論　文

オンラインプロトタイプ生成による大規模データに対する高速SVM構築法

笠井　航[†a)]　戸部雄太郎[†]　申　富饒[††]　長谷川　修[†††b)]

A Fast Prototype-Based SVM Learning for Large Data Sets

Wataru KASAI[†a)], Yutaro TOBE[†], Furao SHEN[††], and Osamu HASEGAWA[†††b)]

あらまし　本論文では，自己組織化マップに基づくサポートベクターマシン (SVM) の学習高速化に有効なプロトタイプ生成法について提案する．提案手法は SVM の学習に必要なプロトタイプ構成をオンラインかつ自律的に取得するものである．生成されたプロトタイプは分布間の距離を導入した SVM によって高速に学習される．複数のデータセットに対して比較実験を行い処理速度についての有効性が確認できた．

キーワード　SVM，自己組織化マップ，高速化，プロトタイプ

1. まえがき

サポートベクターマシン (SVM) [1] は文字認識や顔認識，スパムメールの仕分けなど機械学習における数多くの問題に利用されている．SVM はカーネルトリックによる高次元空間へのデータのマッピングを用い，非線形な識別境界を形成することを可能としており，高性能な識別器として着目されている．しかし SVM の学習には二次計画問題を解く必要があり，必要な計算量が膨大となるため大規模データの学習には向かないという問題がある．高速化を試みた既存手法として，Dong らはブロック対角行列を用いた部分問題への分割を提案している [2]．また Collobert らは SVM Mixture を形成し，並列にこれを解く方法を提唱している [3]．更に更新対象とする訓練サンプル（アクティブワーキングセット）を 2 個に限定し，繰返し選出と重み更新を行う SMO (Sequential Minimal Optimization) 法 [4] もよく知られており，LibSVM [5] 等の実装に用いられている．このほかワーキングセットの選出方法を改良し，MEB (Minimal Enclosing Ball) の概念を利用して高速化を図ったものに Core Vector Machine [6] がある．線形 SVM については切除平面法を利用する最適化計算が提案されている [8], [9]．ただし，双対問題についての導出が課題で現段階ではカーネルを適用できない．

本論文では学習データを効率的に圧縮し，SVM の学習高速化に寄与する手法を提案する．提案手法は，入力データの分布をとらえた学習に有意な少数のプロトタイプを自己組織化マップ (SOM) の概念を利用し，近似的に生成するものである．プロトタイプは分布の情報を保持し，SVM のカーネルには分布間の距離計算を用いたカーネルが導入される．

学習データの圧縮に関する関連研究が Wilson と Martinez [10] に詳しく論じられている．ランダムにサブセットを構築して解く RSVM [11] があるがオリジナルデータのサブセットを構築する限りにおいて外れ値が混入してしまうという問題を解決していない．提案手法に近い方法として，Shih らは RocchioScore を用いてデータの並べ換えを行った後に圧縮を図る手法 [12] をテキストマイニングに用いているが，オンラインなデータ処理への応用が難しい．ClusterSVM [13]

† 東京工業大学大学院総合理工学研究科知能システム科学専攻，横浜市
Department of Computational Intelligence and Systems Science, Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, Yokohama-shi, 226–8502 Japan
†† 中国南京大学計算機ソフトウェア新技術国家重点実験室，中国
The State Key Laboratory for Novel Software Technology, Nanjing University, China
††† 東京工業大学大学院理工学研究科像情報工学研究施設，横浜市
Imaging Science and Engineering Laboratory, Tokyo Institute of Technology, Yokohama-shi, 226–8503 Japan
a) E-mail: kasai.w.aa@m.titech.ac.jp
b) E-mail: hasegawa.o.aa@m.titech.ac.jp

はサポートベクターにならない点のみをクラスタリングによって圧縮しており，厳密解に近い結果が得られる利点があるが，ノイズ耐性を考慮しておらず，提案手法ほど速度向上への寄与が大きいものではない．またこれらの手法はデータの分散情報を考慮したものではない．

提案手法は既存のデータからノイズ耐性をもって，学習に適切な情報のみを取得するものである．また自己組織化マップのオンライン処理性能をもっているので，データの追加や適応のための SVM の再学習，大規模データのストリーム処理においても利点がある．

次章ではプロトタイプの定義と学習の全体像を説明し，**3.** でプロトタイプの実際の生成方法について詳しく述べる．**4.** ではいくつかの大規模データセットに対して行った比較実験を元に提案手法の性能について検討し，**5.** において結論を述べる．

