APPLICATIONS SVM EN AUDIO/PAROLE

A Arias

arias@irit.fr

1. INTRODUCTION

Les Machines à Vecteurs de Support (SVM) ont démontré une très bonne performance dans les tâches de classification [Bur98]. Nèanmoins, les problèmes que cette méthode résout de manière efficace sont appelés «statiques», car tous les vecteurs de son espace d'entrée doivent avoir la même dimension et «binaires», parce que l'idée originale est de séparer deux classes par un hyperplan. La formulation SVM original [Vap95] présente aussi certaines limitations car elle suppose que les échantillons d'apprentissage et de test sont indépendants et identiquement distribués (i.i.d.) [PC00].

Si l'on veut appliquer les SVM au traitement de signal, on doit dépasser les limitations du modèle de base car le signal audio est dynamique et variable [RJ93]. On doit par exemple, être capable de représenter des séquences d'observation de taille variable par des vecteurs de taille fixe, obtenir des décisions de classification à partir de problèmes multiclasses et adapter la solution SVM quand les échantillons ne sont pas i.i.d.

Construire un classificateur qui produise des probabilités *a posteriori* est aussi souhaitable dans certaines tâches de reconnaissance [Pla00], où les sorties de la classification doivent être combinées. Malheureusement le modèle SVM standard ne fournit pas ces probabilités.

Divers efforts ont été menés pour résoudre ces limitations et développer des applications SVM en traitement de signal.

2. EXPLOTATION DES MODÈLES GÉNÉRATIFS DANS LES CLASSIFICATEURS DISCRIMINANTS

Jaakkola et Haussler ont proposé une combinaison entre les modèles génératifs de probabilité et les méthodes de classification discriminants [JH99]. L'idée est de prendre avantage des ceux-là pour transformer des séquences de taille variable en vecteurs de taille fixe et des ceux-ci pour construire les frontières entre classes. De cette manière on incorpore au critère de décision une certaine connaissance du processus qui a généré les données.

On a donc un modèle génératif parametrisé θ d'un ensemble d'exemples et on veut trouver un moyen de comparer ces échantillons. On sait qu'une fonction noyau permet de comparer deux vecteurs [PB04], alors on va utiliser ces fonctions pour établir des relations de distance. D'abord, on passe à l'espace du gradient du modèle de probabilité : le gradient de la vraisemblance d'une séquence par rapport à un paramètre décrit comment ce paramètre contribue au processus de génération de la séquence.

Une classe paramétrique de modèles de probabilité $P(X \mid \theta), \theta \in \Theta$ génére la transformation :

$$\phi(X, \theta) = \nabla_\theta \log P(X \mid \theta)$$

On appele à $\phi(X, \theta)$ «score de Fisher» et à $\log P(X \mid \theta)$ la vraisemblence Λ. Le score de Fisher représente la transformation d'une séquence $X \rightarrow \phi(X, \theta)$ en un vecteur dans l'espace de caractéristiques. Chaque composante du vecteur ϕ est la derivée de la vraisemblance de la séquence X par rapport à un paramètre particulier du modèle. Sa valeur montre comment ce paramètre contribue à la génération de la séquence. Chaque composante a un rang dynamique différent, donc le noyau est défini comme un produit scalaire normalisé par I, la matrice d'information de Fisher. Le noyau permet une comparaison basique entre séquences.

$$K(\theta, X_i, X_j) = \phi(X_i, \theta)^T I^{-1} \phi(X_j, \theta)$$

I est :

$$I = E\{\phi(X, \theta)\phi(X, \theta)^T\}$$

De cette manière on peut utiliser le noyau de Fisher pour trouver un hyperplan séparateur entre séquences dans l'espace de caractéristiques. Si les exemples ne sont pas lineairement séparables on utilise des frontières de décision non lineaires telles que :

$$K'(X_i, X_j) = (1 + K(X_i, X_j))^q \text{ ou autres}$$

La transformation implicite dans le noyau de Fisher est étudié dans [Smi03] et ses propriétés de régularisation sont montrées en [OSS00].

