サポートベクターマシン：最適化からのアプローチ

津田　宏治

1. はじめに

サポートベクターマシンとは、９０年代に有名になったパターン認識の一手法である[18]。従来法、特に８０年代から９０年代にかけて注目されたMultilayer Perceptron（いわゆるニューラルネット）に比べて優れた結果が次々と報告され(例えば[13])、それに呼応して多くの理論的な研究もなされている。SVMに関する理論的な研究には、次のようなアプローチがある。

- 学習理論からのアプローチ: SVMの汎化誤差の推定や、それに基づく新しい学習基準の設定などを課題とする研究(例えば[20])。
- 関数解析からのアプローチ: SVMが表現する識別関数に注目し、再生核ヒルベルト空間の理論などと結びつけて議論する研究(例えば[19])。
- 最適化からのアプローチ: 学習に必要な最適化問題の高速な解法の提案や、「スパース」な解を導く学習法の研究など。

本誌の読者には最適化のエキスパートが多いと思われるので、本解説では、第三のアプローチを扱う。特に最近の機械学習の研究において一つのキーワードとなっているスパースな解を導く手法について中心的に述べる。他のアプローチに関しては、[21, 22]を参照していただきたい。

学習機械は、一般に複数のパラメータ（パラメータベクトル）を持つ。訓練サンプルが与えられたとき、学習手法によってパラメータベクトルの最適解が与えられる。ここで、スパースな解(Sparse Solution)とは、パラメータベクトルのうち、多くの要素の値が０になっている解のことを指す。このような解を出力する学習手法は、スパースな方法と呼ばれる。

スパースな解の利点は大きく分けて三つある。第一には、未知サンプルに対する出力の計算が容易になることである。例えば、SVMのように、カーネル関数の線形結合が識別関数であり、パラメータが各カーネル関数の重みであるような場合には、重みが０のところのカーネル関数は計算しなくていい。第二には、スパース性を用いることによって、高速な学習アルゴリズムが得られることである。SVMの最適化問題は、二次計画問題として定式化できるが、係数行列は一般に巨大でかつ密であるので、高速に解くためには、解のスパース性をうまく用いる必要がある。第三に、学習結果の解釈が容易になることである。例えば、複数の種類の特徴をまとめて学習する場合には、どの特徴の重みパラメータが非ゼロになったかを分析することによって、識別に重要な特徴を知ることができる。このような利点のために、他の非スパースな手法にスパース性を付加しようという研究が現在盛んである[8, 15, 16]。

本稿では、まず、SVMとそのスパース性についての説明から始め（第２章）、スパース性を生かして学習を高速化する方法について述べる(第３章)。さらに、第４章では、SVMのスパース性を向上させる方法について述べる。次に、学習法のスパース化の一例として、伝統的な識別法である線形判別分析のスパース化を取り上げる（第５章)。さらに、スパース性とは直接関係はないが、最適化問題との関連から、Bayes Point Machineを紹介する(第６章)。第７章はまとめである。

2. SVMとスパース解

2.1 線形SVM

まず線形SVMの定式化を行う。$x_1, \ldots, x_n \in \Re^d$を訓練サンプルとし、$y_1, \cdots, y_n \in \{1, -1\}$をクラスラベ

つだ　こうじ
産業技術総合研究所生命情報科学研究センター
〒135-0064　東京都江東区青梅二丁目41番地6

ルとする。線形SVMの識別関数は、パラメータベクトルを $w \in \Re^d, b \in \Re$ とすると、

$$f(x) = w^T x + b \tag{1}$$

と表される。学習においては、次の最適化問題を解く。

$$\min_{\boldsymbol{\alpha}} \frac{1}{n} \sum_{i=1}^{n} c(f(x_i), y_i) + \lambda \|w\|^2 \tag{2}$$