2. プロトタイプ SVM

SVM はクラス間のマージンを最大化する超平面を求める問題で，双対問題まで導けば次の最適化計算に帰着される．

$$\text{maximize}: \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\alpha^T Q \alpha \tag{1}$$

$$\text{subject to}: \quad 0 \le \alpha_i \le C, i = 1, \ldots, N$$

$$\sum_{i=1}^{N} y_i \alpha_i = 0 \tag{2}$$

ここで α_i は学習データ $(x_i, y_i) \in (R^d, \{-1, 1\})$ に対応するラグランジュ乗数，C はソフトマージンの影響を既定する定数，Q は $Q_{ij} = y_i y_j K(x_i, x_j)$ を要素とする行列，ただし $K(x_i, x_j) = \Phi^T(x_i)\Phi(x_j)$ である．SVM が非線形の識別境界を効率的に求めることができるのは，高次元への写像 $\Phi(x)$ を直接用いずに，内積計算できる仕組みをもっているためで，そのような性質をもつ関数 K はカーネル関数と呼ばれる．

データ数が大きい場合に，最適化計算を解くのは式 (1) の行列 Q のコレスキー分解を含むため計算時間が膨大となってしまう．そこで，行列 Q のサイズを小さくすることを目的として，多数の近隣にあるトレーニングデータ点をまとめて表現するプロトタイプ θ を導入する．θ は表現の一般性を保持し，かつ計算上扱いやすい形として単一の正規分布とおくことにする．つまり一つのプロトタイプは次の情報セットで表現される．

$$\theta_i = (\mu_i, \Sigma_i, y_i) \tag{3}$$

ただし，μ_i，Σ_i はそれぞれ分布の平均と分散で y_i はプロトタイプのクラスラベルである．プロトタイプによって代表されるデータの集合 X_i としたとき，各点が生起する確率は

$$p(x \in X_i | \theta_i) = \frac{e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)}}{(2\pi)^{\frac{d}{2}} |\Sigma_i|^{\frac{1}{2}}} \tag{4}$$

となる．提案手法ではプロトタイプ間の距離に基づいて識別境界を学習することを考えて，次の Symmetric Kullback-Leibler 距離を採用することにする．

$$\begin{aligned} &D(p(x|\theta_i)||p(x|\theta_j)) \\ &= \int_{R^d} p(x|\theta_i) \log \frac{p(x|\theta_i)}{p(x|\theta_j)} dx \\ &\quad + \int_{R^d} p(x|\theta_j) \log \frac{p(x|\theta_j)}{p(x|\theta_i)} dx \end{aligned} \tag{5}$$

この計算は単一の正規分布のときに解くことができて次の式となる．

$$\begin{aligned} &tr(\Sigma_i \Sigma_j^{-1}) + tr(\Sigma_j \Sigma_i^{-1}) - 2d \\ &\quad + tr((\Sigma_i^{-1} + \Sigma_j^{-1})(\mu_i - \mu_j)(\mu_i - \mu_j)^T) \end{aligned} \tag{6}$$

ここで d は特徴の次元数である．更に非線形分離に対応するために次の KL カーネルを導入することができる [14]．

$$K(p_i, p_j) = e^{-aD(p_i||p_j)+b} \tag{7}$$

識別時には入力データについて，一定の分散 σI をもつデータと扱えばプロトタイプと同様の基準で比較が可能である．

本研究では Platt によって提案された Sequential Minimal Optimization (SMO) 法 [4] の枠組みをプロトタイプマージン最大化の最適化計算に用いる．この手法は二つのラグランジュ乗数よりなる部分問題を繰り返し解くもので，まず線形拘束条件を保持するために二つ以上の更新する乗数が選択される．更新式は次式で表されるが，制約条件から求まるしきい値を超えた場合には調整が入る．

$$\alpha_i^{\text{new}} = \alpha_i^{\text{old}} - \frac{y_i(E_i - E_j)}{\eta} \tag{8}$$

$$\alpha_j^{\text{new}} = \alpha_j^{\text{old}} - y_i y_j (\alpha_i^{\text{new}} - \alpha_i^{\text{old}}) \tag{9}$$

ここで，

$$\eta = K(p_i, p_j) + K(p_i, p_i) - 2K(p_j, p_j) \quad (10)$$

$$E_k = sign\left(\sum_{l=1}^{N} y_l \alpha_l K(p_l, p_k) - h\right) - y_k \quad (11)$$

h はニューラルネットワークモデルにおけるしきい値であり，合わせて逐次的に更新が行われる．更新方法は参考論文 [4] の **12.2.3** において詳細を確認することができる．SMO アルゴリズムではステップごとに全データに対して E の再計算が必要であるが，オリジナルの解法に比べて十分に高速な手法であるといえる．Dong らの手法 [2] や Core Vector Machine [6] も SMO 法を基本としてそれを拡張させた手法であり，これら他手法との比較は後章で示す．

