H-025

サポートベクターマシンの特徴写像の学習について
On training of feature map in SVM

斉藤 弘紀 † Hironori Saito
趙 晋輝 † Jinhui Chao

1. はじめに

近年、学習マシンの中でサポートベクターマシン (SVM) は、汎化性能が優れることで注目されている。一方、次の問題点も考えられる。まず、予め設定された特徴写像または埋め込み関数によって入力信号を特徴空間に写したときは、一般的に線形分離できる確証はない。例えば常に outlier の存在する可能性がある。また、たとえ線形分離ができたとしても、一般的な場合は数多くの埋め込み関数、即ち非常に高次元な特徴空間が必要である。従って、計算量が莫大となりかねない。そして、従来使われている評価関数では、サポートベクターとサポートベクターでないものとの区別を推測する方法がないため、全ての入力データがサポートベクターとして使われている。そのため無駄な計算も多く含まれている。

本論文では、サポートベクターマシンの以上の問題点を解決するために、埋め込み関数または特徴写像の学習を用いた新しい方式について提案する。まず、埋め込み関数を効率的に学習するための評価関数を新たに定義する。そして、それに基づいた重み係数と埋め込み関数のパラメータの学習アルゴリズムを示す。本方式は、線形分離性を確保できると同時に埋め込み次元の低減も図れる。さらに、誤認識された学習データのみを用いて更新するため、計算効率の向上が可能となる。

2. SVMのLagrange乗数法による評価関数

$\boldsymbol{w}$ をサポートベクターマシンの重みベクトル、b を閾値、$\{x_i, d_i\}$ を訓練データの入力信号と教師信号とする。$\phi(\cdot)$ は、入力空間から特徴空間への特徴写像または、埋め込み関数とする。
入力空間から特徴空間への特徴写像を学習するために、まず Lagrange 乗数によるサポートベクターマシンの評価関数を使うことを試みる。ここで、$\phi = \phi_\theta$ は θ でパラメータ化されたものとする。

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} - \sum_{i=1}^{n}\alpha_i(d_i(\boldsymbol{w}^T\phi_\theta(x_i)+b)-1)$$

ここで、ϕ_θ と重みベクトル $\boldsymbol{w}$ を同時に学習させると、勾配ベクトルは

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{n}\alpha_i d_i \boldsymbol{\phi}(\boldsymbol{x}_i)$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = -\sum_{i=1}^{n}\alpha_i d_i \boldsymbol{w}\frac{\partial \boldsymbol{\phi}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}}$$

となるため、$\boldsymbol{\theta}$ の更新式は

$$\boldsymbol{\theta}(n+1) = \boldsymbol{\theta}(n) + \mu\sum_{i=1}^{n}\alpha_i d_i \boldsymbol{w}\frac{\partial \boldsymbol{\phi}(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}}$$

となる。しかしながら、ここで Lagrange 乗数と埋め込みパラメータ θ の値が未知数であるため勾配の値を決定することができない。

従来の SVM では双対問題を導入することによって未知数である Lagrange 乗数を回避できた。しかし、上記双対評価関数には、パラメータ $\boldsymbol{\theta}$ も同時に含まれているため、双対問題に発展しても問題の解決にならない。
同様に Lagrange 乗数が未知のためサポートベクター ($\alpha_i = 1$) とサポートベクターでない ($\alpha_i = 0$) 学習入力の区別がつかない。そのために、学習には実質上すべての入力をサポートベクターと見なさざるをえない。 次にこれらの問題を克服すべき、新しい評価関数について提案する。

3. 重みと特徴写像を学習するための評価関数

まず、入力空間から特徴空間への埋め込み関数を用いて、特徴写像を次のように定義する。

$$\boldsymbol{y} = (1, \boldsymbol{\phi}^T(\boldsymbol{x}))^T = (y_1, ..., y_M)^T,$$

