Apprentissage par méthodes à noyaux en reconnaissance d'images

Alberto Bietti

Table des matières

Introduction **2**

1 Apprentissage par méthodes à noyaux **2**
1.1 Position du problème et motivation 2
1.2 Méthodes à noyaux 3
1.3 Résolution 4

2 Application à la reconnaissance d'images **4**
2.1 Noyaux pour images 4
2.1.1 Noyaux d'histogrammes 4
2.1.2 Noyaux de *features* 5
2.1.3 Noyaux sur des graphes de segmentation 5
2.2 Exemples et résultats 5

Conclusion **6**

Introduction

Aujourd'hui, les nouveaux médias nous donnent accès à une énorme quantité d'images. Il est usuel d'avoir des milliers de photos sur son ordinateur, et le nombre d'images sur Internet est quasiment infini. L'un des problèmes qui surgit est la difficulté qui apparaît dans l'organisation et la recherche de ces images. Contrairement à la recherche de texte, qui est désormais performante, la recherche et la compréhension des images reste un défi. Néanmoins, les algorithmes actuels les plus performants commencent à reconnaître des scènes ou des objets avec une précision acceptable.

Pour reconnaître un objet précis dans une image – c'est ce qu'on appelle la *reconnaissance d'instances* –, il existe aujourd'hui des méthodes de programmation performantes qui cherchent à identifier l'objet cherché en analysant l'image et en trouvant des éléments d'intérêt. Mais ces méthodes ne sont pas adaptées à la *reconnaissance de classes*, qui cherche plutôt à reconnaître et à distinguer des catégories d'images (par exemple classer un ensemble d'image selon qu'elles contiennent un visage, un paysage, une chaise ou autre chose). Il s'agit d'un problème de *classification*.

Pour ce problème-ci, les techniques d'apprentissage sont plus adaptées : le programme va alors apprendre à classer de nouvelles images à partir d'un grand nombre d'exemples d'images fourni. Il peut s'agir d'apprentissage *supervisé*, lorsque les images d'entrée sont étiquetées par leur classe, ou *non supervisé* lorsque le programme trouve lui-même ces catégories.

Dans mon TIPE, je me suis intéressé plus particulièrement à l'apprentissage par méthodes à noyaux, très utilisé en reconnaissance d'images, et notamment pour la classification. J'ai alors implémenté un programme C++ de reconnaissance d'images en utilisant l'intersection d'histogrammes comme noyau, à l'aide de la bibliothèque de vision OpenCV.

1 Apprentissage par méthodes à noyaux

1.1 Position du problème et motivation

On se place dans le contexte de l'apprentissage supervisé, en particulier les problèmes de classification binaire (deux classes) et de régression (trouver une courbe qui passe par certains points). On se donne des *observations* $(x_i, y_i)_{i=1,\dots,n} \in (\mathcal{X} \times \mathcal{Y})^n$, où $\mathcal{X}$ est l'ensemble de départ (dans le cas de la reconnaissance d'images, c'est les images) et $\mathcal{Y}$ l'ensemble des résultats ($\mathcal{Y} = \{\pm 1\}$ pour la classification binaire, $\mathcal{Y} = \mathbb{R}$ pour la régression), et on veut prévoir $y \in \mathcal{Y}$ à partir de $x \in \mathcal{X}$ de façon cohérente avec les observations, à l'aide d'une fonction $f : \mathcal{X} \to \mathcal{Y}$. On veut alors que f minimise

$$\underbrace{\sum_{i=0}^{n} \ell(y_i, f(x_i))}_{\text{erreur sur les données}} + \underbrace{\frac{\lambda}{2}\|f\|^2}_{\text{régularisation}}$$

où $\ell(y, f(x))$ est une fonction de perte qui mesure l'erreur entre y et $f(x)$ et qu'on prendra égale à la perte quadratique $\frac{1}{2}(y - f(x))^2$ pour faciliter les calculs d'optimisation qui vont suivre (d'autres normes se révèlent être plus efficaces mais compliquent l'optimisation), et $\|.\|$ est une norme qu'on prendra égale à la norme 2 (norme euclidienne) dans le cadre des méthodes à noyaux. Le paramètre $\lambda > 0$ dirige le compromis biais-variance (voir la figure 1), dont les cas extrêmes donnent une fonction trop régularisée qui généralise trop (λ grand, à gauche) ou une fonction trop proche des observations, incapable de généraliser ($\lambda \to 0$, à droite).

Pour la classification binaire, on peut prendre aussi $\mathcal{Y} = \mathbb{R}$ (la classe sera donnée par le signe de $y \in \mathcal{Y}$), et on est ramenés à la même situation que pour la régression.