Exemple. Si on a une classe modelisée par une distribution Gaussienne normale unidimensionnelle avec paramètres $\theta = (\mu, \sigma)$:

$$P(x \mid \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Le *log* de la vraisemblance d'un point x en fonction des paramètres θ est :

$$\Lambda_{(\mu,\sigma)} = -\frac{(x-\mu)^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)$$

Le score de Fisher $\phi(x \mid \theta)$ du Λ par rapport à μ est :

$$\phi(x \mid \theta_\mu) = \nabla_\mu \Lambda = \frac{(x-\mu)}{\sigma^2}$$

Et par rapport à σ :

$$\phi(x \mid \theta_\sigma) = \nabla_\sigma \Lambda = \frac{(x-\mu)^2}{\sigma^3} - \frac{1}{\sigma}$$

Si $\mu = 0$ et $\sigma = 1$, le vecteur crée en utilisant la moyenne est : $\phi(x) = (x)$, et celui crée par la variance est : $\phi(x) = (x^2 - 1)$, ce qui équivaut à une expansion polynomiale de x.

2.1. Utilisation du noyau de Fisher pour la classification audio

Dans [MR00] on décrit un système de discrimination «Parole/Musique/Autre» basé sur le noyau de Fisher.

Un modèle génératif GMM est donc supposé modéliser les trois classes de données. Au lieu de prendre des décisions de classification directement à partir des GMM (avec le critère de maximum de probabilité *a posteriori*), le modèle GMM est utilisé pour générer

des «scores»[JH99], c'est à dire, des vecteurs de taille fixe qui répresentent des séquences audio de taille variable téléchargées sur Internet. Les scores sont considerés comme des vecteurs de caractéristiques des séquences et sont classifiés par une SVM multiclasse (les deux approches classiques, one vs. one et one vs. all ont été mises en œuvre pour éviter des ambiguïtés).

Chaque dimension du score est la derivée du logarithme de la vraisemblence d'une sequence X particulière par rapport à un paramètre du modèle génératif (probabilités *a priori*, moyennes ou covariances). Dans l'application, trois GMM de 68 composants ont été calculés et puis combinés en un seul modèle. Les scores obtenus avec le paramètre covariance ont été écartés car les résultats ont été pauvres, seuls les scores derivés des probabilités *a priori* et de la moyenne ont été utilisés avec des résultats comparables.

3. ARCHITECTURE HYBRIDE SVM/HMM

Dans ce système hybride [GHP98], un HMM triphone avec multiple GMM par état fonctionne comme un pré-traitement pour segmenter les séquences d'entrée (génération d'un alignement à niveau phonémique). Chaque segment identifié est divisé en proportion 30-40-30 et une moyenne est calculée pour chaque partie (figure 1). Le segment est ainsi transformé en un vecteur de taille fixe. Ensuite, un classificateur SVM multiclasse distingue chaque phonème.

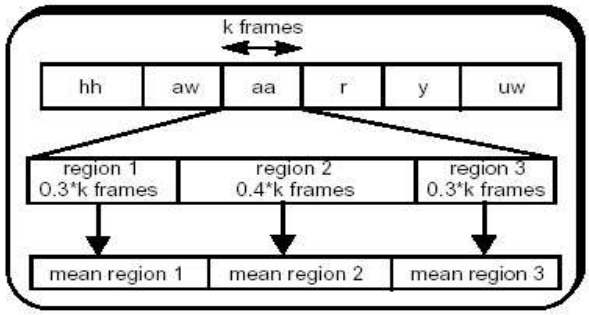

Fig. 1. Exemple de la construction d'un vecteur de caractéristiques.

Les résultats de la classification SVM sont transformés en probabilités *a posteriori* en utilisant une fonction sigmoïde [Pla00] et la meilleure séquence est choisie en utilisant l'algorithme de Viterbi. La modification du système décrit dans [HPG02] utilise une «Relevance Vector Machine (RVM)» au lieu d'une SVM. La sortie d'une RVM est une probabilité donc accorde naturellement à l'architecture hybride.

