 1993.

[11] S. Mika and J. Weston. Fisher discriminant analysis with kernels. *Neural Networks for Signal Processing*, pages 41–48, May 06 1999.

[12] B. Scholkopf, A. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, pages 1299–1319, 1998.

[13] V.T. Ta, A. Elmoataz, and O. Lézoray. Adaptation of eikonal equation over weighted graphs. In *International Conference on Scale Space and Variational Methods in Computer Vision*, LNCS 5567, pages 187–199. Springer, 2009.

[14] R. Taylor. A user's guide to measure-theoretic probability. *Journal of the American Statistical Association*, volume : 98(462), June 2003.

[15] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuro Science*, 3(1) :71–86, 1991.

[16] L. Wiskott, J. M. Fellous, N. Kruger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7) :775–779, July 1997.

[17] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 :210–227, 2008.