次章では，提案手法における具体的なプロトタイプの生成法について述べる．

3. プロトタイプ生成

データ全体を正規分布の混合で推定するためには最ゆう推定を用いるのが一般的である．しかし，提案手法のプロトタイプはそれ自体が近隣データのみから推定される確率密度であって，混合のような重なりと係数を含めたものではない．また解析に混合推定を用いるとしても推定の収束にかかる時間は高速化の目的を達成する上で好ましくない．そこで，提案手法では大規模データという状況を利用した近似的なプロトタイプ生成法を導入する．近似手法は自己組織化マップ (SOM) を利用するものである．SOM を利用した密度推定の関連研究は [15] 等がある．

提案手法は自律的なプロトタイプ数の獲得を実現するために，自己増殖型ニューラルネットワーク [16], [17] の枠組みから，プロトタイプの生成と不必要になったプロトタイプの削除を盛り込む．プロトタイプのパラメータ更新は分散と平均の近似値を逐次的に算出するよう設計する．以降プロトタイプの平均パラメータを SOM におけるノードの重みと置く．この妥当性については後述する．

3.1 生成アルゴリズム

自己増殖型ニューラルネットワークにおけるノード挿入の判断は既存のネットワーク中のノードに割り当てた類似しきい値 T に基づいて行われる．ノードの挿入の例を図 1 に示す．入力パターンとその最近傍にあるノード（勝者ノード）及び 2 番目に近いノード（第二勝者ノード）との距離がそれらのノードの類似しきい値 T を超える場合，入力パターンが新規プロトタイプとしてネットワークに挿入される．各ノードの類似しきい値 T_i の算出法は次のとおりである．

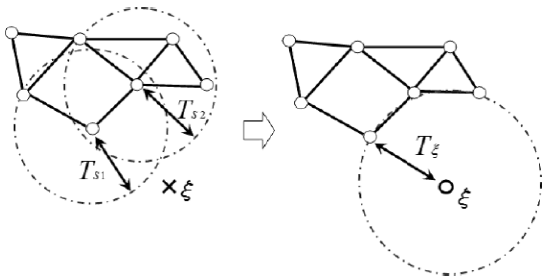

図 1 ノードの挿入処理：入力パターン ξ と勝者ノード s_1 及び第二勝者ノード s_2 との距離が類似しきい値 T_{s_1}，T_{s_2} より大きい場合（左），入力 ξ を新たなノードとして挿入する（右）．新ノードの類似しきい値 T_ξ は勝者ノードとの距離で表される．
Fig. 1 Node insertion process.

アルゴリズム *2.1*: 類似しきい値 T の計算

（1） ノード i の類似しきい値を，ノードの生成時に $+\infty$ に初期化する．

（2） ノード i が勝者ノードまたは第二勝者ノードである場合，T_i を更新する．

- ノード i に隣接ノードが存在する場合，T_i をノード i から最も遠い隣接ノードとの距離値に更新する．

$$T_i = \max_{c \in N_i} \|\mu_i - \mu_c\|, N_i = \{j | \exists (i,j) \in E\} \quad (12)$$

- ノード i に隣接ノードが存在しない場合，T_i をノード i から最も近い他のノードとの距離値に更新する．

$$T_i = \min_{c \in \Lambda \setminus \{i\}} \|\mu_i - \mu_c\| \quad (13)$$

ここで Λ は全ノードの集合，μ_i はノード i の重み，E はエッジ集合を示す．勝者ノードは入力パターンについての最近傍ノードを意味し，第二勝者ノードは入力パターンに 2 番目に近いノードを意味する．

ノード隣接関係の構成要素であるエッジについては，competitive Hebbian rule (CHL) [18] を採用する．CHL は入力パターンについてユークリッド距離で最も近い二つのノード（勝者ノードと第二勝者ノード）をエッジで連結する．構成されるネットワークは Delaunay グラフの部分グラフとなる．一方，ノードの重みの更新はネットワークの入力パターンへの局所的な適応を目的として行われる．

アルゴリズム *2.2*: 更新処理

（1） ノード集合 Λ を，学習データからランダムに選択した位置ベクトルをもつ二つのノード ($\Lambda = \{c_1, c_2\}$) に初期設定する．また，エッジ集合 E ($E \subset \Lambda \times \Lambda$) は空集合とする．

