仮想サポートベクトルによるサポートベクトルマシンの高速学習 ― Condensed Vector Machine の提案と評価 ―

松本　一則　　服部　元　　柳原　正　　池田　和史　　滝嶋　康弘

グエン　ズン・ディユック†　　橋本　和夫‡

KDDI 研究所　〒356-8502 埼玉県ふじみ野市大原 2-1-15

† Institute of Information and Technology, Vietnum

‡ 東北大学大学院情報科学研究科　〒980-8579 宮城県仙台市青葉区荒巻字青葉 6-6-11

E-mail:　{matsu, gen, td-yanagihara, kz-ikeda, takishima}@kddilabs.jp

あらまし　本稿では Condensed Vector Machine と名付けたカーネル学習の手法を提案する．提案の学習アルゴリズムは，大規模学習に適した decomposition の枠組みに，サポートベクトル簡約化の処理を統合したものである．提案手法は，学習の全過程においてワーキングセットのサイズを小さく保つことが可能なため，訓練に必要なメモリと訓練時間が少なくて済む．また，学習で得られる SV 集合自体がコンパクトになっているので，学習後に別途実施される決定関数の簡単化処理が不要にもかかわらず，高速な判定処理が実現できる．著名な実装である libSVM と Condensed VM との比較を 50 万件の訓練データを用いて行った． libSVM に比べ，Condensed VM は判定精度を落とさずに，訓練で 86.7 倍，得られた SV を用いた判定処理で 83.2 倍高速であった．

キーワード　サポートベクトルマシン，サポートベクトルの簡約化，ワーキングセットの逐次構築

Fast Learning Algorithm for Vector Machines with Artificial Support Vectors ― Proposal and Evaluation of Condensed Vector Machines ―

Kazunori MATSUMOTO　Gen HATTORI　Tadashi YANAGIHARA　Kazushi IKEDA

Yasuhiro TAKISHIMA　NGUYEN Dun Duc　and　Kazuo HASHIMOTO

KDDI R&D Laboratories,　2-1-15 Ohara, Fujimino-shi, Saitama, 356-8502 Japan

† Institute of Information and Technology, 18 Hoang Quoc Viet, Hanoi 10000, Vietnam

‡ Graduate School of Information Sciences, Tohoku University, 6-3-09 Aramaki-Aza-Aoba,　Aoba-ku, Sendai, 980-8579, JAPAN

E-mail:　{matsu, gen, td-yanagihara, kz-ikeda, takishima}@kddilabs.jp

Abstract　This manuscript proposes the kernel learning algorithm called as Condensed Vector Machine.　The proposed method is an integration of the decomposition schema for scalable learning and the simplification technique of support vectors. The key features of the method are to save consumed memory and processing time of learning, because the algorithm can keep the small working-set size on the whole training process.　And more, the processing time of decision function is fast without vector simplification, as the set of trained vectors has already enough small.　In the experiments which compares Condensed Vector Machine and libSVM which is one of the most famous implementation of SVM, the training time of Condensed SVM achieves 86.7 times faster than that of libSVM, and the execution of the decision is also 83.2 times faster .

Keyword　Support Vector Machine,　Simplification of support vectors,　Incremental working set construction

1. はじめに

サポートベクトルマシン(SVM) の学習は訓練事例ごとに適切な重み係数を与える最適化問題とみなせる．この最適化問題ではメモリ量と処理時間は訓練事例数の２乗および３乗に比例する．このため，大規模な問題で SVM の学習を行うことは困難である．また，未知の事例に対する SVM 判定処理は，先の重み係数が 0 でない訓練事例の特徴量すなわちサポートベクトル

(SV)の数に比例することから，学習で得られる SV の数が大きくなる大規模問題では評価時の判定処理にも時間がかかる．

そこで，評価時の判定処理を高速化するために reduced set (RS) 法[1][2]では，本来の決定関数(decision function)

$$y = \mathrm{sign}\left(f(x) = \sum_{i=1}^{N_S} \alpha_i K(x, x_i) + b \right)$$

から，元のサポートベクトル数 Ns より十分小さい Nz 個の新たな特徴ベクトルで表現される簡約版の決定関数

$$y' = \mathrm{sign}\left(f'(x) = \sum_{i=1}^{N_Z} \beta_i K(x, z_i) + b \right)$$