ここで、$\|w\|^2$ は、過学習を防ぐための正則化項であり、λ は、正則化パラメータと呼ばれる定数である。また、c は、Soft margin 損失関数

$$c(f(x), y) = \max(1 - yf(x), 0) \tag{3}$$

である。この問題の最適解においては、Representer定理[7]により、w は次のように分解できる。

$$w = \sum_{i=1}^{n} \alpha_i x_i \tag{4}$$

また、最適化問題(2)は、α をパラメータとする次の凸二次計画問題と等価となる。

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{subject to} \quad & 0 \le \alpha_i \le C,\ i = 1, \ldots, n, \qquad (5) \\ & \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0. \qquad (6) \end{aligned}$$

ここで、C は、λ の代わりに正則化の度合を制御するパラメータである。

2.2 カーネルトリックによる非線型への拡張

線形SVMは、2クラスが線形分離可能な場合には、高い認識率を達成できるが、実際にはそうでない場合が多い。そこで、前処理として、非線型な写像を用いて、より高次元の空間に写像を行い、線形分離性を高めることが考えられる。

$$\Phi : \Re^d \to \Re^q \tag{7}$$

ただし、この写像は、元の空間におけるサンプル同士の距離関係をある程度保存する必要がある。そうでないと、クラス内のまとまりが無くなって、識別が困難になるからである。そこで、元の空間で定義されるカーネル関数 $k(x, x')$ を用意して、Φ は次の条件を満たすと仮定する。

$$\Phi(x)^T \Phi(x') = k(x, x') \tag{8}$$

このような Φ が存在すると仮定し、$\Re^q$ において、SVMを適用しよう。このとき、識別関数は、

$$\begin{aligned} f(x) &= w^T \Phi(x) = \sum_{i=1}^{n} \alpha_i \Phi(x)^T \Phi(x') \qquad (9) \\ &= \sum_{i=1}^{n} \alpha_i k(x, x') \qquad (10) \end{aligned}$$

と書ける。また、学習の最適化問題は、次のようになる。

$$\min_{\boldsymbol{\alpha}} \frac{1}{n} \sum_{i=1}^{n} c(f(x_i), y_i) + \lambda \boldsymbol{\alpha}^T K \boldsymbol{\alpha} \tag{11}$$

ここで、K は、訓練サンプル間のカーネル関数の値から成る $n \times n$ 行列である。この最適化問題も、(2)と同様に二次計画問題に書きかえられる。

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ \text{subject to} \quad & 0 \le \alpha_i \le C,\ i = 1, \ldots, n, \qquad (12) \\ & \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0. \qquad (13) \end{aligned}$$

2.3 SVMのスパース性

Karush-Kuhn-Tucker条件から、問題(13)の最適解は、次の条件を満たす[18]:

$$\begin{aligned} \alpha_i = 0 \quad &\Rightarrow \quad y_i f(x_i) \ge 1 \\ 0 < \alpha_i < C \quad &\Rightarrow \quad y_i f(x_i) = 1 \\ \alpha_i = C \quad &\Rightarrow \quad y_i f(x_i) \le 1 \end{aligned} \tag{14}$$

この条件より、識別結果 $\mathrm{sgn} f(x)$ が y_i と一致していて、また、マージン値 $y_i f(x)$ が閾値1より大きいサンプルに対応する α_i は0になることがわかる。幾何的には、図1のように表される。図の○、□は、それぞれ異なるクラスの訓練サンプルを表す。また、実線は識別面を表し、破線は $f(x) = \pm 1$ の面を表す。$y_i f(x) = 1$ の面（図の点線）より境界側に存在する点に対応するパラメータのみが非ゼロとなり、残りのパラメータは0となる。図では、非ゼロになるパラメータに対応するサンプルを塗りつぶして示している。

一般に低次元空間に多くのサンプルがある場合には、0になるパラメータの数が多くなる傾向があるが、高次元に少数しかない場合には、0になるサンプルの割合は小さくなってしまうことが知られている。高次元空間で、よりスパースな解を得るためには、4.章で述べる方法を用いる必要がある。