（2） $\xi \in R^n$ を入力パターンとする．

（3） 入力パターンに対する勝者ノード s_1 と第二勝者ノード s_2 を探索する．

$$s_1 = \underset{c \in \Lambda}{\operatorname{argmin}} \|\xi - \mu_c\| \tag{14}$$

$$s_2 = \underset{c \in \Lambda \setminus \{s_1\}}{\operatorname{argmin}} \|\xi - \mu_c\| \tag{15}$$

入力パターン ξ とノード（s_1 または s_2）との距離が類似しきい値（T_{s_1} または T_{s_2}）より大きい場合，入力パターンを新ノードとして Λ に追加する．その後，新しい入力パターンの学習のためにステップ（2）に戻る．類似しきい値 T はアルゴリズム *2.1* で算出される．

（4） s_1 と s_2 との間のエッジが存在しなければ，新たに作成して E に追加する．存在する場合は該当するエッジの年齢を 0 にリセットする．

（5） s_1 につながるすべてのエッジの年齢をインクリメントする．

（6） 勝者についてプロトタイプの平均（ノードの重み）と分散を，以下の式を用いて更新する．ただし，t を該当ノードが勝者ノードに選択された回数，$\mu(t)$, $\Sigma(t)$ を t ステップ目における平均，分散の値とする．

$$\Delta\mu = \frac{1}{t}(\xi - \mu(t-1)) \tag{16}$$

$$\Delta\Sigma = \frac{(\xi - \mu(t))^T(\xi - \mu(t)) - \Sigma(t-1)}{t} \tag{17}$$

（7） 事前定義したしきい値 a_d を超える年齢のエッジを削除する．その結果，隣接関係をもたないノードが現れた場合は，該当するノードを削除する．

（8） 入力パターン数が λ の倍数となった場合，隣接ノード数が 1 以下のノードを削除する．

（9） 新しい入力パターンについてステップ（2）に戻り繰返し計算を行う．

アルゴリズムにはエッジの加齢処理 (*edge aging* scheme [19]) が含まれる．これは学習初期段階で生成され，以降の入力で参照回数が少なくなったエッジを除去するための処理である．具体的には勝者ノードに連結するすべてのエッジを加齢し，その一方で勝者ノードと第二勝者ノードの間のエッジ年齢を 0 に更新する．その後，事前定義されたしきい値 a_d を超える年齢になったエッジは削除される．ノードの移動によって不適切となったエッジは隣接関係が成り立たないため，エッジの年齢が更新されずに削除されることになる．また Step(8) ではノイズの影響を受けたノードの隣接ノード数が 1 以下であるという仮定をおき，入力パターン数が λ の整数倍になった時点で全ノードの評価を行い，不必要なノードの削除を行っている．

3.2 近似アルゴリズム

先に述べたアルゴリズムにおけるノードの重み更新式は，漸化式を解けば平均の算出式である．つまり，大規模データであることを考慮して，各時刻におけるノード位置の揺らぎを無視できると仮定すれば k 平均法における重心，つまり近隣データの平均と等価である．

本研究では分散を対角共分散に限定する．これは次の二つの観点から都合が良い．先のアルゴリズムによって生成されるプロトタイプの分散は特に近隣データの数が特徴次元数を下回るほど少数の場合に，正則性が損なわれることがある．正則化処理を入れることもできるが，対角共分散に限定することでこの問題を回避しやすくなる．より重要な理由は，式 (6) の計算時間の短縮である．式 (6) の計算式は逆行列の算出を含んでいるため，このままでは計算時間が大きく膨らんでしまう．対角共分散を用いることによって逆数の計算のみで済むのは学習高速化に都合が良い．特に分散を σI とすれば，式 (7) はよく知られたガウスカーネルに収れんする．

$$K(x_i, x_j) = e^{-a\|x_i - x_j\| + b} \tag{18}$$

この性質は KL カーネル以外で正規分布の PPK (Probability Product Kernel) [20] にも見られる．

人工データを用いて提案手法をテストした結果を図 2 に示す．データはガウス分布，正弦波と同心円に従う分布及びノイズの混合から 50000 点をサンプリングしたものである．提案手法のプロトタイプ生成を既定するパラメータは $(\lambda, a_d) = (50, 50)$ とした．形成されたプロトタイプ数は 81 点で 0.16%という高圧縮率を実現できている．データは完全に逐次入力で処理されており，複雑な分布に対しても提案手法によってノイズ耐性と自律的なプロトタイプ生成が実現できていることが確認できる．提案手法では混合と異なりガウス分布が単一のプロトタイプではなく，一定間隔の近隣データで区切られた複数の分布の集合で表現され