を構築する．このとき，オリジナルのサポートベクトル $x_1, x_2, \cdots, x_{N_s}$ の重み係数は $\alpha_1, \alpha_2, \cdots, \alpha_{N_s}$ であり，簡易版の特徴ベクトル $z_1, z_2, \cdots, z_{N_z}$ の重み係数は $\beta_1, \beta_2, \cdots, x\beta_{N_z}$ である．$K(x, x') = \Phi(x) \cdot \Phi(x')$ はカーネル関数であり，写像空間上のベクトル $\Phi(x)$ と $\Phi(x')$ の内積である．RS 法[1][2]では，学習終了後に，新サポートベクトルとその重み係数を下記の式で求めており，種々のデータセットに対し，$f(x)$ と同程度の識別性能を持つ $f'(x)$ を実現できることが示されている．

$$\min \leftarrow \left\| \sum_{i=1}^{N_S} \alpha_i \Phi(x_i) - \sum_{i=1}^{N_Z} \beta_i \Phi(z_i) \right\|$$

本稿では，従来，学習終了後に実施されていたサポートベクトルの削減による決定関数の簡約化を，学習の過程で行う手法を提案する．提案手法は，学習中に，適切に選んだ 2 個の SV から最適な特徴ベクトルをもつ仮想 SV を生成する．筆者らは，こうした仮想 SV で実現される判別器を Condensed Vector Machine と呼ぶことにした．

Condensed Vector Machine の学習では，仮想 SV の数の増大を抑えることができるため，使用するメモリが一定量の高速学習が可能になる．また，決定関数に使用する特徴ベクトルの数も SVM より少ないため，判定処理も高速化できる．さらに，提案手法はオンライン学習のアルゴリズムを採用しているため，学習事例を適時追加しながら未知事例に対する判定処理を行うこともできる．

提案手法によって精度を落とすことなく，学習時間を大幅に削減でき，さらに判定処理も高速化できたことを報告する．

2. Condensed VM 学習の利点とその理由

本章では，SVM に比べて，なぜ，Condensed VM が大規模問題を高速に学習できるかを説明する．

大規模問題で SVM の学習が困難になる理由は，以下の Quadratic Programming Problem (QP 問題) を解く必要があるからである．

$$\min_{\alpha} \quad L(\alpha) = \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j K_{ij} - \sum_{i=1}^{l} \alpha_i ,$$

$$\text{s.t.} \quad \sum_{i=1}^{l} y_i \alpha_i = 0, \quad 0 \le \alpha_i \le C, i = 1, \ldots, l.$$

ただし，$K_{ij} = K(x_i, x_j)$; C は訓練事例 $T = \{(x_i, y_i) | x_i \in R^d, y_i = \pm 1, i = 1, \ldots, l\}$. に含まれる noisy な事例に対するペナルティを表す．

上記の QP 問題解くためには，カーネルマトリックス K_{ij} の値を保存するための $O(l^2)$ のメモリと $O(l^3)$ の処理時間が必要である．このため，Divide-and-conquer の考え方に従い，大きな QP 問題をいくつかの小さな QP 問題に還元する decomposition アルゴリズムが提案されている[3][4][5].

decomposition のフレームワークを表 1 に示す．decomposition の基本的な考え方は，訓練データの集合をアクティブ・ワーキングセット B_t と非アクティブ N_t の 2 つに分け，各訓練用サンプルに対する重み係数の更新をアクティブ B_t に限定し，非アクティブ N_t の重み係数は一時的に固定したままにする．通常，B_t は T に比べて十分に小さいので，たいていの問題が高速に解けるとされている．しかし，SV の数が多い大規模問題の場合，最終的な B_t の大きさは SV の数になるため，表 1 の step5 と step6 の処理に時間がかかることが指摘されている[6].

提案の Condensed VM の学習アルゴリズムは RS 法による SV の簡約化と decomposition のフレームワークを組み合わせたものである．SV の簡約を学習時に行うことで，学習の全過程においてワーキングセット B_t のサイズを小さく保つことができ，表 1 の step 5 と 6 が高速に行えるのである．B_t の大きさを最小限に保てることはいくつかの利点を生む．第一に step6 の最適解を求めるのに必要なメモリと計算時間が小さくて済む．また，最適解に使用する SV の数が簡約化によって少なくて済むので step5 も高速化される．さらに，学習で得られる SV 集合自体がコンパクトになっているので，RS 法で決定関数を簡単化しなくても高速な判定処理が実現できる．

3. Condensed VM の学習アルゴリズム

Condensed VM の学習は，簡約化された仮想 SV 集合とその重み係数を計算する過程である．本章では，簡約のために選ばれた 2 個の SV から仮想 SV を求める