3.1. D'autres combinaisons

Dans le contexte des SVM en reconnaissance du locuteur, Campbell [Cam02] a proposé une fonction noyau de séquences basé sur l'analyse discriminante linéaire généralisé pour produire les scores utilisées dans ce cadre. Une évolution du système inclut un HMM pour segmenter les séquences d'entrée.

Fine *et al.* [FNG01] ont combiné les modèles GMM et SVM pour la vérification du locuteur. Ils considèrent que les erreurs de ces classificateurs ne sont pas corrélés. Leur hypothèse est que les résultats obtenus avec les GMM sont essentiellement corrects et robustes au changements d'environnement acoustique entre les phases d'apprentissage et de test. Un système fondé sur SVM et les scores de Fisher (calculés avec le UBM du système

GMM) est utilisé seulement en cas d'indécision des scores GMM (figure 2). L'approche multiclasse utilisé dans l'évolution du système en tâches de reconnaissance de chiffres [FSG02] est basé sur la théorie de codes de correction d'erreurs, ainsi chaque résultat de classification est codé dans un mot binaire et traité comme s'il s'agissait d'une information transmise sur un canal de communication. On utilise le critère de la distance de Hamming pour résoudre problèmes d'indécisions dans la classification.

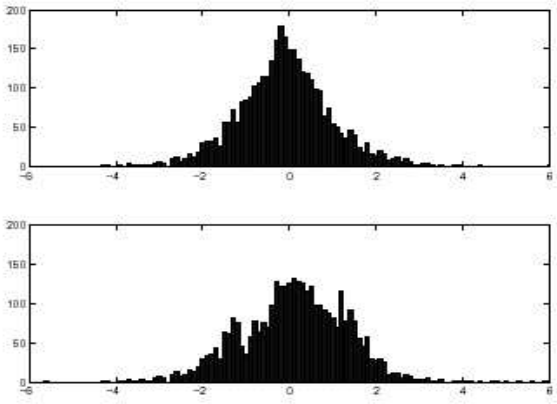

Fig. 2. Modification de la distribution du rapport de vraisemblance en utilisant des SVM. Avant (normal - dessus) et après (binomial - dessous).

4. SUPPRESSION DE CERTAINS VECTEURS MFCC

Dans ce projet [BK00], les SVM ont été utilisés pour reconnaître les chiffres (1-10) isolés. Chaque chiffre (en apprentissage et reconnaissance) a la même taille, donc ils sont directement comparables. La procedure qui transforme une séquence de taille variable à un vecteur de taille fixe est la suivante :

1. Une paramétrisation acoustique de la séquence est calculée (MFCC-14 dimensions).
2. La distance euclidienne entre trames voisines est calculée. Les vecteurs qui ne changent pas trop sont enlevés iterativement jusqu'a en avoir 30 par mot. On garde donc les vecteurs de paramètres les plus «représentatifs» de chaque mot.
3. Une analyse PCA est realisée et la dimensionalité de la séquence est réduite de 420 (30x14) à 45.
4. Un système SVM multiclasse 1-versus-9 donne 94.9 % d'exactitude. Pour résoudre les conflits de classification on analyse les valeurs de distance entre les vecteurs en question et leurs hyperplans séparateurs.

En travaillant avec des sons monophones de durée variable [LSTI02], Lee *et al.* ont égalisé la taille des sons de deux manières : (a) au moyen de la division de chaque segment en un nombre fixe de subsegments d'égal durée plus le calcul de la moyenne sur ceux-ci et (b) grâce à l'élimination des vecteurs en segments longs ou l'insertion des vecteurs dupliqués en segments courts. On garde aussi la durée du segment comme une variable pour le classificateur. Les auteurs ont considéré que le système multiclasse one vs. all est très asymétrique et que le système one vs. one demande trop de classificateurs, donc ils on proposé une variante à ces méthodes appelée «close-class set». L'idée est de créer ensembles de classes proches et faire d'abord une première discrimination entre groupes et après identifier une classe à l'intérieur du sous-ensemble.

