Réduction du temps d'apprentissage des SVM par Quantification Vectorielle

Gilles LEBRUN Christophe CHARRIER Olivier LEZORAY

LUSAC EA 2607, groupe Vision et Analyse d'Image

120 Rue de l'exode, F-50000 Saint-Lô, France

{gilles.lebrun, c.charrier, Olivier.Lezoray}@chbg.unicaen.fr

Résumé

Une nouvelle méthode pour entraîner les SVM avec des bases d'apprentissage de taille importante est présentée. L'idée principale est d'utiliser la quantification vectorielle pour remplacer les exemples par un ensemble réduit de prototypes. L'objectif est de diminuer fortement le temps d'entraînement pour choisir les paramètres optimaux. Des résultats d'expérimentations montrent que les temps d'entraînement peuvent être réduis par un facteur de 100 tout en préservant le taux de classification. De plus, cette méthode permet de trouver une fonction de décision réduite lorsque les données sont bruitées.

Mots clefs

SVM, Quantification Vectorielle

1 Introduction

Les Machines à Vecteurs de Support (SVM) [1] sont des outils très performants pour la reconnaissance des formes. Cependant l'utilisation des SVM est difficile lorsque la taille de la base d'apprentissage est très élevée. La première raison est due à une recherche des paramètres du modèle appliqué. En effet, le temps d'apprentissage augmente rapidement avec la taille du problème et l'éloignement des paramètres des valeurs optimales. La seconde raison est liée à la complexité de la fonction de décision qui augmente avec le nombre d'exemples présentés pendant la phase d'apprentissage. La classification d'un flux de données continu peut ainsi devenir rapidement problématique. La méthode proposée utilise la technique de la Quantification Vectorielle (QV) pour réduire la taille de la base d'apprentissage. Chaque vecteur produit par la QV représente alors un prototype résumant un ensemble d'exemples de la base initiale. Un schéma d'apprentissage itératif augmente progressivement la taille de la base d'apprentissage et élimine les modèles qui ne peuvent correspondre à l'optimal.

2 Machine à Vecteurs de Support

La théorie de l'Apprentissage Statistique de Vapnik et de Chervonenkis [1] a conduit au développement d'une classe d'algorithmes connus sous le nom de SVM. Ils permettent de réaliser des estimations en classification (et en régression). Une des originalités de la méthode est de produire une fonction de décision qui n'utilise qu'un sous-ensemble de la base d'apprentissage. Les éléments de ce sous-ensemble sont nommés Vecteurs de Support (**SV**).
Soit un ensemble d'apprentissage $A = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ composé de m couples (vecteur d'attributs, label) avec $x_i \in \mathcal{R}^n$ et $y_i \in \{-1, +1\}$. L'algorithme des SVM projette les vecteurs x_i dans un espace de travail $\mathbf{H}$ à partir d'une fonction non linéaire $\phi : \mathcal{R}^n \rightarrow \mathbf{H}$. L'hyperplan optimal de séparation des deux classes dans l'espace $\mathbf{H}$ est ensuite recherché. Cet hyperplan $(\mathbf{w}, b)$ matérialise la frontière de séparation entre les deux classes. La classe y d'un nouvel exemple $\mathbf{x}$ est définie par :

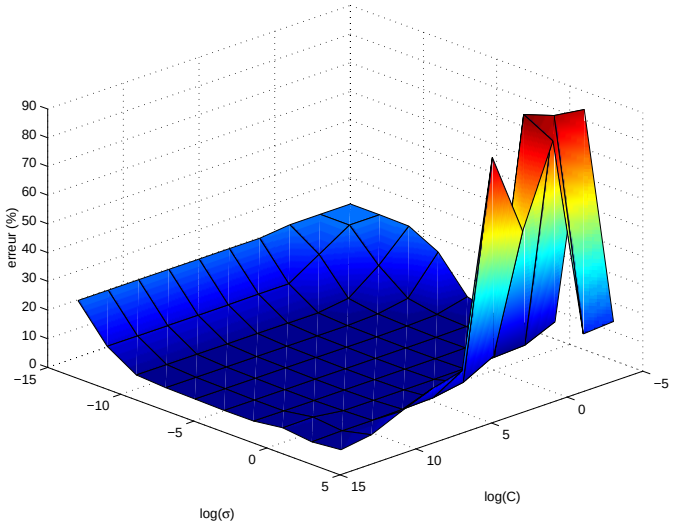
$$y = sign\left(\mathbf{w} \cdot \phi(\mathbf{x}) + b\right) \tag{1}$$