方法(3.1)，簡約化対象となる 2 個の SV を選び出す方法(3.2)，Condensed VM の逐次学習を行うための手順(3.3)を説明する．

表 1 SVM 学習における decomposition のフレームワーク

Initialization
1. $T = B_0 \cup N_0$
2. Find initial solution f_0 on B_0

Main optimization loop
3. Set iteration $t = 0$
4. **While** *StoppingCondition* is not satisfied
5. Use f_t to update $B_{t+1} \leftarrow B_t$, $N_{t+1} = T - B_{t+1}$
6. Find temporal SVM solution f_{t+1} on B_{t+1}
7. $t = t + 1$
8. **Endwhile**

3.1. 仮想ベクトルの生成方法

同一クラスに属するサポートベクトル x_1 と x_2 から仮想 SV である z を生成するとする．オリジナルの SVM と Condensed SVM との差を最小にすることから，z^*は次式で求めることにする．

$$z^* = \arg\min_z \left\| D_2\Phi(z) - \left(C_1\Phi(x_1) + C_2\Phi(x_2)\right)\right\|^2$$
$$= \arg\min_z D_2\|\Phi(z) - M\|$$

ただし $M = (m\Phi(x_1) + (1-m)\Phi(x_2))/2$ ，
$m = C_1/D2$， D2=C1+C2 である．

これは，特徴ベクトルz^* の写像空間上の位置$\Phi(z^*)$と $\Phi(x_1)$ と $\Phi(x_2)$ の重み付き重心との距離を最小化することを意味している．このようにz^* を定めるとz^*はx_1 と x_2 から効率よく得られることが，RS 法における筆者らの先行研究[2]で分かっている．実際，ガウスカーネル
$K(x,y) = \exp(-\gamma\|x-y\|^2)$ の場合，z^*は以下のようになる．

$$z^* = k^* x_i + (1-k^*)x_j$$

ただし $k^* = \arg\max_k g(k) = mK_{12}^{(1-k)^2} + (1-m)K_{12}^{k^2}$
$K_{12} = K(x_1, x_2)$;

なお，多項式カーネルでも同様に 2 つの SV から仮想ベクトルを効率良く求めることができる[2].

3.2. 簡約化対象の求め方

前節の手法の場合，K12 が大きいほど，$\Phi(z^*)$と M の距離は小さくなり，オリジナルの SVM と Condensed VM との差も小さくなる．そこで，Condensed VM の学習では，簡約対象の特徴ベクトルを選び出す手法として下記ヒューリスティックスを用いることにする．

$$(x_1, x_2) = \arg\max_{y_i y_j > 0, i \neq j} K(x_i, x_j).$$

ただし大規模問題の場合，上記手法のようにすべての組み合わせで簡約化対象を選び出すとコストがかかりすぎる．そこで，次節では、逐次学習の枠組みで効率良く簡約対象を求める手法について述べる。

3.3. 逐次学習の実現

大規模問題に対して効率良く学習を行うことを目的として、ワーキングセットを逐次大きくしていく手法が提案[7][8][9][10]されている，逐次アルゴリズムの場合，最少サイズのワーキングセットで学習を始めて，新たに適切な訓練事例を 1 個ずつワーキングセットに加えながら，より良い学習解を求めていく．本提案では，逐次追加の枠組に，2 段階式の逐次学習のアルゴリズムが特徴の i-SVM[10]の手法を採用し，学習しながらの簡約を行うこととした． 表 2 は i-SVM の枠組みを取り込んだ Condensed VM 学習のアルゴリズムである．以下にその詳細を示す．

1) 初期化(Step 1)
選び出した 2 個の訓練用サンプルだけでワーキングセットを構成し，2 個の重み係数だけを最適解に近づくように更新する Sequential Minimal Optimization (SMO) の手法[11]を使って，2 個の学習事例から最初のワーキングセットB_0 を構築する．2-クラスの識別問題の場合，単に異なるクラスの事例を 2 個選び出すことになる．

2) 最適化(Step 2, 6)
ループの各段階で，ワーキングセットの中から 2 個の訓練事例が選択され，それらの事例の重み係数が更新される．最適化関数 L の偏微分から，下記の式に従って係数の更新を行えば最適化関数 L の減少量が最大になることが分る．

$$\alpha_i^{new} = \alpha_i^{old} + \frac{y_i(E_j^{old} - E_i^{old})}{K_{ij}}$$
$$\alpha_j^{new} = y_i(const - y_i\alpha_i^{new})$$

