Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1

頑健なスパースカーネル分類器の学習

ブロンデル マチュー [†1]　関　和広[†1]　上原邦昭[†1]

カーネル分類器は多くのデータセットに対して優れた精度を示すことが分かっている．しかし，カーネル分類器のモデルの複雑性は訓練事例数に応じて線形に増加するため，訓練データの規模が大きくなるほど効果的にカーネル分類器を学習することが難しくなる．本研究では，スパースカーネル分類器を学習するための新しい逐次最適化アルゴリズムを提案する．提案アルゴリズムは，カーネルパーセプトロンと kernel matching pursuit に着想を得たものであり，a）訓練データを有効に使用できる，b）ラベルノイズに頑健である，c）任意の損失関数を利用できる，d）実装も容易であるという多くの特長がある．複数のデータセットで評価実験を行ったところ，多くの実験設定において，提案手法は従来手法と同等か高い精度を示すことが明らかになった．

Learning Robust Sparse Kernel Classifiers

Mathieu Blondel ,[†1] Kazuhiro Seki [†1]
and Kuniaki Uehara[†1]

Despite state-of-the-art accuracy on many real-world datasets, kernel classifiers remain notoriously difficult to train efficiently because the model complexity has a linear dependency with the number of training instances. In this paper, we propose a novel incremental optimization algorithm for learning sparse kernel classifiers in the primal. Strongly influenced by the kernel perceptron and kernel matching pursuit, our algorithm makes efficient use of training data, is robust to label noise, can employ any convex subdifferentiable loss function and is simple to implement. Extensive experiments on several standard datasets show that our algorithm achieves comparable or better performance than several existing methods.

†1 神戸大学
Kobe University

1. はじめに

過去 10 年間以上に渡り，カーネルサポートベクトルマシン（カーネル SVM）は盛んに研究されてきた．カーネル SVM は特徴を高次元空間に拡張するため，多くのデータセットに対して優れた精度を示している．また，系列，木，グラフ等を効率的に操作できるカーネル関数が存在するので，カーネル SVM はそのような非ベクトルデータも自然に利用できる．一方，最近の研究によって，線形 SVM の効率的な学習が可能になったのに対し，カーネル SVM の効率的な学習は未だに難しい問題である．これは，モデル中のサポートベクトルの数が事例数と共に増加することによる．さらに，カーネル SVM の予測関数がサポートベクトルに依存するため，予測に要するコストも高くなる．Wang ら[17] は，この問題を *the curse of kernelization*（カーネル化の呪い）と呼んでいる．

本稿では，学習と予測の両方の問題に対処するため，スパース性の制約を守る（すなわち，サポートベクトル数を制限する）カーネル分類器の学習について検討する．この NP 困難な組み合わせ問題を緩和するため，スパース性を守りつつ，計算コストを低く抑える逐次最適化アルゴリズムを提案する．提案アルゴリズムは，カーネルパーセプトロン（KP）[10] と kernel matching pursuit （KMP）[16] の長所を組み合わせている．その結果，KMP と同じように，提案アルゴリズムは任意の劣微分可能な凸損失関数が利用できる．このため，SVM で用いられるヒンジ損失だけでなく，たとえば確率的な予測が必要な場合は，対数損失を用いてスパースカーネルロジスティック回帰を行うことも出来る．また，提案手法では重みバックフィッティングと呼ぶ方法により，サポートベクトルを追加せずにモデルを改良することができる．よって，モデルの複雑性を低く抑えつつ，訓練事例を有効に利用することができる．また，KMP が全データセット上の計算コストの高いラインサーチを行うことと対照的に，提案アルゴリズムでは，KP と同様にサポートベクトルを効率的に追加することができる．KP と KMP はそれぞれオンライン的，バッチ的にサポートベクトルをモデルに増分的に追加していく．提案アルゴリズムはその中間的な方法であり，ランダム探索を用い，訓練データの一部のサンプルの中から有望なサポートベクトルを選択する．サポートベクトルの選択についてはこれまでに様々な方法が提案されており，本研究では，確率的な枠組によるラベルノイズに頑健な選択方法を提案する．提案アルゴリズムは，学習が効率的，かつ実装が容易という特長も持つ．複数のデータセットを用いた評価実験により，提案手法の有効性を検証する．

以降，2 章で, 本研究を把握するために重要となる概念について述べ，3 章では，提案ア

Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1

ルゴリズムを詳説する．4 章では実験の結果を示す．5 章で関連研究について述べ，6 章では，本研究のまとめと今後の課題について述べる．

2. 背　　景

本章では，提案アルゴリズムの重要な要素である線形分類器について述べる．続いて，主問題による分類器のカーネル化について述べ，カーネルの出力値を類似度素性と見なす概念について述べる．最後に，スパース性の制約を守る（サポートベクトル数を制限する）カーネル分類器の学習問題を形式的に定義する．説明を簡潔にするため，以降では 2 値分類問題について議論する．しかし，*one-vs-all* のようなヒューリスティックを使うことで，提案アルゴリズムは容易に多クラス分類に拡張できる．