3. SVMの学習における最適化手法

一般に、二次計画問題を高速に解く手法に関しては多くの文献があり、また、フリーや商用のパッケージ

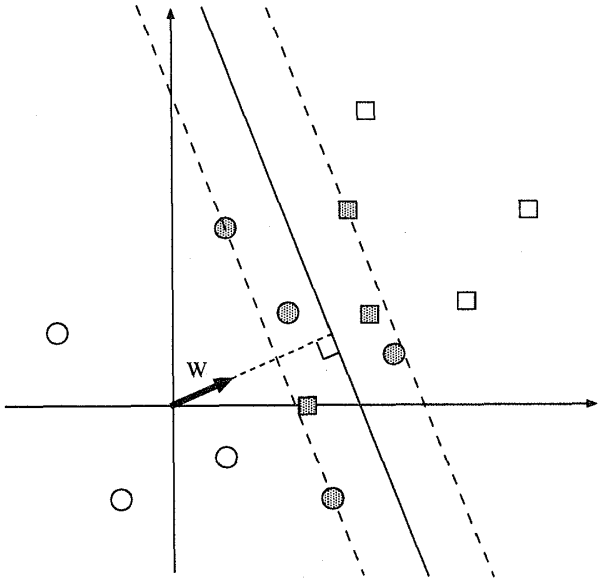

図 1: SVM のスパース性

も普及している [2]。しかし、従来の手法には、小規模な問題にしか適用できなかったり、また、係数行列が疎であることを仮定するものが多い。SVM においては、係数行列は巨大であり、また、疎でもない。このような困難にもかかわらず、少ない記憶量で、高速に大規模な問題を解ける手法がいくつか提案されている。本章では、このような SVM に特化した手法を紹介する。ここで紹介する手法はすべて上に述べた解のスパース性を利用している。

3.1 Chunking

ほとんどの問題においては、最適な α_i は 0 または C である。もしもどの α_i が 0 になるかを知っていれば、それに対応する係数行列の行と列を、目的関数の値を変えることなく取り除くことができる。また、最適解が KKT 条件 14 を満たすという点も変化がない。文献 [17] で提案された Chunking という手法では、スパース性と、KKT 条件を用いて高速化を行う。Chunking においては、各ステップで、すべての非ゼロ α_i と、KKT 条件を満たさない数個の α_i からなる副問題が解かれる。各ステップにおいて、副問題のサイズは異なるが、最終的には真の非ゼロ α_i の数に一致する。この手法をもちいれば、ある程度の大きさの問題を扱うことができるが、非ゼロ α_i の割合が高い場合には、やはり大きな二次計画問題を解かなければならないという欠点がある。

3.2 Decomposition Method

Decomposition Method は、二次計画問題を繰り返し解く点は Chunking に類似しているが、各副問題のサイズが固定されているところが異なる。この手法は、各二次計画問題が、一つでも KKT 条件を満たさない α_i を含んでいれば、結局最適解に達することができるという性質に立脚している。ここでは、副問題のサイズを固定して、各ステップにおいて、α_i を一つづつ入れ替えていく方法がとられる。しかし、実際には、この手法の収束は非常に遅い。実用的には、入れかえる変数を選ぶための良いヒューリスティックと、高速なキャッシングの方法が必要になる。SVM light[5] では、このような改良を施して、実際に Decomposition Method を使っており、数千の非ゼロ α_i を含む問題を高速に解くことができる。

3.3 SMO

Sequential Minimum Optimization[10] は、Decompostion Method の最も極端な場合として捉えることができる。各ステップでは、2 変数の二次計画問題が解かれる。この最適化は、解析的に行うことができるので、二次計画問題を解くパッケージは必要ない。SMO では、最も重要な問題は、この 2 変数をどのように選ぶかという点にある。文献 [10] で示されたヒューリスティックは、KKT 条件に基づくシンプルなものであるが、その改良版も提案されている [6]。提案されたときには分類問題だけを対象としていたが、回帰問題に適用できる手法も提案されている [14]。