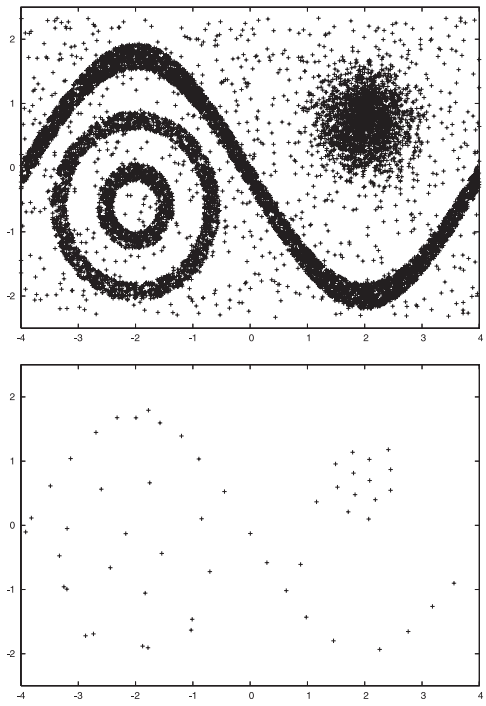

図 2　人工データ：50000 点（上），提案手法によって生成されたプロトタイプの平均位置：81 点（下）
Fig. 2　Artificial data processing.

ることに注意する．

4. 評価実験

実験は 6 種類の大規模データセットに対して行った．提案手法に必要なパラメータ設定は (λ, a_d) 及び，カーネルパラメータの a と b，ソフトマージンを既定する式 (2) の C であり，事前実験により適切な値に調整している．実験に用いた PC スペックは Intel Xeon 3 GHz，メモリ 3 GByte のマシンで OS は Windows XP である．原則としてすべての手法を同一マシンにおいてテストするが，後述の Dong らの手法のみ実装を用いた実験が行えないため参考値として論文の値を引用した．

4.1 Forest CoverType

一つ目のデータは UCI Repository にある Forest CoverType [21] で 30 m × 30 m の環境内における森林観測データである．実験では 7 クラス中データ数の多い Spruce-Fir と Lodgepole Pine の 2 クラスを使用した．データ数は 495141 点でバイナリーや実数値の混成よりなる 54 次元データで構成される．トレーニングデータとテストデータの割合を 9 : 1 とした 10 回の交差検定によって評価する．実験では提案手法と SMO 法を用いた実装である LibSVM [5]，収束性能に改善を加えた切除平面法の実装である OCAS（Vine Linux4.2 Octave インタフェースを使用），更にプロトタイプ数若しくは提案手法と同等の計算時間に合わせてランダムにトレーニングデータからサンプリングして LibSVM に入力した場合（各 Random1, Random2）について比較した．結果を表 1 に示す．提案手法の学習時間は前項がプロトタイプ生成時間で後項が SVM のトレーニング時間である．識別時間については 49514 点のテストサンプルの合計時間である．実験のパラメータは提案手法について $(\lambda, a_d, C, a, b) = (500, 50, 2^9, 2^{-5}, 0)$，LibSVM は RBF カーネル（ガウスカーネルの $b = 0$）を用いて $(C, a) = (1, \frac{1}{54})$，ランダムの場合には $(C, a) = (2^{15}, 2^{-15})$，OCAS については $(C, \epsilon) = (0.1, 10^{-5})$，ただし ϵ は収束を判定するパラメータでコストについては $C = 0.001, 0.1, 0.5, 1, 10$ と試行した結果，認識率と速度の点で最も有効であった値を採用した．

提案手法のデータサイズはプロトタイプ数を示し，オリジナルデータに対して 200 倍以上の圧縮が実現されている．SVM の計算時間は 1 秒以下でプロトタイプ生成時間を含めても 46 秒と高速である．ランダムサンプリングにおいては安定した識別性能が確認されず，提案手法が有意なプロトタイプの生成を行っていることが確認できた．提案手法では少数のサポートベクターで効率良く識別器が構築されている．サポートベクターの数が多いほど一般に識別処理には時間がかかるので，提案手法は識別についても高速化することができる手法であるといえる．OCAS は線形識別なのでサポートベクターの数は少ない．また，主問題を解くことで直接的に線形加重を算出しているので，識別計算の速度はサポートベクターの数に依存せず高速である．切除平面法は高速な最適化計算として将来的に有望であるが，現段階では非線形に対応していないのでスパース性を担保できない場合認識率の低い点が問題である．本手法でも SMO 法から切除平面法へ移行することが可能である．

サポートベクターの数について，汎化性能を Leave-one-out 誤差において評価するとき，学習された SVM (α_{emp}) の期待損失 R の上限はサポートベクターの数 k に依存する．

表 1　Forest CoverType を用いた比較：提案手法におけるデータサイズは生成されたプロトタイプ数．学習時間の左項はプロトタイプ生成，右項は SVM の最適化計算に要した時間．
Table 1　Result on Forest CoverType: “Data Size” means inputs for SVM (Number of prototypes in case of proposed). First term of training time is prototype generating time, and second term is optimization time.