L'hyperplan est optimal s'il maximise la distance qui le sépare des exemples lui étant le plus proche. Cette distance est usuellement appelée marge du classifieur. Il a été démontré [1] que maximiser cette marge correspond à maximiser le «pouvoir» généralisateur du classifieur. Comme la valeur de la marge est inversement proportionnelle à la norme de $\mathbf{w}$, la recherche de l'hyperplan optimal est équivalente à la minimisation de $\mathcal{W}(\mathbf{w}, b, \xi) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C\left(\sum_{i=1}^{m} \xi_i\right)$ en respectant les contraintes $\forall_{i=1}^{m} : y_i \left[\mathbf{w} \cdot \phi\left(\mathbf{x}_i\right) + b\right] \geq 1 - \xi_i\,,\ \xi_i \geq 0$. Le compromis entre erreurs de classification et complexité du modèle dépend du paramètre C et des variables ξ_i. Sous forme duale le problème revient à minimiser $\mathcal{W}(\alpha) = \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K\left(\mathbf{x_i}, \mathbf{x_j}\right) - \sum_{i=1}^{m} \alpha_i$ sous les contraintes $\forall_{i=1}^{m} : \sum_{i=1}^{m} y_i \alpha_i = 0\,,\ 0 \leq \alpha_i \leq C$. L'avantage de la version duale est de diminuer le nombre de variables (suppression des $\xi_\mathbf{i}$) et d'éviter sa résolution dans l'espace de projection $\mathbf{H}$. La technique de simplification utilise une fonction noyau K respectant l'égalité

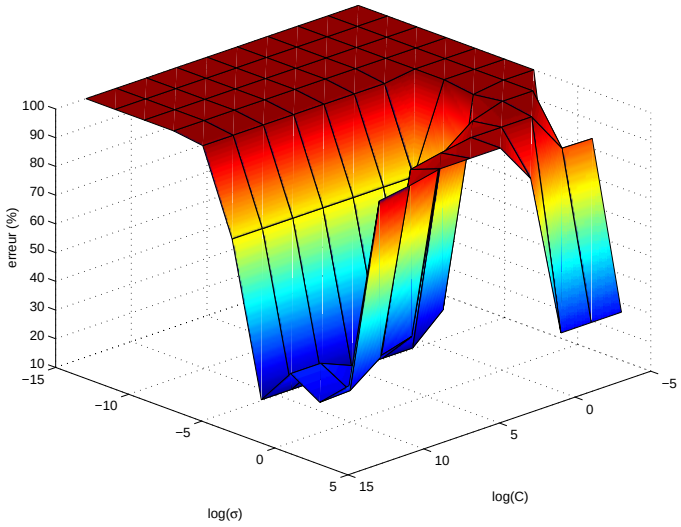
$$K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left\langle \phi\left(\mathbf{x}_i\right), \phi\left(\mathbf{x}_j\right) \right\rangle \tag{2}$$

La solution optimale α^* définit l'ensemble $\mathbf{SV}$ des vecteurs de support ($\forall_{i=1}^{m} i \in \mathbf{SV} : \alpha_i > 0$) et permet de définir la fonction de décision [1] :