2.1 線形分類器の学習

ラベル $y_* \in \mathcal{Y} = \{-1, +1\}$ を予測するため，線形分類器が入力 $\mathbf{x}_* \in \mathcal{X}$ を特徴ベクトル $\phi(\mathbf{x}_*)$ に変換し，以下の識別関数の符号を求める．

$$f(\mathbf{x}_*; \mathbf{w}) = \langle \mathbf{w}, \phi(\mathbf{x}_*) \rangle \tag{1}$$

特徴に対する重み $\mathbf{w} \in \mathbf{R}^d$ を学習するには, 通常，経験的リスク最小化の枠組が使われる．例えば，事例の集合 $\mathbf{x}_1, \ldots, \mathbf{x}_n$ と関連するラベル $y_1, \ldots, y_n$ を与えたときに, 以下の L_2 正則化付き主問題を最小化できる．

$$\underset{\mathbf{w}}{\text{minimize}} \ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(\mathbf{x}_i; \mathbf{w})) + \frac{\lambda}{2} ||\mathbf{w}||_2 \tag{2}$$

$\lambda > 0$ は正則化を制限するハイパーパラメータであり，$\ell(y_i, \hat{y}_i)$ は予測ラベル $\hat{y}_i = f(\mathbf{x}_i)$ が正解ラベル y_i と異なる際の損失関数である．良く使われる損失関数として，パーセプトロン損失 $\ell(y_i, \hat{y}_i) = \max(0, -y_i \hat{y}_i)$，ヒンジ損失 $\ell(y_i, \hat{y}_i) = \max(0, 1 - y_i \hat{y}_i)$，対数損失 $\log(1 + \exp(-y_i \hat{y}_i))$，二乗損失 $(y_i - \hat{y}_i)^2$ がある．近年，目的である式 (2) を解くため，大規模な問題も扱える確率的劣勾配降下法が盛んに使われてきた[14),19)]．

2.2 カーネル分類器の学習

リプレゼンターの定理[1)] より，式 (2) の解は，訓練データの事例を線形に結合した $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i)$ で表現できる．よって，式 (1) を次のように書き直すことができる．

$$f(\mathbf{x}_*; \boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i \kappa(\mathbf{x}_*, \mathbf{x}_i) \tag{3}$$

ここで，$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \mathbf{R}^n$ は訓練事例に対する重みであり，$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ は $\mathbf{x}_i$ と $\mathbf{x}_j$ の間の類似度を示す半正定カーネルである．非ゼロの α_i に相当する事例はサポートベクトルと呼ばれる．加えて，α_i の符号は，関連するサポートベクトルのラベル（-1 か $+1$）を示している．いわゆる「カーネルトリック」は，線形分類器を非線形分類器に変換させることができるため，実用上，非常に強力である．カーネルが存在する限り，非ベクトルデータ（木，グラフ，系列等）にも自然に汎化できる．

従来の研究では，カーネル化を行うため，SVM に現れる目的関数を双対問題に変換することが一般的であったものの，Chapelle[7)] はリプレゼンターの定理を用いると，以下の主問題を最小化できると証明した．

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \ \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(\mathbf{x}_i; \boldsymbol{\alpha})) + \frac{\lambda}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \tag{4}$$

式 (2) と同じように，式 (4) は標準的な確率的劣勾配降下法を用いて解けるように見える．しかし，最終的な解と違い，$\boldsymbol{\alpha}$ に関する勾配がスパースではないため，各反復で $\boldsymbol{\alpha}$ を更新するには n 回のカーネル計算が必要となる．よって，この方法で大量のデータを扱うことは現実的ではない．一方で，カーネルパーセプトロン[10)] とカーネル pegasos[14)] は主問題を効果的に最適化できる．それは，$\mathbf{w}$ に関する更新をたった一つの α_i に関する更新に変換できるからである．

ここで，もしサポートベクトルの集合が既知，かつ固定されていると仮定すると，入力 $\mathbf{x}_*$ を以下のように n 次元のスパースベクトルに射影することができる．

$$\psi(\mathbf{x}_*) = (\ldots, \kappa(\mathbf{x}_*, \mathbf{x}_{S_1}), \ldots, \kappa(\mathbf{x}_*, \mathbf{x}_{S_B}), \ldots) \tag{5}$$

$S = \{i : \alpha_i \neq 0\}$ はサポートベクトルに対応する添字の集合であり，$B = |S|$ は集合の大きさである．式 (5) を用い，カーネル付きの識別関数である式 (3) を内積形で書き直せる．

$$f(\mathbf{x}_*; \boldsymbol{\alpha}) = \langle \boldsymbol{\alpha}, \psi(\mathbf{x}_*) \rangle \tag{6}$$

式 (6) は式 (1) と全く同じ形式であり，直感的に，$\mathbf{x}_*$ とサポートベクトルの間のカーネルの出力値を類似度として見なすことができる．通常，$B \ll n$ であるため，pegasos のような効率的なソルバーを用い，式 (2) を最適化することで，$\boldsymbol{\alpha}$ を学習することができる[⋆1]．3.4 節では，重みのバックフィッティングを行う際に，この考えを利用する．