	提案手法	LibSVM	OCAS	Random1	Random2
Data size	2100	445627	445627	2100	16800
Num of SV	900	221620	38	850	6427
認識率 (%)	78.01	78.58	72.65	60.99	58.65
Train-Time (s)	45 + 1	52494	71	1	49
Test-TIme (s)	9	3714	0.15	9	83

表 2　MNIST を用いた比較
Table 2　Result on MNIST.

	提案手法	Dong 2005	LibSVM
Data size	11080	60000	60000
Number of SV	1901	24470	9527
認識率 (%)	99.00	99.40	99.42
Train-Time (s)	129.5 + 29	(230)	1682
Test-Time (s)	51.7	—	303

表 3　CVM，BVM との比較に用いたデータ
Table 3　Datasets used for comparison to CVM, BVM.

	Training	Test	Dimention	Class
KDD Cup 1999	4898431	311029	127	2
WEB	49749	14951	300	2
IJCNN	49990	91701	22	2
USPS	266079	75383	676	2

$$E[R(\alpha_{emp})] \leq \frac{E[k_{m+1}]}{m+1} \tag{19}$$

実験結果は提案手法が他手法に比べて高い汎化能力を併せ持つ可能性を示唆している．

4.2 MNIST

二つ目の実験は手書き文字認識データセットである MNIST [22] について行った．トレーニング 60000 個，テスト 10000 個の 28×28 pixel で 0 から 9 の数字識別である．実験では Dong らの高速化手法 [2]（Pentium4 1.7 GHz マシンによる参考値）との比較を行うため，同様の特徴抽出 [23] による 576 次元からなるデータを用いる．参考のため LibSVM を用いた結果も算出した．提案手法におけるパラメータは $(\lambda, a_d, C, a, b) = (500, 6, 2^3, 2^{-1}, 0)$ と設定した．結果を表 2 に示す．提案手法が他手法と同等程度の識別率を保持しながら，計算速度を向上させていることが確認できる．また CoverType データセットほど顕著ではないが，プロトタイプ生成時で約 6 倍，最終的なサポートベクターの数で約 14 倍の圧縮が達成されている．

4.3 KDDCup1999，Web，IJCNN，USPS

提案手法と CVM (Core Vevtor Machine) [6]，更に MEB (Minimum Enclosing Ball) を EB (Enclosing Ball) に置き換えた BVM (Ball Vector Machine) [7] との比較を行った．実験は 4 種類のデータセット [24] に対して行い，各データセットの詳細は表 3 に示すとおりである．提案手法のパラメータはそれぞれ $(\lambda, a_d, C, a, b) = (50, 6, 2^2, 2^{-7}, 0)$，$(1200, 500, 1, 1, 0)$，$(6000, 400, 2^2, 1, 0)$，$(200, 50, 2^3, 2^{-7}, 0)$ と設定した．

結果を表 4 に示す．CVM と BVM ともにパラメータ ϵ の増減によって認識率と学習速度の間にトレードオフがある．詳しい性能は [7] に報告されている．ここでは $\epsilon = 10^{-5}$ として実験を行った．また，参考のため LibSVM に対して行った実験結果も併記した．

提案手法はプロトタイプ生成を含めた時間では BVM に劣っているが，プロトタイプ生成後の最適化計算では他手法を大きく上回っている．また，全体的に少ないサポートベクターで高い識別を実現できている．提案手法におけるプロトタイプ生成はオンラインに動作するもので，データ構成が時間変化する場合や，データの収集を継続するシステムにおいて再学習を行う場合に大きな利点がある．提案手法による再学習は WEB と USPS が 1 秒程度で KDD Cup 1999 に関しては 1 ミリ秒以下である．

図 3 は継続的にデータを収集するシステムを想定した実験で，IJCNN のトレーニングデータ 1000 点ごとに再学習を行った結果である．再学習の頻度が多くなるほど，提案手法の累積計算時間が相対的に少なく済むことが確認できる．認識率はデータの収集によって向上する傾向があるが，提案手法は CVM，BVM よりも安定するまでに多くのデータを収集する必要がある．これは生成するプロトタイプの数が学習初期段階

表 4　CVM，BVM との比較結果
Table 4　Comparison between proposed method and CVM, BVM.