ただし $const = y_i\alpha_i^{old} + y_j\alpha_j^{old}$,
$K_{ij} = K_{ii} + K_{jj} - K_{ij}$,,
$E_i = \sum_{k=1}^{|B_t|} y_k\alpha_k K(x_k, x_i) - y_k$ である．

この時，最適化関数 L の減少量は

$\Delta L_{ij} = -\frac{(E_j^{old} - E_i^{old})^2}{2K_{ij}}$ である．

3) 非ワーキングセットからの事例選択(Step 5.1)

ここでは，最適化関数 L の減少量を最大化するような訓練事例を非ワーキングセットから 1 つ選択する．ここで注意すべきことは，より適切な訓練事例を求めることの利点とそのための計算コストのトレードオフである．以下では，トレードオフをバランスさせるために我々が導入する 2 段式の選択手法を説明する．

t 回目の繰り返しで得られる SVM の解を

$$f_t(x) = \sum_{k=1}^{|B_t|} y_k \alpha_k K(x_k, x) + b_t$$

とし，非ワーキングセット N_t に x_i が含まれるとする．この時，

$$y_t f_t(x_i) - 1 < 0$$

であるならば $\alpha_i = 0$ が成り立たなくなる．また，上式の $y_t f_t(x_i)$ が小さいほど，最適化関数の収束が良くなる．しかし，Bt のサイズが大きいと $y_t f_t(x_i)$ を求めるための計算コストが大きくなってくる．そこで，訓練の初期段階では「サンプリングを用いた選択」，それ以降は，「総あたりによる選択」を行う．各選択手法の概要は以下の通り．

サンプリングを用いた選択：

訓練の初期段階では，ワーキングセット Bt のサイズが小さく，$y_t f_t(x_i)$ の計算コストも小さい．そこで，質の良い訓練事例を得るため，最適化のループ毎に，非ワーキングセットから抽出した m 個の訓練事例の集合 S から，

$$x_t = \arg\min_{x_k \in S} y_k f_t(x_k) - 1$$

となる事例を最善な学習事例とみなして選択する．しかし，Bt が大きくなるにつれ，計算に時間がかかるようになるので，以下のいずれかの条件が満たされたら，選択戦略を切り替える．

条件１　抽出した m 個のどの事例も $y_t f_t(x_i) - 1 < 0$ ではない．(m=1000)

条件２　Bt の各事例での $y_i f_t(x_i) - 1 < 0$ のチェック回数の平均が 1 を超える．

条件３　Bt のサイズが事前設定した大きさ(=1,000)を超える

総あたりによる選択：

１段目の「サンプリングを用いた選択」の段階で得られる SVM の解は，必ずしも良い近似解であるとは限らない．そこで，この段階では，Nt に含まれる事例に対して，総あたりで $y_t f_t(x_i) - 1 < 0$ のチェックを行うとする．ただし，前回の繰り返しで $y_i f_{t-1}(x_i) - 1 < 0$ でなかったのに，次の繰り返しで $y_i f_t(x_i) - 1 < 0$ であるかをチェックしても無駄になると考えられるので，前回チェックした xt については，チェックを行わない．チェックによって最初に見つかった事例を選択する．

4) ワーキングセットの更新(Step 5.2-5.8)

ここでは，「簡約化」，「SV の削減(shrinking)」，「ワーキングセットへの事例追加」のいずれかの処理が行われる．各処理の概要は以下の通り．

簡約化：

非ワーキングセットから選択した事例 xt に対し，現時点でのワーキングセット Bt のとある特徴ベクトルとで簡約化を行うかどうかの判断が行われる．簡約化の条件を以下に示す．

$$\begin{cases} x_i = \arg\max_{x_k \in B_t, \alpha_k > 0, y_k y_t > 0} K(x_k, x_t), \\ K(x_i, x_t) \geq \theta. \end{cases}$$

これは，xt に最もよく似たベクトル xi とで簡約化が起こりえることを表している．この条件が満たされた場合，xi が取り除かれ，xi と xt とから生成された仮想ベクトルがワーキングセット Bt に追加される．この処理がおおなわれても，Bt のサイズは変化しない．

SV の削減(shrinking)：

xt の簡約化対象が Bt の中に見つからない場合の処理である．ワーキングセット Bt に含まれる非 SV を 1 個見つけ，Bt から取り除く．この処理は，QP のサイズを減らす効果があり，shrinking technique[12]という名称で知られている．