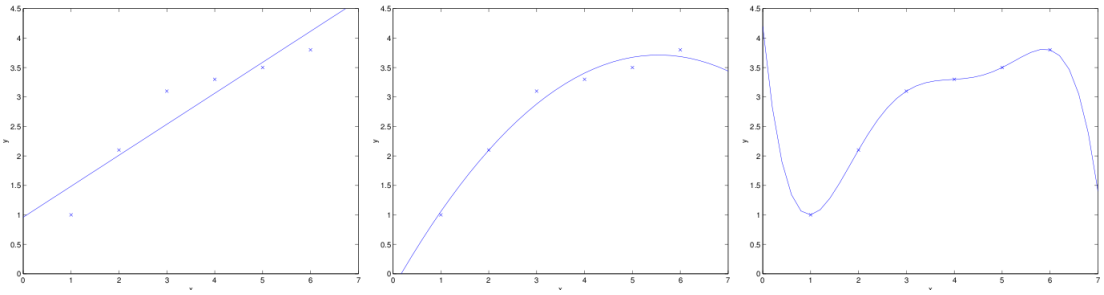

FIGURE 1 – Compromis biais-variance

Le cas linéaire Avec $\mathcal{X} = \mathbb{R}^p$, on cherche f de la forme $f(x) = w^\top x$, avec $w \in \mathbb{R}^p$. Il suffit alors de trouver w qui minimise $J(w) := \frac{1}{2}\|y - Xw\|^2$, où $y \in \mathbb{R}^n$ est le vecteur des y_i et $X \in \mathbb{R}^{n\times p}$ la matrice dont les lignes sont les $x_i^\top$, ce qui est aisé car la fonction est convexe différentiable et il suffit de chercher un point de gradient nul : on obtient $w = (X^\top X)^{-1} X^\top y$.

Motivation A ce stade, on aimerait franchir la barre de la linéarité pour faire de l'apprentissage non-linéaire, en partant d'objets autres que des vecteurs (par exemple, des images). C'est là qu'interviennent les noyaux, qui permettent de ramener le problème étudié aux ressemblances entre deux objets de $\mathcal{X}$ donnés, sans recourir à l'objet lui-même.

1.2 Méthodes à noyaux

Plutôt qu'apprendre à partir de l'objet-même, ce qui serait peu pratique et pas très efficace dans l'exemple d'une image, on effectue l'apprentissage à partir de certaines caractéristiques de l'objet, appelées *features*, données par un *"feature map"* $\phi : \mathcal{X} \to \mathcal{F}$. $\phi(x)$ peut être un vecteur ($\mathcal{F} = \mathbb{R}^p$) contenant, par exemple, des puissances de x pour représenter des polynômes, ou des caractéristiques intéressantes de l'objet x si $\mathcal{X}$ est arbitraire (pour les images, voir 2.1).

On cherche alors f linéaire sur les features, de la forme $f(x) = w^\top \phi(x)$, et le problème d'optimisation devient le suivant :

$$\min_{w\in\mathbb{R}^p} \sum_{i=1}^{n} \ell(y_i, w^\top \phi(x_i)) + \frac{\lambda}{2}\|w\|^2$$

Le *théorème du représentant* assure que la solution doit être de la forme $w = \sum_{i=1}^n \alpha_i \phi(x_i)$. En notant K la matrice $[k(x_i, x_j)]_{i,j} \in \mathbb{R}^{n\times n}$ (*matrice du noyau*), où $k(x_i, x_j) = \phi(x_i)^\top \phi(x_j) = \langle \phi(x_i), \phi(x_j)\rangle_{\mathcal{F}}$, le problème équivaut à résoudre :

$$\min_{\alpha\in\mathbb{R}^n} \sum_{i=1}^{n} \ell(y_i, (K\alpha)_i) + \frac{\lambda}{2}\alpha^\top K\alpha$$

Sous cette nouvelle forme, on voit que le feature map ϕ n'apparaît plus, et que seule la matrice K est utilisée. Seuls interviennent alors les termes $k(x_i, x_j)$ qui correspondent à un produit scalaire d'éléments de $\mathcal{F}$ et en ce sens mesurent la ressemblance entre x_i et x_j. De fait, les $k(x_i, x_j)$ sont habituellement calculés directement à partir de noyaux connus, sans l'intervention de ϕ : c'est l'*astuce du noyau* (*"kernel trick"*). L'espace $\mathcal{F}$ peut alors être de dimension infinie sans que cela ne pose de problème, et on effectue implicitement de l'apprentissage non-linéaire.