ここで y_j のパラメータベクトルを $\boldsymbol{\theta}_j$ とする。重み係数 w は次のように定義する。

$$\boldsymbol{w} = (b, w_1, ..., w_M)^T = (w_0, w_1, ..., w_M)^T$$

また、ステップ関数を次のように定義する。

$$s(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}$$

ステップ関数 $s(1 - d_i\boldsymbol{w^T y_i})$ によって、定義される認識誤差を用いて重みベクトル $\boldsymbol{w}$ の値を更新できる。学習データ $\boldsymbol{x}_i, i = 1, .., N$ に対して、w と θ を学習するための評価関数としては以下の二つを用いる。

$$J_1(\boldsymbol{w}, \{\theta_j\}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \sum_{i, d_i\boldsymbol{w}^T\boldsymbol{y}_i<1}(1 - d_i\boldsymbol{w}^T\boldsymbol{y}_i)$$

ステップ関数 s が 0 でない場合、評価関数 J_1 は連続かつ滑らかである。実際に更新する計算はステップ関数が 0 でない場合のみ行なわれるため、勾配法が適用可能である。次に、評価関数 J_2 を以下のように定める。

$$J_2(\boldsymbol{w}, \{\theta_j\}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \sum_{i, d_i\boldsymbol{w}^T\boldsymbol{y}_i<1}(1 - d_i\boldsymbol{w}^T\boldsymbol{y}_i)^2$$

関数 J_2 は、連続かつ微分可能であり、同様に勾配法が適用可能である。J_1 に比べて J_2 は大きな誤りに対しては、大きなペナルティーを与えるが、小さな誤りに関しては敏感でないなどの特徴を有する。

以上の評価関数を用いることにより、従来 のサポートベクターマシンと比べて全てのデータが更新されるのではなく入力データ毎にデータをチェックして正しく判定されないデータに対してのみ重みを更新している。これにより、計算量が従来法に比べ削減、収束性能が改善されることが期待できる。

†中央大学大学院 理工学研究科 電気電子情報通信工学専攻

4. 学習アルゴリズム

以下では、前節で定義された二つの評価関数に対する学習アルゴリズムを導くために、まず w と $\{\boldsymbol{\theta}_j\}$ に関する勾配を示す。

$$\frac{\partial}{\partial \boldsymbol{w}} J_1(\boldsymbol{w},\{\boldsymbol{\theta}_j\}) = \boldsymbol{w}^T - \sum_{i, d_i \boldsymbol{w}^T \boldsymbol{y}_i<1} d_i \boldsymbol{y}_i^T$$

$$\frac{\partial}{\partial \boldsymbol{\theta}_j} J_1(\boldsymbol{w},\{\boldsymbol{\theta}_j\}) = \begin{cases} -d_i w_j \frac{\partial y_j(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_j} & (\text{if } d_i \boldsymbol{w}^T \boldsymbol{y}_i < 1) \\ 0 & (\text{if } d_i \boldsymbol{w}^T \boldsymbol{y}_i \geq 1) \end{cases}$$

$$\frac{\partial}{\partial \boldsymbol{w}} J_2(\boldsymbol{w},\{\boldsymbol{\theta}_j\}) = \boldsymbol{w}^T + \sum_{i, d_i \boldsymbol{w}^T \boldsymbol{y}_i<1} -2d_i(1 - d_i \boldsymbol{w}^T \boldsymbol{y}_i)\boldsymbol{y}_i^T$$

$$\frac{\partial}{\partial \boldsymbol{\theta}_j} J_2 = \begin{cases} -d_i w_j (1 - d_i \boldsymbol{w}^T \boldsymbol{y}_i) \frac{\partial y_j(\boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_j} & (\text{if } d_i \boldsymbol{w}^T \boldsymbol{y}_i < 1) \\ 0 & (\text{if } d_i \boldsymbol{w}^T \boldsymbol{y}_i \geq 1) \end{cases}$$

2 つの評価関数に対して最大勾配法により、$\boldsymbol{w}$ と $\boldsymbol{\theta}_j$ を学習させることができる。

5. RBF SVM への応用

一般的な RBF ネットワークの出力を次に定義する。

$$y_j(\boldsymbol{x}) = \phi_j(\boldsymbol{x}) = \exp\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{a}_j)^T \Sigma_j (\boldsymbol{x} - \boldsymbol{a}_j)\}$$

