H-007

SVMにおける特徴写像の学習に関する考察
On training of feature map in SVM

和田 健 † Ken Wada　斎藤 弘紀 † Hironori Saitou　塚原 裕史 ‡ Hiroshi Tsukahara　趙 晋輝 § Jinhui Chao

1. まえがき

現在知られている中で，サポートベクターマシン (SVM) は，パターン認識などの分野において最も優秀な学習モデルの一つである．その SVM の特徴としては，カーネルトリックとマージン最大化があげられる．動作としては，カーネルトリックにより，直接の計算は避けながらも入力を高次元の特徴空間に写像することにより，線形分離した後にマージンを最大化することによって汎化性能を確保するものである．

また，汎化性能を評価する指標として VC 次元が用いられるが，この VC 次元はマージンだけでなく特徴空間の入力の分布にも影響を受けるものである．しかし，従来の SVM では，特徴空間でのマージンは最大化しているものの入力の特徴空間への写像先の入力分布に関しては明らかではない．

本論文では，SVM の汎化性能を向上させるために，カーネルトリックを用いずに，一般的な学習ネットワークを特徴写像として用いて、VC 次元を減少させるような特徴写像を学習により求めるという方式を提案する．本文では，特徴写像として RBF ネットワーク，MLP を用いているが，M ピラミッド型ネットワークを用いることも考えられる．

2. 従来の評価関数での表現

$\boldsymbol{w}$ をサポートベクターマシンの重みベクトル，b を閾値，$\{\boldsymbol{x}_i, d_i\}$ 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 乗数を導入した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，$\phi = \phi_\theta$ であり，θ は学習されるべき特徴写像のパラメータとする．

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

となる．しかしながら，ここで重みベクトルと Lagrange 乗数，特徴写像のパラメータの値が未知数であるため，双対問題に帰着しても，この評価関数を解くことはできない．そこで，次に新しい評価関数について提案する．

3. 新しい評価関数の定式化

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．

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

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 $\boldsymbol{w}$ 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 $s(1 - d_i\boldsymbol{w}^T\boldsymbol{y}_i)$ 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，$\boldsymbol{x}$ と $\boldsymbol{\theta}$ 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．

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

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。

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

評価関数 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．

また，各評価関数 J_1, J_2 において，C は第一項と第二項の間のバランスを調整する regularization パラメータであり，この値がある程度大きい程，この評価関数の解は認識誤差よりも識別関数の複雑さが強調されて，ネットワークの VC 次元を下げることとなり汎化の向上が期待できる．しかし，最適な C の値の導出法は，今後の課題の一つとなっている．

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 $\boldsymbol{w}$ と $\{\boldsymbol{\theta}_j\}$ に関する勾配を示す．

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

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

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

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

各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。

†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
‡中央大学大学院 理工学研究科 物理学専攻
§中央大学大学院 理工学研究科 情報工学専攻

5. RBF を特徴写像として用いた SVM

特徴写像として用いる RBF ネットワークの出力を次に定義する．

$$\boldsymbol{y}_j(\boldsymbol{x}) = \phi_j(\boldsymbol{x}) = \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}_j)^T G_j(\boldsymbol{x}-\boldsymbol{c}_j)\}$$

ここでは，センターを $\boldsymbol{c}_j = \boldsymbol{x}_j + \boldsymbol{a}_j$ とする．$\boldsymbol{a}_j$ を加えることにより学習によりセンターを求めることが可能となる．これにより，トレーニングデータの写像先をより適正に学習させることができ，VC 次元を低くする効果が期待される．

5.1 RBF を特徴写像としたときの J_1 の学習アルゴリズム

RBF のセンターと分散を勾配法により学習する際の式は以下の様に表すことができる．

$$\begin{aligned}
\frac{\partial J_1}{\partial \boldsymbol{w}} &= \boldsymbol{w} - C \sum_{i, d_i \boldsymbol{w}^T \boldsymbol{y}_i < 1} d_i \boldsymbol{y}_i \\
\frac{\partial J_1}{\partial \boldsymbol{a}_j} &= C \sum_{d_i \boldsymbol{w}^T \boldsymbol{y}_i < 1} d_i \boldsymbol{w}_j \frac{\partial \boldsymbol{y}_j(\boldsymbol{x}_i)}{\partial \boldsymbol{a}_j} \\
&= C \sum_{d_i \boldsymbol{w}^T \boldsymbol{y}_i < 1} d_i \boldsymbol{w}_j \boldsymbol{y}_j(\boldsymbol{x}_i) \cdot (\boldsymbol{x}_i - \boldsymbol{c}_j)^T G_j \\
\frac{\partial J_1}{\partial g^j_{lk}} &= C \sum_{d_i \boldsymbol{w}^T \boldsymbol{y}_i < 1} d_i \boldsymbol{w}_j \frac{\partial \boldsymbol{y}_j(\boldsymbol{x}_i)}{\partial g^j_{lk}} \\
&= \frac{C}{2} \sum_{d_i \boldsymbol{w}^T \boldsymbol{y}_i < 1} d_i \boldsymbol{w}_j \boldsymbol{y}_j(\boldsymbol{x}_i) \cdot (\boldsymbol{x}_l - \boldsymbol{c}^j_l)(\boldsymbol{x}_k - \boldsymbol{c}^j_k)
\end{aligned}$$

6. RBF ネットワークに対する入力の前処理

通常，RBF ネットワークでは、入力数とニューロン数が等しくなる．ここで，入力数が多くなるとニューロン数，つまり特徴空間がより高次元になるため汎化性能の低下を招くことにつながる．また，ネットワークの複雑さは入力数ではなく判別すべき問題の複雑さによって決められるべきはずである．