4. ℓ_1 正則化項によるスパース解

従来から、正則化項として、ℓ_1 ノルム、すなわち、

$$\sum_{i=1}^{n} |\alpha_i| \tag{15}$$

を用いることによって、スパースな解を得られることが知られている [1]。従って、SVM の正則化項 (式 11) を、ℓ_1 ノルムと入れ替えれば、よりスパースな解を得ることが期待される。このような手法は、Linear Programming Machine と呼ばれる [3]. この最適化問題は、$\boldsymbol{\alpha}$ を

$$\alpha_i = \alpha_i^+ - \alpha_i^-, \ \alpha_i^+ \geq 0, \alpha_i^- \geq 0 \tag{16}$$

のように分割することにより、線形計画法で解くことができる。

$$\begin{aligned} \max_{\boldsymbol{\alpha}^+,\boldsymbol{\alpha}^-} \quad & \sum_{i=1}^{n}(\alpha_i^+ + \alpha_i^-) + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i f(x_i) \geq 1-\xi \qquad (17) \\ & \alpha_i^+, \alpha_i^-, \xi_i \geq 0 \qquad (18) \end{aligned}$$

この方法により、識別性能を大きく下げずに、解のスパース性を大幅に向上させることができる。

5. 線形判別分析をスパースに

前章のようにℓ_1ノルムを正則化項に用いれば、様々な手法にスパース性を付加することができる。本章では、長い歴史を持つ線形識別器である線形判別分析にスパース性を付加する方法について述べる。ここでは、まず、線形判別分析法をカーネルトリックを用いて、非線型に拡張し、その上でスパースにする。

まず、線形判別分析法について説明する。この手法では、線形識別関数$f(x)=w^Tx$によって、訓練サンプルを射影したとき、最も2クラスが分離されるようにwを決定する。分離度は、次のRayleigh Coefficientによって定義される。

$$J(w) = \frac{w^T S_B w}{w^T S_W w}, \qquad (19)$$

ここで、S_B,S_Wは、それぞれクラス間、クラス内分散を表し、m_kをクラスkのサンプル平均、I_kをクラスkのインデックス集合とすると、次のように表される。

$$S_B = (m_2-m_1)(m_2-m_1)^T$$

$$S_W = \sum_{k=1,2}\sum_{i\in\mathcal{I}_k}(x_i-m_k)(x_i-m_k)^T.$$

そして、wは、この分離度を最大にするように決定される。

$$\min_w -J(w) \qquad (20)$$

カーネル特徴空間で線形判別分析を行うため、特徴空間での分離度を定義しよう。SVMと同様に、wを、特徴空間に写像された訓練サンプルの線形結合で表すと、

$$w = \sum_{i=1}^{n}\alpha_i\Phi(x_i). \qquad (21)$$

(19)の中のx_i¥を、$\Phi(x_i)$で置き換え、wには、(21)を代入すると、分離度は次のように書ける。

$$J(\boldsymbol{\alpha}) = \frac{(\boldsymbol{\alpha}^T\boldsymbol{\mu})^2}{\boldsymbol{\alpha}^T N\boldsymbol{\alpha}} = \frac{\boldsymbol{\alpha}^T M\boldsymbol{\alpha}}{\boldsymbol{\alpha}^T N\boldsymbol{\alpha}}, \qquad (22)$$

ここで、$\boldsymbol{\mu}_k = \frac{1}{|\mathcal{I}_k|}K1_k$, $N = KK^T - \sum_{k=1,2}|\mathcal{I}_k|\boldsymbol{\mu}_k\boldsymbol{\mu}_k^T$, $\boldsymbol{\mu} = \boldsymbol{\mu}_2 - \boldsymbol{\mu}_1$, $M = \boldsymbol{\mu}\boldsymbol{\mu}^T$, $K_{ij} = (\Phi(x_i)\cdot\Phi(x_j)) = k(x_i,x_j)$である。(20)の最適化問題にスパース性を付加するためにℓ_1ノルム正則化項を加えると、

$$\max_{\boldsymbol{\alpha}} -J(\boldsymbol{\alpha}) + \lambda\sum_{i=1}^{n}|\alpha_i| \qquad (23)$$