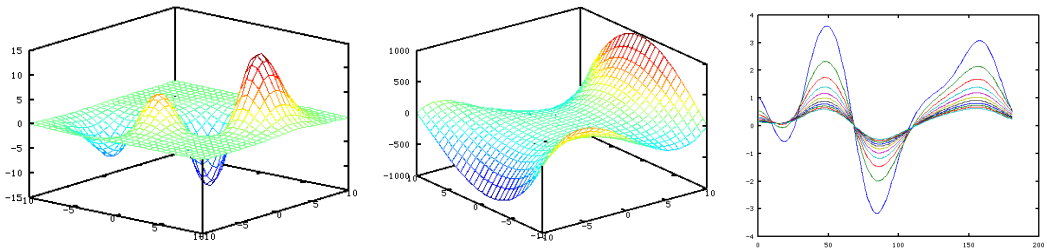

FIGURE 2 – Régression à l'aide de noyaux

Des exemples de noyaux couramment utilisés sont le noyau polynomial $k(x,y) = (1 + x^\top y)^d$ ($\mathcal{F}$: polynômes) ou le noyau gaussien $k(x,y) = e^{-\alpha\|x-y\|^2}$ ($\mathcal{F}$: fonctions $\mathcal{C}^\infty$). Des exemples de noyaux pour images sont présentés dans la partie 2.1.

1.3 Résolution

Dans le cas de la perte quadratique, on est amenés à résoudre le problème suivant : trouver $\alpha \in \mathbb{R}^n$ qui minimise

$$J(\alpha) := \frac{1}{2}\|y - K\alpha\|^2 + \frac{\lambda}{2}\alpha^\top K\alpha$$

La fonction J est différentiable, et $\nabla J(\alpha) = K^\top(K\alpha - y) + \lambda K\alpha$. En supposant que k est un noyau défini positif, donc que K est symétrique positive (par définition), la solution est $\alpha = (K + \lambda I)^{-1} y$ (à un élément du noyau de K près, qui ne modifie pas f) et on a alors $f(\cdot) = \sum_{i=1}^n \alpha_i k(\cdot, x_i)$.

A titre d'exemple, j'ai implémenté un programme de "régression" ou "génération" de surface en Matlab/Octave qui crée une fonction passant par certains points fixés, avec un noyau gaussien (figure 2, à gauche) ou polynomial (figure 2, au centre). L'image de droite de la figure 2 montre une régression similaire, mais pour une courbe à une dimension, avec différents paramètres de régularisation λ.

2 Application à la reconnaissance d'images

2.1 Noyaux pour images

Il existe plusieurs types de noyaux pour la reconnaissance d'images. J'en présente trois.

2.1.1 Noyaux d'histogrammes

Les noyaux d'histogrammes reposent sur la notion d'intersection d'histogrammes d'images. Il s'agit ici d'histogrammes représentant la distribution des couleurs d'une image : à chaque couleur (ou intervalle de couleurs) i ($i = 1, \ldots, D$) on associe le nombre de pixels de cette couleur, $h_x(i)$ (pour une image x). Certains les dénomment *bags of pixels* ("sacs de pixels"), ce qui indique ce regroupement de pixels. L'intersection de deux histogrammes est donnée par

$$\mathcal{I}(h_x, h_y) = \sum_{i=1}^{D} \min(h_x(i), h_y(i))$$

et on définit alors le noyau suivant, dont on peut montrer qu'il est défini positif :

$$k(x,y) = \mathcal{I}(h_x, h_y)$$

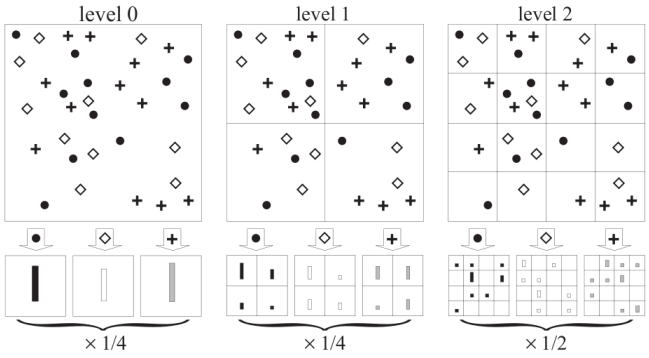

FIGURE 3 – Pyramide d'histogrammes (source : [3])

2.1.2 Noyaux de *features*

On peut construire des noyaux d'intersection sur d'autres structures que des histogrammes de couleurs. Un exemple est de prendre des *bags of features* à la place des *bags of pixels*. Le terme *feature* désigne un élément d'intérêt ou un point saillant d'une image. Les *features* sont en général obtenues par des algorithmes de traitement d'image (*feature extraction*). Les plus célèbres sont les descripteurs SIFT ([5]), très performants en reconnaissance d'instances. En classant les *features* obtenues sur une image par ressemblance dans des "sacs" de features, on obtient des histogrammes de *features* et on peut utiliser un noyau d'intersection comme celui du paragraphe précédent.