		提案手法	CVM	BVM	LibSVM
KDD Cup 1999	Number of SV	13	53	219	1132
	認識率 (%)	92.60	92.34	91.95	92.48
	Train-Time (s)	142.9 + 0.001	1.27	2.09	25200
WEB	Number of SV	2975	9889	3540	4660
	認識率 (%)	99.19	99.07	99.04	99.32
	Train-Time (s)	116 + 1.7	220.17	23.17	214
IJCNN	Number of SV	2932	9316	5721	3154
	認識率 (%)	98.72	98.37	98.38	98.99
	Train-Time (s)	170.4 + 16	259.4	67.38	55
USPS	Number of SV	484	4094	1426	1598
	認識率 (%)	99.35	99.50	99.53	99.54
	Train-Time (s)	400 + 1.6	735.8	67.03	1432

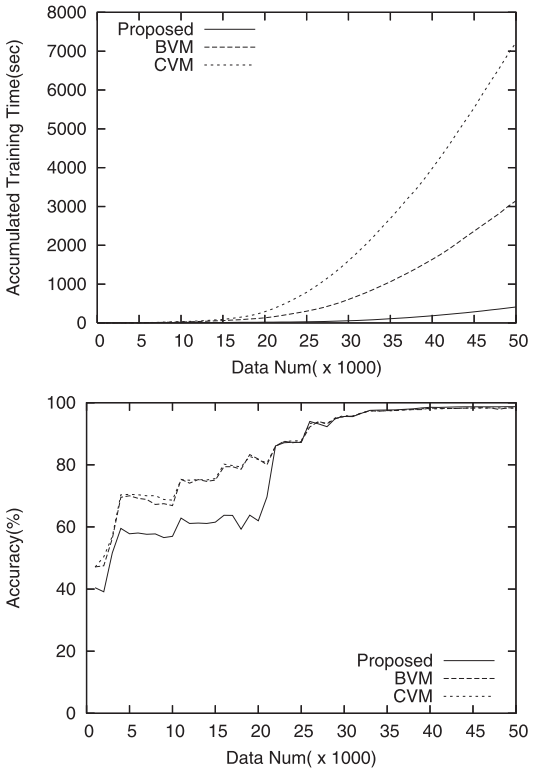

図 3　オンライン再学習の累積時間（上），オンライン再学習時の認識率（下）
Fig. 3　Online processing comparison.

で少なすぎることが問題で，収集したデータ数に応じたパラメータの自動決定が今後の課題である．

最後に，本手法はデータの読込みについてストリーム形式での処理が可能である．数十万規模のデータを直接メモリ上にロードするのは，次元数にもよるがメモリ占有量が数百 MByte 規模となり，読込み時間も学習の高速化を打ち消すほどに大きくなってしまう．比較したすべての手法が全データをメモリ上に読み込んだ上で学習が行われるのに対し，提案手法は生成されるプロトタイプの情報だけで，効率的に学習を遂行することが可能である．

5. むすび

本論文では，プロトタイプ生成に基づいた SVM の高速構築法について提案した．提案手法は SOM の概念を利用しており，オンラインに SVM の学習に有効な情報をもつプロトタイプを生成可能なものである．複数のデータベースに対して既存手法との比較実験を行い有効性を確認することができた．

我々はアルゴリズムをより洗練されたものにする改良を検討中である．提案手法はプロトタイプの生成消去がユーザパラメータに依存してしまう点が問題である．パラメータの自動決定が，本手法を展開する上で今後の重要な課題となる．構造化データ等に対して適切なカーネルとの親和性を考慮したプロトタイプ生成についても議論したいと考えている．高速化についてはボトルネックとなる勝者ノードの探索に近似最近傍探索等の適切な設計を行うことで更なる改良が可能であり，現在併せて研究中である．

謝辞　本研究は NEDO 産業技術研究助成事業から支援を頂きました．記して感謝致します．

文　献

[1] C. Cortes and V. Vapnik, “Support vector networks,” Mach. Learn., vol.20, no.3, pp.273–297, 1995.
[2] J.X. Dong, A. Krzyzak, and C.Y. Suen, “Fast SVM training algorithm with decomposition on very large datasets,” IEEE Trans. Pattern Anal. Mach. Intell., vol.27, no.4, pp.603–618, 2005.
[3] R. Collobert, S. Bengio, and Y. Bengio, “A paral-