この最適化問題は、次のように書きかえられる[8]。

$$\begin{aligned} \min_{\boldsymbol{\alpha},b,\boldsymbol{\xi}} \quad & \|\boldsymbol{\xi}\|^2 + C\sum_{i=1}^{n}|\alpha_i| \qquad (24) \\ \text{subject to} \quad & K\boldsymbol{\alpha} + 1b = y + \boldsymbol{\xi} \\ & 1_k^T\boldsymbol{\xi} = 0 \text{ for } k = 1,2 \end{aligned}$$

ここで、$\boldsymbol{\xi}\in\Re^n$, $b,C\in\Re$である。$\boldsymbol{\xi}$,bは、補助的に用いられるスラック変数であり、Cはλに変わって正則化の度合いを制御するためのパラメータである。線形判別分析は、もともと全くスパースでない手法であるので、この改良によるスパース化の効果は絶大である。データセットによっては、パラメータ全体の97%が0になった事例も報告されている[8]。

他の手法に関しては、Kernel PCAのスパース化が報告されている[16]。同様にして、因子分析、クラスタリングの諸手法などもスパース化できると考えられる。

6. Bayes Point Machine

SVMの関連研究においては、大規模な問題に対応させるために、線形計画問題か、二次計画問題に帰着させるのが一般的であるが、理論的な立場から、より複雑な最適化を必要とするものもある。ここでは、そのなかから、Bayes Point Machine[11, 4]を紹介する。

Bayes Point Machineは、線形識別器であるので、識別関数は(1)で表される。但し、$\|w\|=1, b=0$とする。このとき、すべての訓練サンプルを正しく分類するwの集合は、

$$\mathcal{V} = \{w | y_i f(x_i) > 0; i = 1,\cdots,n, \|w\| = 1\}$$

と表される。この集合$\mathcal{V}$は、Version spaceと呼ばれ[11]、SVMの解は、Version spaceのTchebycheff中心($\mathcal{V}$に含まれる最大の球の中心)に対応することが知られている[11]。しかし、理論的に最適な点は、Bayes pointと呼ばれる点であり、これは、Version Spaceの重心によってよく近似される[9]。Version Spaceは、図2,3のように、半径1の球の一部分として表される。

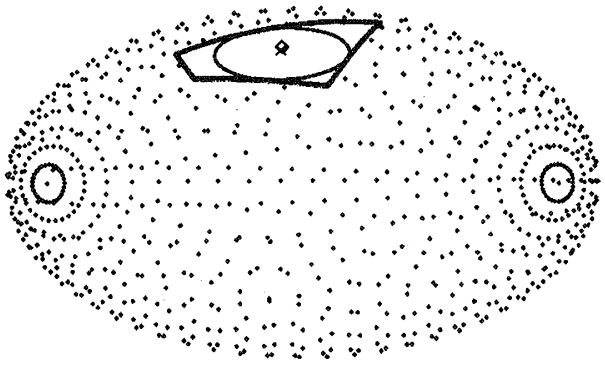

図2: SVMがうまく働くVersion Spaceの例。重心(◇)がSVMの解(×)に近い。

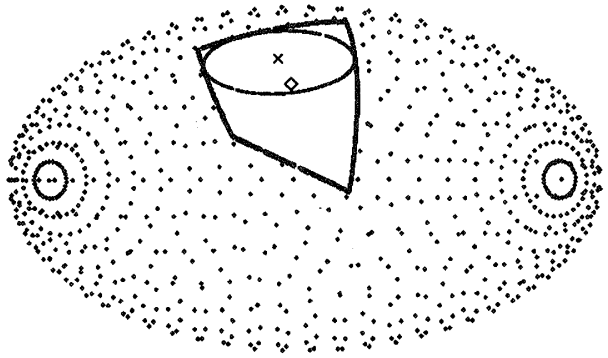

図3: SVMがうまく働かないVersion Spaceの例。Version Spaceが一方向に長く、重心(◇)は、SVMの解(×)から離れている。

もしも、Version Spaceが図2のような形をしていれば、SVMの解は、重心と近くなるが、図3のように、一方向に長い形をしていると、SVMの解は、重心から遠くなってしまう。Bayes Point Machineでは、この問題に対処するため、Billiard法[12]を用いてVersion Spaceの中心を近似的に求める。この方法は、いくつかのベンチマークでSVMを上回る結果を出している[11]。