ワーキングセットへの事例追加：

簡約化も SV の削減も行われなかった時の処理で，非ワーキングセットから選ばれた事例 xt がワーキングセット Bt に追加される．

5) 終了判定(Step 4)

理論的には，非ワーキングセットの事例の中に $y_i f_t(x_i) - 1 < 0$ の条件を満たすものが無くなったときが終了の条件である．しかし，Condensed VM の解が安定していて，判定時の精度も十分であることから，「総あたりによる選択」で候補

となる事例が見つからなくなった段階で終了するのが現実的である．

表 2 Condensed SVM の学習アルゴリズム

Initialization
1. $B_0 = \{x_1, x_2\}, y_1 y_2 > 0, N_0 = T - B_0$
2. Find initial solution f_0 on B_0

Main optimization loop
3. Set iteration $t = 0$
4. **While** *StoppingCondition* is not satisfied
5.1. Select $x_t \in N : f_t(x_t) - y_t < 0$
5.2. **If** there exists $x_i \in B_t : \alpha_i = 0$ **Then**
5.3. $B_{t+1} = B_t - \{x_i\} \cup \{x_t\}, N_{t+1} = N_t \cup \{x_i\}$
5.4. **Else** $B_{t+1} = B_t \cup \{x_t\}$
5.5. $N_{t+1} = N_t - \{x_t\}$
6. Find solution f_{t+1} on B_{t+1}
7. $t = t + 1$
8. **Endwhile**

4. 実験と評価

decomposition の枠組みを生かしつつ，学習時に SV の簡約化を行う Condensed VM は，アルゴリズムの性質から高速な学習が期待される．また，最終的な SV 数の少なさによって判定処理時間の短縮も期待される．そこで，学習に要する時間と判定時間を，decomposition の枠組みを利用した，優れた実装として広く知られる libSVM [12]と比較する．

4.1. 実験環境

比較に使用するデータは，swis roll と呼ばれる 3 次元の人工データで以下のようにして生成した．

$x_\tau = r_\tau \cdot \cos\theta_\tau$
$y_\tau = r_\tau \cdot \sin\theta_\tau$
$z_\tau = [-1 \sim +1 \text{ の一様分布}]$

ただし $\theta_\tau = \tau + \varepsilon_1$ ，と $r_\tau = \alpha \cdot \theta_\tau + \varepsilon_2$ $(\alpha = \frac{1}{3\pi},)$ であり，$\varepsilon_1, \varepsilon_2$ は平均値 0，分散 σ の正規乱数．

2 クラスの訓練事例および評価用事例の数はそれぞれ 50 万件で，正クラスの事例は $0 < \tau < {}^{3\pi}/_{2}$ ，負事例のクラスは ${}^{3\pi}/_{2} < \tau < 3\pi$ に分布させた．事例の散らばり具合を制御するための分散は $\sigma = 0.05$ とした．

学習は libSVM と Condensed VM で同じガウスカーネル（γ =0.1）と同じコスト値（C=1.0）を用いた．

4.2. 実験結果

図 1 と図 2 は訓練用事から得られた SV を訓練用事例のプロットに重ねて 3 次元表示したものである．ただし，見やすくなるよう，学習用事例は正例・負例とも 5 万件に絞ってプロットしている．図 1 では libSVM で得られる SV を，図 2 では Condensed SVM で得られる SV を重畳した．図 1，図 2 を比べてわかるように，それぞれの手法が生成する SV の位置はほぼ同じ場所に分布しているが，SV の数は，Condensed SVM の方が少ない．

表 3 は， libSVM と提案の Condensed SVM の性能比較である．提案の Condensed SVM は，libSVM とほぼ同じ分類精度でありながら，生成した SV の数は非常に少ない．そして，Condensed SVM の訓練と判定の各処理は，libSVM に比べてそれぞれ 86.7 倍，83.2 倍高速であった．

5. おわりに

本稿では Condensed Vector Machine と名付けたカーネル学習の一手法を提案した．提案の Condensed VM の学習アルゴリズムは RS 法による SV の簡約化と decomposition のフレームワークを組み合わせたものである．SV の簡約を学習時に行うことで，学習の全過程においてワーキングセットのサイズを小さく保つことができる．ワーキングセットのサイズを最小限に保てることで，訓練に必要なメモリと計算時間が小さくて済む．また，学習で得られる SV 集合自体がコンパクトになっているので，RS 法で決定関数を簡単化しなくても高速な判定処理が実現できる．