以上のことから，汎化性能の向上のために入力を RBF ネットワークに通す前に自己組織化写像 (SOM) を用いて，クラスタ化することにより，入力分布の特徴を抽出して入力数を減少，つまり RBF ネットワークのニューロン数を減少させることを考える．また，ニューロン数を減少させることにより，ネットワークをハード化した際の規模を縮小させることにもつながる．

6.1 前処理のアルゴリズム

入力 $\boldsymbol{x}_k, k = 1, ..., N$ を各クラス毎に分け，2 次元配置された新たな重み $\boldsymbol{W}_{ij}$ を設定し，各 $\boldsymbol{x}_k$ と最もユークリッド距離の近い $\boldsymbol{W}_{ij}$ とその近傍の重みに対して以下の式で充分回更新する．このとき，μ と重みの近傍は徐々に減少させていく．

$$\begin{aligned}
\boldsymbol{W}_{ij}(n+1) &= \boldsymbol{W}_{ij}(n) + \eta \|\boldsymbol{x}_k - \boldsymbol{W}_{ij}\| \\
\eta &= \begin{cases} +\mu & \text{if} \|\boldsymbol{x}_k - \boldsymbol{W}_{ij}\| > \delta \\ -\mu & \text{if} \|\boldsymbol{x}_k - \boldsymbol{W}_{ij}\| < \delta \end{cases}
\end{aligned}$$

学習終了後，$\|\boldsymbol{x}_k - \boldsymbol{W}_{ij}\| < \delta$ なる全 $\boldsymbol{x}_k$ を $\boldsymbol{W}_{ij}$ を代表値として置き換える．

7. MLP を特徴写像として用いた SVM

L 段の隠れ層を持つ多層ネットワークにおいて，l 段目に J_l のユニットを持つとする．ここで，入力層については $l = 0$ と定義する．はじめに、l 番目の隠れ層の j 番目のユニットについて考える．但し ($1 \leq l \leq L$)
このユニットは重み係数ベクトル $\boldsymbol{\theta}^l_j$ と，活性化関数 f^l_j を持つ．各ユニットでは，前の隠れ層の出力を入力として受け取る．ここで，出力ベクトルは

$$\boldsymbol{x}^l_j = f^l_j(\boldsymbol{x}^{l-1}, \boldsymbol{\theta}^l_j)$$

とする．また，l 段目の出力ベクトルは

$$\boldsymbol{x}^l = (x^l_1, \ldots, x^l_{J_l})$$

と定義する．但し l 段目の隠れ層ユニットを J_l とする．また，ネットワークの入力ベクトルは $\boldsymbol{x}^0 = \boldsymbol{x}$ である．

7.1 MLP を特徴写像としたとき J_1 の学習則

MLP-SVM を勾配法を用いることにより学習する．ここで，i 番目の入力ベクトルを $\boldsymbol{x}^{(i)}, i = 1, \ldots, m$ と定義する．

その入力に対する MLP の L 段出力ベクトルは，$\boldsymbol{x}^L_{(i)} := \boldsymbol{x}^L|_{\boldsymbol{x}^0 = \boldsymbol{x}^{(i)}}$ と記する．

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{w}} J_1(\boldsymbol{w}, \Theta) &= \frac{\partial}{\partial \boldsymbol{w}} \left(\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \sum_{i,\ d_i \boldsymbol{w}^T \boldsymbol{x}^L_{(i)} < 1} 1 - d_i \boldsymbol{w}^T \boldsymbol{x}^L_{(i)} \right) \\
&= \boldsymbol{w}^T - \sum_{i,\ d_i \boldsymbol{w}^T \boldsymbol{x}^L_{(i)} < 1} d_i (\boldsymbol{x}^L_{(i)})^T
\end{aligned}$$

また，MLP に l 段 j 番ニューロンの重みベクトルを $\boldsymbol{\theta}^l_j$ とすると，

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\theta}^l_j} J_1(\boldsymbol{w}, \Theta) &= \frac{\partial}{\partial \boldsymbol{\theta}^l_j} \left(\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \sum_{i,\ d_i \boldsymbol{w}^T \boldsymbol{x}^L_{(i)} < 1} 1 - d_i \boldsymbol{w}^T \boldsymbol{x}^L_{(i)} \right) \\
&= - \sum_{i,\ d_i \boldsymbol{w}^T \boldsymbol{x}^L_{(i)} < 1} d_i \boldsymbol{w}^T \frac{\partial \boldsymbol{x}^L_{(i)}}{\partial \boldsymbol{\theta}^l_j}
\end{aligned}$$

ここでは，ヤコビアン行列 $\frac{\partial \boldsymbol{x}^L_{(i)}}{\partial \boldsymbol{\theta}^l_j}$

$$\frac{\partial \boldsymbol{x}^L_{(i)}}{\partial \boldsymbol{\theta}^l_j} = \begin{pmatrix} \frac{\partial x^L_1}{\partial \boldsymbol{\theta}^l_j} \\ \vdots \\ \frac{\partial x^L_{J_L}}{\partial \boldsymbol{\theta}^l_j} \end{pmatrix}_{\boldsymbol{x}^0 = \boldsymbol{x}^{(i)}}$$

は，通常の BP 法によって計算することができる．ここでは，ヤコビアンは，行ベクトルとして定義している．

参考文献

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

[2] H.Saito , J.Chao , M.Hoshino “On training of embedding functions in SVM”,Proceedings of the 2003 IEICE General Conference,D-12-42 IEICE,Japan,March 2003