5. ALIGNEMENT DYNAMIQUE

Cette approche [SNNS02] modifie la formulation recursive typique de l'alignement dynamique (DTW) et au lieu de minimiser la mesure de distance/distorsion on maximise la similarité en utilisant le produit scalaire et en general une fonction noyau.

Si deux séquences sont répresentées par les suites de vecteurs X_i et X_j, l'algorithme DTW minimise la distance cumulée $G(i,j)$ allant du vecteur $(1,1)$ au (I,J) au moyen d'une fonction «warping» qui génére un chemin optimal entre l'axe temporel de X_i et X_j.

$$G(i,j) = \min \left\{ \begin{array}{c} G(i-1,j) + d(i,j) \\ G(i-1,j-1) + 2d(i,j) \\ G(i,j-1) + d(i,j) \end{array} \right\}$$

La nouvelle formulation utilise un produit scalaire :

$$G(i,j) = \max \left\{ \begin{array}{c} G(i-1,j) + < i,j > \\ G(i-1,j-1) + 2 < i,j > \\ G(i,j-1) + < i,j > \end{array} \right\}$$

Dans les deux cas on respecte les conditions de continuité et monotonie. Les experimentations ménées pour tester la fonction d'alignement SVM reconnaissent un groupe de 6 consonnes dans un premier temps et de 5 voyelles dans un deuxième temps. Elle est plus performante qu'un système HMM de référence quand le nombre d'échantillons d'entraînement par phonème est de 50 et 100. Quand on dispose de 200 échantillons d'entraînement par phonème, le système HMM avec 16 MMG par état est d'une performance légerement supérieure.

6. SVM MULTICLASSE

Le SVM a été conçu comme un classificateur binaire, néanmoins, il y a différentes manières d'appliquer l'algorithme de base en problèmes de classification qui concernent k classes.

Le problème désormais se pose de la façon suivante : étant donné un ensemble d'échantillons $(x_1, y_1), \ldots, (x_i, y_i)$, $x \in \Re^d$ et $y_i = \{1, \ldots, k\}$, il faut trouver une fonction de décision tel que $f : X \rightarrow Y$.
Différents solutions ont été proposées :

- Utiliser k «one vs. all» classificateurs. C'est la solution la plus simple. k classificateurs binaires sont construits, un pour chaque classe. Le $k^{ième}$ classificateur sépare les données de la classe k de tout le reste de données d'apprentissage. Pour prendre une décision de classification, on garde la classe qui a eu la valeur maximale de toutes les fonctions de décision.

- Utiliser $\frac{k(k-1)}{2}$ «one vs. one» classificateurs. Dans ce cas là, chaque classificateur est entraîné avec un sous-ensemble des données d'apprentissage qui contient les vecteurs de deux classes. Les classificateurs sont combinés dans un système de vote pour définir une sortie de classification. Pour un exemple x de test, si le classificateur C_{ij} dit que x appartient à la classe y_i, le vote pour la classe y_i est incrémenté en 1. Cette stratégie assigne x à la classe qui a reçu le plus grand nombre de votes.
  Hastie et Tibshirani [HT98] ont proposé une variante appelée «pairwise coupling». Les probabilités *a posteriori* générées à partir de la sortie des classificateur binaires

sont utilisées pour faire la classification multiclasse. Une p.d.f. Gaussienne de chaque classe est estimé avec les valeurs de la fonction de décision de tous les exemples de la classe.