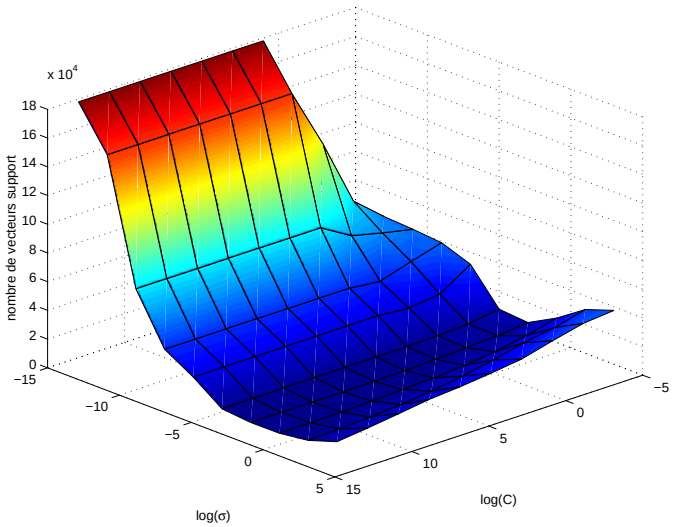
$$f\left(\mathbf{x}\right) = sign\left(\sum_{i \in SV} \alpha_i^* y_i K\left(\mathbf{x_i}, \mathbf{x}\right) + b^*\right) \tag{3}$$



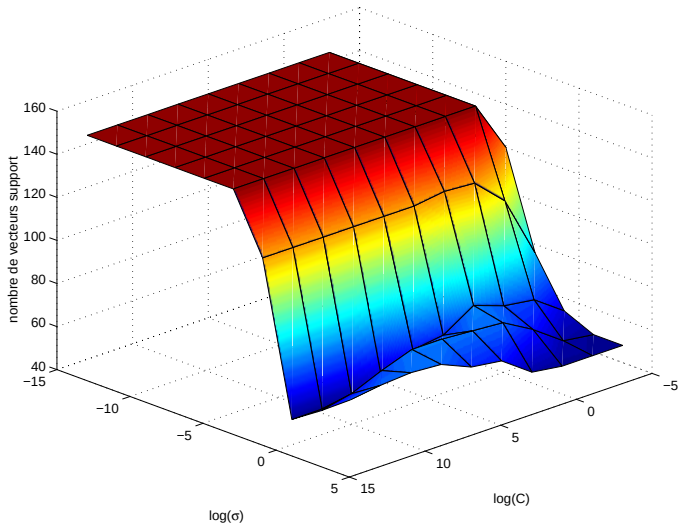
(a) Shuttle - Taux d'erreur.



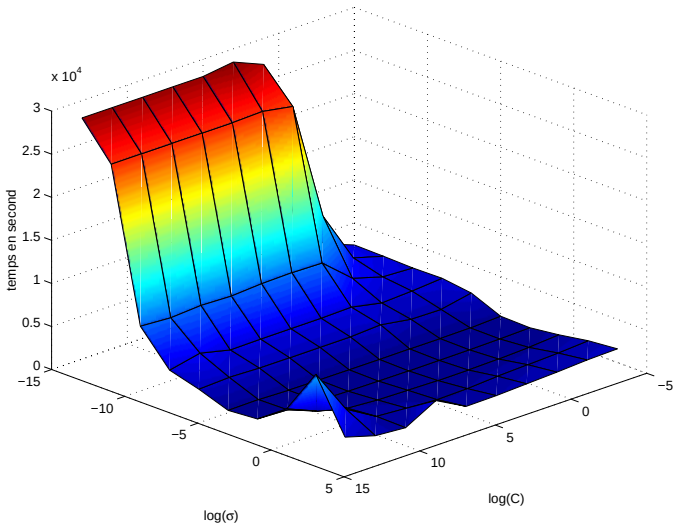
(b) Prototype shuttle - taux d'erreur.



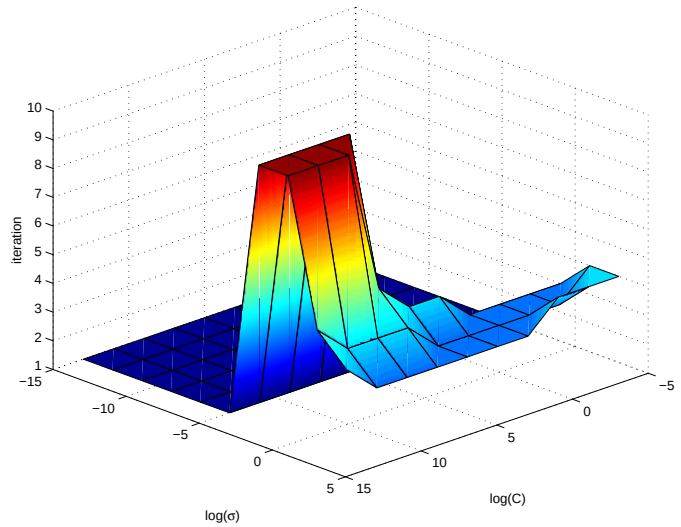
(c) Shuttle - nsv.