lel mixture of SVMs for very large scale problems," Neural Comput., vol.14, no.5, pp.1105–1114, 2002.
[4] J.C. Platt, Advances in Kernel Methods: Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pp.185–208, MIT Press, Cambridge, MA, 1999.
[5] R.E. Fan, P.H. Chen, and C.J. Lin, "Working set selection using second order information for training SVM," J. Machine Learning Research, vol.6, pp.1889–1918, 2005.
[6] I.W. Tsang, J.T. Kwok, and P.-M. Cheung, "Core vector machines: Fast SVM training on very large data sets," J. Machine Learning Research, vol.6, pp.363–392, 2005.
[7] I.W. Tsang, A. Kocsor, and J.J.T. Kwok, "Simpler core vector machines with enclosing balls," Proc. 24th International Conference on Machine Learning, Corvallis, Oregon, USA, June 2007.
[8] T. Joachimes, "Training linear svms in linear time," Proc. ACM Conference on Knowledge Discovery and Data Mining, pp.217–226, 2006.
[9] V. Franc and S. Sonnenburg, "Optimized cutting plane algorithm for support vector machines," Proc. 25th International Conference on Machine Learning, pp.320–327, 2008.
[10] D.R. Wilson and T.R. Martinez, "Reduction techniques for instance-based learning algorithms," Mach. Learn., vol.38, no.3, pp.257–286, 2000.
[11] Y.J. Lee and O.L. Mangasarian, "RSVM: Reduced support vector machines," Proc. First SIAM International Conference on Data Mining, pp.1–17, 2001.
[12] L. Shih, J.D.M. Rennie, Y.-H. Chang, and D.R. Karger "Text bundling: Statistics-based data reduction," Proc. 20th International Conference on Machine Learning, pp.696–703, Aug. 2003.
[13] D. Boley and D. Cao, "Training support vector machine using adaptive clustering," Proc. 4th SIAM International Conference on Data Mining, pp.126–137, April 2004.
[14] N. Vasconcelos, P. Ho, and P. Moreno, "The Kullback-Leibler kernel as a framework for discriminant and localized representations for visual recognition," European Conference on Computer Vision, vol.3023, pp.430–441, 2004.
[15] H. Yin and N.M. Allinson, "Self-organizing mixture networks for probability density estimation," IEEE Trans. Neural Netw., vol.12, no.2, pp.405–411, March 2001.
[16] F. Shen and O. Hasegawa, "An on-line learning mechanism for unsupervised classification and topology representation," International Conference on Computer Vision and Pattern Recognition (CVPR 2005), San Diego, CA, USA, June 2005.
[17] F. Shen and O. Hasegawa, "An incremental network for on-line unsupervised classification and topology learning," Neural Netw., vol.19, pp.90–106, 2006.
[18] T.M. Martinetz, "Competitive hebbian learning rule forms perfectly topology preserving maps," Proc. Int'l Conf. on Artificial Neural Networks, pp.427–434, 1993.
[19] T.M. Martinetz and K.J. Schulten, "Topology representing networks," Neural Netw., vol.7, no.3, pp.507–522, 1994.
[20] T. Jebara, R. Kondor, and A. Howard, "Probability product Kernel," J. Machine Learning Research, vol.5, pp.819–844, 2004.
[21] Forest CoverType: http://kdd.ics.uci.edu/databases/covertype/covertype.html
[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE, vol.86, no.11, pp.2278–2324, 1998.
[23] J.X. Dong, A. Krzyzak, and C.Y. Suen, "A fast SVM training algorithm," Int. J. Pattern Recognition and Artificial Intelligence, vol.17, no.3, pp.1–18, 2003.
[24] KDD Cup 1999, WEB, IJCNN, USPS are form http://www.cse.ust.hk/~ivor/cvm.html

（平成 20 年 12 月 5 日受付，21 年 1 月 26 日再受付）

笠井　航

2005 東工大・工・情報工学卒．2007 同大大学院知能システム科学専攻修士課程了．現在，同大学院総合理工学研究科知能システム科学専攻博士後期課程に在籍中．主として，パターン認識に関する研究に従事．

戸部雄太郎

2007 東工大・工・情報工学卒．現在，同大大学院総合理工学研究科知能システム科学専攻修士課程に在籍中．

申　富饒

1995 中国南京大・数学卒．1998 同数学専攻卒．2006 東京工業大学大学院総合理工学研究科知能システム科学専攻博士課程了．現在，中国南京大学計算機ソフトウェア新技術国家重点実験室准教授．

長谷川　修　（正員）

1993 東京大学大学院博士課程了．博士（工学）．同年電子技術総合研究所入所．1999 年 6 月より 1 年間カーネギーメロン大学ロボティクス研究所滞在研究員．2001 産業技術総合研究所主任研究員．2002 年 5 月東京工業大学大学院理工学研究科付属像情報工学研究施設准教授．2002 から 3 年間科技団さきがけ研究 21 研究員．パターン認識，ニューラルネット，実世界知能システムなどの研究に従事．人工知能学会，日本認知科学会，日本顔学会，IEEE CS 等各会員．