– Étendre la formulation SVM pour gérer les k-classes au même temps [WW99] [BB99] [CS01]. C'est une manière naturel de résoudre le problème multiclasse. Dans ce cas, une fonction de décision est calculé pour chaque classe. Le problème d'optimisation SVM est généralisé et il considère toutes les fonctions de décision au même temps. Les contraintes sont aussi relâchées. Au lieu de forcer les fonctions de décision à avoir une valeur zéro sur la frontière de décision, c'est désormais suffisant que la fonction pour la classe correcte aie une valeur plus grande que pour le reste des classes.
Le problème avec cette formulation est que, d'abord, les résultats de la classification ne sont nettement améliorés et en plus, l'optimisation devient trop compliquée.

7. REFERENCES

[BB99] E. J. Bredensteiner and K. P. Bennet. Multicategory classification by support vector machines. In *Computational Optimizations and Applications*, pages 53–79, 1999.

[BK00] I. Bazzi and D Katabi. Using support vector machines for spoken digit recognition. In *International Conference on Spoken Language Processing*, Beijing, China, October 2000.

[Bur98] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 1998.

[Cam02] W. M. Campbell. Generalized linear discriminant sequence kernels for speaker recognition. In *International Conference on Audio, Speech and Signal Processing*, Orlando, USA, May 2002.

[CS01] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. In *Journal of Machine Learning Research*, pages 265–292, 2001.

[FNG01] S. Fine, J. Navrátil, and R. A. Gopinath. Enhancing gmm scores using svm hints. In *European Conference On Speech Communication And Technology*, Aalborg, Denmark, September 2001.

[FSG02] S. Fine, G. Saon, and R. A. Gopinath. Digit recognition in noisy environments via a sequential gmm/svm system. In *International Conference on Audio, Speech and Signal Processing*, Orlando, USA, May 2002.

[GHP98] A. Ganapathiraju, J. Hamaker, and J. Picone. Support vector machines for speech recognition. In *International Conference on Spoken Language Processing*, pages 2923–2926, Sydney, Australia, November 1998.

[HPG02] J.E. Hamaker, J. Picone, and A. Ganapathiraju. A sparse modeling approach to speech recognition based on relevance vector machines. In *International Conference on Spoken Language Processing*, volume 3, pages 1001–1004, Denver, USA, September 2002.

[HT98] T. Hastie and R. Tibshirani. Classification by pairwise coupling. In *Advances in Neural Information Processing Systems*, volume 10, 1998.

[JH99] T. Jaakkola and D.. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, pages 487–493. MIT Press, 1999.

[LSTI02] W. Lee, C. Sekhar, K. Takeda, and F. Itakura. Recognition of continuos speech segments of monophone units using support vector machines. In *International Conference on Spoken Language Processing*, volume 3, pages 2653–2656, Denver, USA, September 2002.

[MR00] P. Moreno and R. Rifkin. Using the fisher kernel method for web audio classification. In *International Conference on Audio, Speech and Signal Processing*, pages 2417–2420, Istanbul, Turkey, June 2000. IEEE.

[OSS00] N. Oliver, B. Schölkopf, and A. Smola. Natural regularization from generative models. In *Advances in Large Margin Classifiers*. MIT Press, 2000.

[PB04] F. Perez and O. Bousquet. Kernel methods and their potential use in signal processing. *IEEE Signal Processing Magazine*, 21(3), May 2004.

[PC00] F. Pérez Cruz. *Máquina de Vectores Soporte Adaptativa y Compacta*. PhD thesis, Universidad Politécnica de Madrid, 2000.

[Pla00] J. Platt. Probabilities for sv machines. In *Advances in Large Margin Classifiers*. MIT Press, 2000.

[RJ93] L. Rabiner and B. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.

[Smi03] N. Smith. *Using Augmented Statistical Models and Score Spaces for Classification*. PhD thesis, Christ's College, 2003.

[SNNS02] H. Shimodaira, K. Noma, M. Nakai, and S. Sagayama. Dynamic time-alignment kernel in support vector machine. In *Advances in Neural Information Processing Systems*, pages 921–928. MIT Press, 2002.

[Vap95] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[WW99] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Seventh European Symposium on Artificial Neural Networks*, April 1999.