(d) Prototype shuttle - nsv.



(e) Shuttle - durée apprentissage (s).



(f) Iteration de rejet.

Figure 1 – *Résultats expérimentaux*

Un algorithme efficace SMO [2] a été proposé par Platt pour résoudre le problème dual. Les SVM étant des classifieurs binaires, la résolution d'un problème multi-classes est réalisé en le transformant en une combinaison de problèmes binaires [3].

3 Méthode hybride d'apprentissage

En étudiant l'algorithme des SVM on remarque facilement que la complexité associée (en temps de calcul) augmente rapidement en fonction de la taille de la base d'apprentissage (pour la version SMO elle se situe entre $O(n)$ et $O\left(n^2\right)$ [2]). De plus, le nombre de vecteurs de support augmente avec la taille du problème [2]. Cette augmentation est d'autant plus importante que les données du problème sont bruitées. En effet, lorsque le problème n'est pas linéairement séparable dans l'espace de projection $\mathbf{H}$, plusieurs variables ξ_i sont actives ($\xi_i > 0$). Les vecteurs de support associés sont alors dans la marge. Sous l'hypothèse que la position de l'hyperplan ainsi que la taille de la marge ne change pas, la proportion d'exemples dans la marge (et à moindre mesure sur la marge) augmente avec la taille du problème. Cette hypothèse est justifiée si la taille de la base initiale est suffisamment importante et que tout nouvel exemple n'ajoute pas d'information. Lorsque la taille de la base d'apprentissage est élevée, la difficulté est alors d'extraire seulement les exemples significatifs.

L'idée principale de notre méthode est d'entaîner les SVM sur une base représentative de la base initiale, mais avec un nombre d'exemples réduit. L'algorithme LBG [4] (utilisé lors de l'étape de construction du dictionnaire dans le cadre de la QV) est appliqué afin de réduire le nombre d'exemples. L'objectif est de diminuer le temps d'entraînement et de permettre rapidement le rejet des mauvais paramètres pour le choix du modèle. La taille de la base d'apprentissage augmente à chaque itération, jusqu'à ce que l'amélioration du taux de classification devienne inférieure à un seuil. Le noyau choisi est un noyau gaussien $\mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) = exp\left(-\frac{\|\mathbf{x_i}-\mathbf{x_j}\|^2}{2\sigma^2}\right)$. La méthode *un-contre-tous* [3] est utilisée pour transformer un ensemble de SVM binaires en un classifieur multi-classes. Le schéma itératif proposé est le suivant :

Soit $\mathbf{A} = \{(\mathbf{x}_i, y_i)\}_{1\leq i\leq m}$ la base d'apprentissage, $\mathbf{T} = \{(\mathbf{x}_i, y_i)\}_{1\leq i\leq m'}$ la base de test, $\mathbf{P} = \{(C_i, \sigma_i)\}_{1\leq i\leq l, 1\leq j\leq l'}$ l'ensemble des couples de paramètres pour le choix du modèle et k le nombre de classes différentes dans $\mathbf{A}$ et $\mathbf{T}$.

Etape 1 : $\mathbf{A}$ est séparé en k ensemble de vecteurs d'attributs $\mathbf{SB}_j = \{\mathbf{x}_i : y_i = j\}$ de taille s_j ($1 \leq j \leq k$).

Etape 2 : Pour chaque $\mathbf{SB}_j$ un dictionnaire $\mathbf{C}_j$ de taille 2^α est produit en appliquant l'algorithme LBG ($\mathbf{C}_j = \mathbf{SB}_j$ si $s_j \leq 2^\alpha$). L'union de tous les dictionnaires (en ajoutant le label de la classe correspondante) produit une nouvelle base d'apprentissage $A_\alpha = \bigcup_{j=1}^{k} \mathbf{C}_j \times \{j\}$.