D'autres approches consistent, comme dans [3], à effectuer ces mêmes opérations d'intersection, mais sur différentes grilles de résolutions différentes (avec des facteurs différents) pour obtenir un *"pyramid match kernel"* (voir figure 3). Ceci permet de mieux tenir compte de la position des *features* les unes par rapport aux autres.

2.1.3 Noyaux sur des graphes de segmentation

Une autre façon de procéder est de partir d'une segmentation de l'image, par exemple à l'aide de l'algorithme watershed (ligne de partage des eaux), comme dans [2]. Une fois l'image divisée en plusieurs segments de même couleur, on peut construire un *graphe de segmentation* (voir figure 4) et utiliser ensuite des noyaux sur ces graphes, que je ne vais pas étudier en détail (on peut construire des noyaux sur un graphe à partir de chemins [1], de marches [2] ou de sous-arbres, voir [2]).

2.2 Exemples et résultats

J'ai implémenté en C++ un programme de reconnaissance d'image utilisant le noyau d'intersection d'histogrammes de la partie 2.1.1. Le programme crée la matrice du noyau K à partir des observations (images et classe ± 1) lues sur un fichier texte, et prend en paramètre une nouvelle image x dont il renvoye la valeur $f(x)$. Le signe de cette valeur en indique la classe. La figure 6 page 8 donne quelques résultats pour une classification de scènes intérieures/extérieures, à partir d'une quarantaine d'observations (voir figure 5).

1. suite de sommets voisins distincts
2. suite de sommets voisins

FIGURE 4 – Ligne de partage des eaux et graphe de segmentation (source : [2])

Conclusion

Les méthodes à noyaux se révèlent donc très adaptées aux problèmes d'apprentissage relatifs à la classification en reconnaissance d'images, avec de nombreuses qualités tels que la non-linéarité et la très grande dimension de l'espace. Mais elles ne sont pas sans inconvénients.

Problèmes des méthodes présentées L'interprétation des résultats est souvent difficile à cause de la grande dimension. Il est difficile de tenir compte de la structure d'une image et de la position des objets analysés dans la construction d'un noyau pour images. Enfin, si la classification binaire se fait bien, la classification multi-classe reste un défi (on passe en général par plusieurs classificateurs binaires "un contre tous").

Etat actuel de la recherche Les méthodes à noyaux sont très répandues aujourd'hui, dans de nombreux domaines dont la reconnaissance d'images, la bio-informatique ou l'intelligence artificielle. Elles sont utilisées surtout pour les *machines à vecteur de support* (SVM), qui utilisent des fonctions de perte plus efficaces pour des problèmes d'optimisation plus compliqués. J'ai rencontré Francis Bach, chercheur de l'INRIA et du Laboratoire d'Informatique de l'École Normale Supérieure (Willow project), qui m'a parlé des recherches actuelles dans le domaine, s'appuyant sur les normes ℓ^1 plutôt que sur les normes ℓ^2 des méthodes à noyaux. Celles-ci entraînent l'apparition de valeurs nulles (c'est pourquoi on les appelle *méthodes parcimonieuses*), ce qui présente certains avantages tels qu'une meilleure visualisation et interprétation des résultats et des variables d'intérêt, mais d'autres difficultés apparaissent, notamment au niveau de l'optimisation non-différentiable.

Références

[1] F. Bach, *Méthodes à noyaux en apprentissage statistique*, Colloque RASMA, 2008.

[2] Z. Harchaoui, F. Bach, *Image classification with segmentation graph kernels*, Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), 2007.

[3] S. Lazebnik, C. Schmid, J. Ponce, *Beyond bags of features : spatial pyramid matching for recognizing natural scene categories*, CVPR, 2006.

[4] B. Schölkopf, A. J. Smola, *Learning with Kernels*, MIT Press, 2002.

[5] D. G. Lowe, *Object recognition from local scale-invariant features*, International Conference on Computer Vision, 1999.

FIGURE 5 – Quelques images d'entrée pour une classification intérieur/extérieur. Première ligne : classe +1 (intérieur). Deuxième ligne : classe −1 (extérieur)

FIGURE 6 – Quelques résultats. Il y a tout de même une erreur : la dernière image donne un résultat négatif, alors que c'est une scène d'intérieur.