ここで、埋め込み関数の学習は、次の勾配を用いる更新によって行われる。

$$\frac{\partial y_j}{\partial \boldsymbol{a}_j} = y_j(\boldsymbol{x} - \boldsymbol{a}_j)^T \Sigma_j$$

今、 $\Sigma = (\sigma_{lk}), \boldsymbol{x}_i = (x_{il})^T, \boldsymbol{a}_j = (a_{jk})$ と定義すると、

$$\frac{\partial y_j(\boldsymbol{x}_i)}{\partial \Sigma} = -\frac{y_j(\boldsymbol{x}_i)}{2}(\boldsymbol{x}_i - \boldsymbol{a}_j)^T(\boldsymbol{x}_i - \boldsymbol{a}_j)^T$$

6. シミュレーション結果

評価関数 J_1 によって特徴写像を学習した SVM と従来型の SVM を比較した。

学習データは、UCI machine learning benchmark repository から Australian credit card problem と Heart disease problem を使用。

従来型の SVM は RBF ネットワークを用いた。提案法は同じ RBF ネットワークを使用し各データに対して 5 回学習した。μ は全て 0.01 とする。シュミレーション条件と認識率は、表 1 の通りである。

また、学習データの分離超平面に対する分布を図 1 と図 2 に示している。(横軸は $W^T y_i$)

本手法によって汎化性能と線形分離性の改善が確認できた。

7. まとめ

小文では、最適な埋め込み関数を学習データにより定めるための、新しい評価関数を定義し、学習方式を示した。本方式は線形分離を確保すると同時にマージンを学習によって大きくすることができるため、認識と汎化性能の向上が考えられる。また、特徴空間の次元を抑えることと、誤判定された入力のみ更新することで、計算量の低減が期待できる。今後、RBF の他、多層ネットワークおよび pyramid network などへの応用も考えられる。

	heart disease		card problem	
training data	243		621	
test data	27		69	
input	13		14	
—	J_1	SVM	J_1	SVM
Mean	0.7813	0.7148	0.8334	0.7709
Max	0.9259	0.8519	0.8841	0.8551
Min	0.6223	0.5556	0.7826	0.6667
SD	0.0893	0.1019	0.0315	0.0519

表 1: J_1 と従来型 SVM の汎化性能の比較

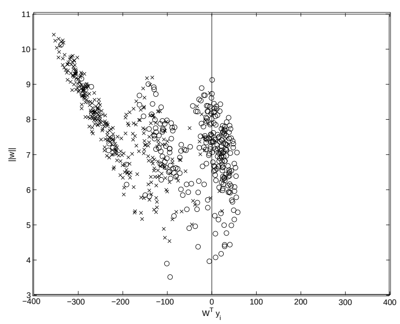

図 1: 従来型 SVM の判別関数と認識率の分布

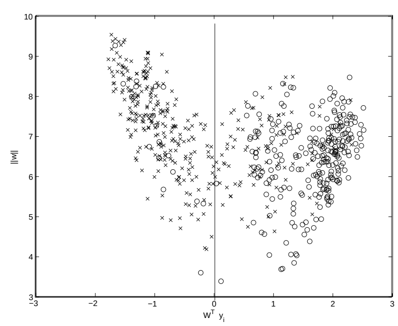

図 2: J_1 を用いた提案法の学習結果

参考文献

[1] S.Haykin,"Neaural networks,a comprehensive foundation",2nd edition,Prentice Hall,1999

[2] R. Herbrich, "Learning kernel classifier" The MIT Press, 2001.

[3] B. Scholkopf and A. J. Smola, "Learning with kernels-support vector machines, regulation, optimization, and behyond", The MIT Press, 2002.

[4] H.Saito , J.Chao , M.Hoshino Proceedings of the 2003 IEICE General Conference,D-12-42 IEICE,Japan,March 2003