Etape 3 : Les SVM sont entraînés à partir de $\mathbf{A}_\alpha$ pour chaque couple de paramètres $(C, \sigma) \in \mathbf{P}$. L'erreur empirique de classification $f_{(\mathbf{A}_\alpha, C, \sigma)}(\mathbf{x})$ est déterminée à partir de la base $\mathbf{T}$. L'erreur empirique minimale e^*, ainsi que le nombre minimal nsv^* de vecteurs de support pour les fonctions de décision f, sont déterminés à partir de cet ensemble de tests.

Etape 4 : Tous les couples (C, σ) dont la fonction de décision associée $f_{(\mathbf{A}_\alpha, C, \sigma)}(\mathbf{x})$ à une erreur empirique $e > \psi_1(e^*)$ ou un nombre de vecteurs de support $nsv > \psi_2(nsv^*)$ sont supprimés de P.

Etape 5 : α est incrémenté, puis les étapes 2 à 5 sont répétées, jusqu'à ce que l'amélioration du taux de classification ne soit plus significatif ou que $\mathbf{A}_\alpha = \mathbf{A}$.

Nous faisons l'hypothèse que l'étape 2 fournit des prototypes suffisamment représentatifs des exemples de la base d'apprentissage. La topologie des données de chaque classe est alors préservée. Un même modèle doit donc avoir des performances en généralisation de même qualité en utilisant la base d'apprentissage $\mathbf{A}$ ou $\mathbf{A}_\alpha$. Plusieurs critères existent pour estimer les performances des SVM [5]. Les plus couramment utilisés sont l'erreur empirique et le nombre de vecteurs de support. Comme ces critères ne fournissent qu'une estimation et pour éviter de rejeter trop rapidement des couples de paramètres qui pourraient être optimaux, les fonctions de rejet ψ_1 et ψ_2 permettent de définir un seuil de sécurité (dans notre étude $\psi_1(x) = \psi_2(x) = 2x$).

4 Expérimentations

Trois bases d'apprentissage comportant un nombre élevé d'exemples (Tableau 1) ont été utilisées : *satimage*, *letter* et *shuttle* [6]. Elles sont régulièrement utilisées commme référence pour la classification. La librairie *svmtorch* a été utilisée pour l'implentation des SVM multi-classes [7]. Les

base	apprentissage	test	classe	attributs
satimage	4435	2000	6	36
letter	15000	5000	26	15
shuttle	43500	14500	7	9

Tableau 1 – *description des bases.*

figures 1(a) et 1(c) mettent en évidence les fortes variations de e et nsv en fonction des valeurs de C et σ. La figure 1(e) montre que le temps d'apprentissage est fortement lié au nombre de vecteurs de support (nsv). La raison principale est que l'algorithme SMO utilise un cache et une technique de $shrinking$ [2] pour réduire les temps de calcul. L'efficacité de ces deux techniques diminues lorsque le nombre de vecteurs de support augmente. Les figures **??** et 1(d) montrent les résultats obtenus avec notre méthode ($\alpha = 4$). On remarque que les régions d'intérêt ont été correctement identifiées, mais que le relief des surfaces est plus raide. Ceci est du au fait que l'algorithme des SVM utilise un grand nombre de sv lorsque la taille de la base est importante et que les paramètres C et σ sont éloigné des valeurs

optimales. Ceci à pour effet de réduire le taux d'erreur mais correspond à un phénomène de sur-apprentissage. La figure 1(f) permet de déterminer à quelle itération un couple de paramètres a été rejeté. Elle illustre que les plus mauvais couples sont rejetés rapidement. Le tableau 2 donne les temps d'apprentissage et les taux d'erreur pour notre methode et la methode classique $grid\ search$ [8]. L'approche proposée réduit fortement les temps d'apprentissage sur des bases de grande taille. De plus, le taux d'erreur est quasi-identique à la méthode classique. La complexité des fonctions de décision produites sont par contre faiblement réduites. Le faible bruitage des données de ces trois bases explique cette faible réduction.

base	**notre méthode**		**méthode classique**	
	temps	**erreur**	**temps**	**erreur**
satimage	8.5	10.7	29.9	10.1
letter	19	6.34	827	6.34
shuttle	3.8	0.09	412	0.09