50 万件の訓練データを用いた評価では，decomposition や shrinking の実装として著名な libSVM と Condensed VM との比較を行った．Condensed VM は libSVM に比べ，訓練で 86.7 倍，得られた SV を用いた判定処理で 83.2 倍高速であった．

表 3 libSVM と Condensed VM の性能比較

	libSVM	Condensed VM
生成した SV の数	3,690	26
訓練時間(秒)	1,926.38	22.24
判定時間(秒)	111.51	1.34
acculacy (%)	99.76	99.74

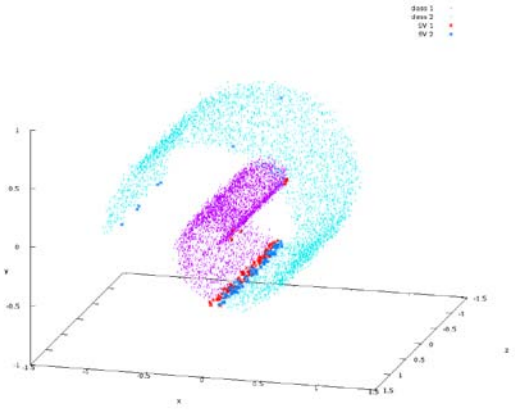

図 1 訓練事例と libSVM によって得られた SV の分布

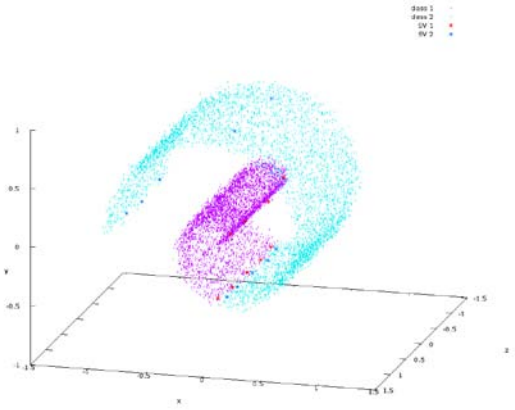

図 2 訓練事例と Condensed SVM によって得られた SV の分布

文　献

[1] C. J. C. Burges, “Simplified support vector decision rules,” in Proc. 13th International Conference on Machine Learning, San Mateo, CA, 1996, pp. 71–77.

[2] D. D. Nguyen and T. B. Ho, “A bottom-up method for simplifying support vector solutions,” IEEE Transactions on Neural Networks, vol. 17, no. 3, pp. 792–796, 2006.

[3] E. E. Osuna, R. Freund, and F. Girosi, “Support vector machines: Training and applications,” Tech. Rep., 1997.

[4] T. Joachims, “Making large-scale support vector machine learning practical,” in Advances in Kernel Methods: Support Vector Machines, A. S. B. Scholkopf, C. Burges, Ed. MIT Press, Cambridge, MA, 1998.

[5] J. Platt, “Fast training of support vector machines using sequential minimal optimization,” in Advances in Kernel Methods - Support Vector Learning, B. Schoelkopf, C. J. C. Burges, and A. J. Smola, Eds

[6] B. L´eon, C. Olivier, D. Dennis, and W. Jason, Eds., Large Scale Kernel Machines. Cambridge, MA.: MIT Press, 2007.

[7] G. Cauwenberghs and T. Poggio, “Incremental and decremental support vector machine learning,” in NIPS, 2000, pp. 409–415.

[8] L. Pavel, G. Christian, K. Stefan, and M. Klaus-Robert, “Incrementalsupport vector learning: Analysis, implementation and applications,” J.Mach. Learn. Res., vol. 7, pp. 1909–1936, 2006.

[9] B. Antoine, E. Seyda, W. Jason, and B. L´eon, “Fast kernel classifiers with online and active learning,” Journal of Machine Learning Research, vol. 6, pp. 1579–1619, September 2005.

[10] D. D. Nguyen, K. Matsumoto, Y. Takishima, K. Hashimoto, and M. Terabe, “Two-stage incremental working set selection for fast support vector training on large datasets,” Research, Innovation and Vision for the Future, 2008. RIVF 2008. IEEE International Conference on, pp. 221–226, July 2008.

[11] J. Platt, “Fast training of support vector machines using sequential J. Platt, “Fast training of support vector machines using sequential minimal optimization,” in Advances in Kernel Methods - Support Vector Learning, B. Schoelkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, pp. 185–208.

[12]