7. おわりに

本稿で紹介した多くの手法では、数理計画法が学習のために用いられている。特に、線形計画問題や、凸二次計画問題のように、確実に大局的最適解を求められる手法が好まれている。これは、かつてニューラルネットワークにおいて、目的関数が複雑になりすぎ、深刻な局所解の問題を生じたことの反省からである。数理計画法は、勾配降下法に比べて、理論的に明確であり、また、実用的にも高速である。KKT条件など、数理計画法特有の道具を用いて、学習の問題に取り組むこともできる。

最適化理論から学習を扱う研究は、未だ未開拓であるので、さまざまな研究課題が考えられる。例えば、正則化パラメータの値とスパース性の関係など、未だよく分かっていない問題もあるので、最適化の専門家の機械学習分野への貢献は重要であろうと思われる。

参考文献

[1] K.P. Bennett and O.L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–34, 1992.

[2] D.P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, MA, 1995.

[3] T. Graepel, R. Herbrich, B. Schölkopf, A.J. Smola, P.L. Bartlett, K.-R. Müller, K. Obermayer, and R.C. Williamson. Classification on proximity data with LP-machines. In D. Willshaw and A. Murray, editors, *Proceedings of ICANN'99*, volume 1, pages 304–309. IEE Press, 1999.

[4] R. Herbrich and T. Graepel. Large scale Bayes point machines. In *Advances in Neural Information System Processing 13*, 2001. accepted for publication.

[5] T. Joachims. Making large–scale SVM learning practical. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, 1999. MIT Press.

[6] S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K.R.K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. Technical Report CD-99-14, National University of Singapore, 1999. `http://guppy.mpe.nus.edu.sg/~mpessk`.

[7] G.S. Kimeldorf and G.Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.*, 2:495–502, 1971.

[8] S. Mika, G. Rätsch, and K.-R. Müller. A mathematical programming approach to the Kernel Fisher algorithm. In *Advances in Neural Information Processing Systems 13*, 2001. to appear.

[9] M. Opper and D. Haussler. Generalization performance of Bayes optimal classification algorithm for learning a perceptron. *Physical Review Letters*, 66:2677, 1991.

[10] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999. MIT Press.

[11] T. Graepel R. Herbrich and C. Campbell. Bayes point machines: Estimating the bayes point in kernel space. In *Proceedings of IJCAI Workshop Support Vector Machines*, pages 23–27, 1999.

[12] P. Ruján. Playing billiard in version space. *Neural Computation*, 9:197–238, 1996.

[13] B. Schölkopf, K.-K. Sung, C.J.C. Burges, F. Girosi, P. Niyogi, and V.N. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing*, 45(11):2758–2765, 1997.

[14] A. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 2001. Forthcoming.

[15] A.J. Smola and P.L. Bartlett. Sparse greedy gaussian process regression. In *Advances in Neural Information Processing Systems 13*, 2001. to appear.

[16] M. Tipping. Sparse kernel principal component analysis. In *Advances in Neural Information Processing Systems 13*. MIT-Press, 2001. to appear.

[17] V.N. Vapnik. *Estimation of Dependences Based on Empirical Data.* Springer-Verlag, Berlin, 1982.

[18] V.N. Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

[19] G. Wahba. Support vector machines, reproducing kernel hilbert spaces, and randomized GACV. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 69–87, Cambridge, MA, 1999. MIT Press.

[20] R.C. Williamson, A.J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. NeuroCOLT Technical Report NC-TR-98-019, Royal Holloway College, University of London, UK, 1998. To appear in *IEEE Transactions on Information Theory.*

[21] 津田 宏治. サポートベクターマシンとは何か. 電子情報通信学会誌, 83(6):460–466, 2000.

[22] 赤穂 昭太郎 and 津田 宏治. サポートベクターマシン：基本的仕組みと最近の発展. 数理科学, 38(6):52–58, 2000.