Tableau 2 – *taux d'erreur et temps d'apprentissage(heures).*

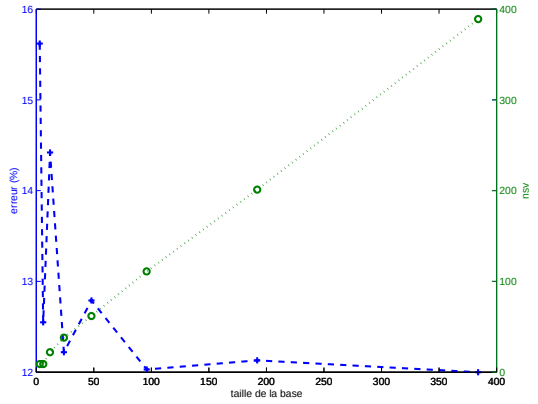


Figure 2 – *Evolution du taux d'erreur et de* nsv.

Nous avons aussi appliqué notre méthode à la classification de pixels issu d'images microscopiques [9]. Les bases d'apprentissage et de test sont construites à partir de 4 images couleurs microscopiques de tumeur bronchique (574*752 pixels). Pour chaque image, une segmentation manuelle (fond, cytoplasme et noyaux) est réalisée par un expert. La base d'apprentissage comporte plusieurs millions d'exemples (pixels) et les données sont fortement bruitées. La figure 2 montre qu'avec notre méthode le taux d'erreur ne varie plus avec la taille de la base au-delà d'un certain seuil. Cependant la complexité de la fonction de décision augmente linéairement avec la taille de la base. En effet le nombre d'exemples inclus dans la marge géométrique est important du fait du fort bruitage des données. Le nombre de vecteurs de support augmente donc proportionnellement à la taille de la base. La fonction de décision produite par notre méthode est 100 fois plus rapide que celle utilisée par MEURIE *et al.*[9] et la segmentation obtenue dans les deux cas est identique (la faible différence dans le taux de classification est annulée par l'ensemble des opérations morphologiques).

5 Conclusion

Une nouvelle méthode pour l'entraînement des SVM avec des bases de tailles importantes est introduite. L'idée principale repose sur l'utilisation de la QV pour construire une base d'apprentissage de taille réduite en produisant un ensemble de prototypes exemples. Les résultats exérimentaux obtenus montrent que la méthode proposée réduit fortement le temps d'apprentissage. De plus cette méthode produit une fonction de décision de complexité réduite lorsque les données sont bruitées.
Les travaux futurs devront étendre notre méthode à des fonctions noyau avec plus de paramètres [5]. En effet la technique de *grid search* peut difficilement être utilisée avec un nombre de paramètre supérieur à deux. Une étude de l'influence de notre méthode avec différentes techniques multi-classes et d'autres classifieurs à base de fonctions noyau sera également envisagée. [10].

Références

[1] V. N. Vapnik. *Statistical Learning Theory*. New York, wiley édition, 1998.

[2] J. Platt. *Fast Training of Support Vector Machines using Sequential Minimal Optimization, Advances in Kernel Methods-Support Vector Learning*. MIT Press, pp. 185-208, 1999.

[3] C-W. Hsu et C-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions in Neural Networks*, Vol 13, n° 2, pp. 415-425, 2002.

[4] A. Gersho et R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic, 1991.

[5] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, et Sayan Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3) :131–159, 2002.

[6] C. Blake et C. Merz. Uci repository of machine learning databases. advances in kernel methods, support vector learning. *University of California, Irvine, Dept. of Information and Computer Sciences*, 1998.

[7] Ronan Collobert et Samy Bengio. SVMTorch : Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1 :143–160, 2001.

[8] C.-C Chang et C.-J Lin. Libsvm : a library for support vector machines. Sofware Available at http ://www.csie.ntu.edu.tw/˜cjlin/libsvm, 2001.

[9] C. Meurie, G.Lebrun, O. Lezoray, et A. Elmoataz. A comparison of supervised pixels-based color image segmentation methods. application in cancerology. Dans *WSEAS Transactions on Computers*, volume Vol. 2, pages pp. 739–744, July 2003.

[10] M. E. Tipping. The relevance vector machine. *In Advances in Neural Information Processing Systems, San Mateo, CA*, 2000.