2.3 スパースカーネル分類器の学習

標準的なソフトマージン SVM の解がスパースであることは良く知られているものの，実際はスパース性（サポートベクトル数）を明確に管理することはできない．また，SVM の解はさらにスパースにできる可能性があるという主張もある[13)] [p.555]．一方，カーネルリッ

⋆1 新しい目的関数では，正則化項が式 (4) と少し異なるものの，さらに効果的に計算できるメリットを持つ．

Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1

ジ回帰とカーネルロジスティック回帰の解は，密であることが良く知られている．さらにスパースな分類器を得る 1 つの方法は，サポートベクトル数の上限 B をあらかじめ固定することである．これは，L_0 擬似ノルムを用いた制約ありの目的関数と見なせる．

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \frac{1}{n}\sum_{i=1}^{n}\ell(y_i, f(\mathbf{x}_i;\boldsymbol{\alpha})) \\ \text{subject to} \quad & ||\boldsymbol{\alpha}||_0 \leq B \end{aligned} \tag{7}$$

ここで，$||\boldsymbol{\alpha}||_0$ は $\boldsymbol{\alpha}$ の L_0 擬似ノルムであり，$\boldsymbol{\alpha}$ 中の非ゼロ要素の数である．このような目的関数を最小化することは，NP 困難な組み合わせ問題であるため[2]，従来は L_1 正則化付きの制約なしの目的関数を最小化することが一般的であった（L_1 ノルムもスパースな解を見つけるため）．ただし，正則化のハイパーパラメータと B が直接関連しないので，前もって解がどれだけスパースになるかを知ることはできない．提案アルゴリズムはスパース性の制約を守る保証があり，また計算コストが低いというメリットもある．

3. 提案アルゴリズム

3.1 概　　要

本研究では，L_0 擬似ノルムを用いた制約ありの目的関数である式 (7) をより効率的に最適化するために，サポートベクトルの追加と重みバックフィッティングという 2 つのステップを繰り返す逐次最適化アルゴリズムを提案する．一番目のステップでは，ランダム探索で選択したサポートベクトルをモデルに追加する．二番目のステップでは，確率的劣勾配降下法を用い，既存のサポートベクトルの重みを修正する．後者のステップはサポートベクトルを追加しないので，モデルをスパースに抑えつつ，精度を向上させることができる．次節以降では，逐次最適化，サポートベクトル選択，重みバックフィッティングのそれぞれについて詳しく述べる．

3.2 逐次最適化

カーネルパーセプトロンと同様，提案アルゴリズムはサポートベクトルを現在のモデルに一個ずつ追加する．しかし，提案アルゴリズムはカーネルパーセプトロンと異なる点がいくつかある．まず，提案アルゴリズムは正則化を活用し，任意の損失関数が利用できる．また，訓練事例を連続して一個ずつ取得するのではなく，ランダム探索を用い，m 個の事例を含んだ集合 $A \subset [n] = \{1, \ldots, n\}$ を利用する．続いて，提案アルゴリズムの概要とサポートベクトルの重みの初期化について述べる．

【概要】Algorithm 1 では，サポートベクトル候補 C は訓練データの全事例に初期化される．そして，各反復において *subsample* 関数を用いて m 個の事例を含んだ集合 A を取得した後，A の中で有望なサポートベクトルを *select* 関数で 1 個選択する．失われた損失がゼロではなかった場合，選択されたサポートベクトルの重みを初期化し（次段落参照），候補 C から取り除く．*learning_rate* は更新の大きさを制御する関数である．L_0 の制約が保たれている間は，この過程を繰り返す．アルゴリズムが終了した際の最終的な解が $\boldsymbol{\alpha}$ であり，予測関数は, $sign(f(\mathbf{x}_*;\boldsymbol{\alpha}))$ である．対数損失関数を使用した場合，確率的な出力が $p(y=+1|\mathbf{x}_*) = 1/(1+\exp(-f(\mathbf{x}_*;\boldsymbol{\alpha})))$ から得られる．

本アルゴリズムにおけるランダム探索の利用は，カーネルパーセプトロンと kernel matching pursuit に対する重要な違いとなっている．ランダム探索を使うことで，カーネルパーセプトロンよりずっと良いサポートベクトルの選択が行え，kernel matching pursuit より計算コストをずっと低く抑えれる．事実，ランダムで選択したわずか 59 個の訓練事例の中にあるもっとも良いサポートベクトルは，95%の確率で，全データセットのもっとも良い 5%のサポートベクトルに所属することが証明されている[13] [定理 6.33].

【重みの初期化】カーネルパーセプトロンは，$\mathbf{w} \leftarrow \mathbf{w} + y_i\phi(\mathbf{x}_i)$ の更新を $\alpha_i \leftarrow \alpha_i + y_i$ の更新に置き換え，$\phi(\mathbf{x}_i)$ の計算を予測時まで行わないという巧妙な方法でカーネル化を行う．ここで，2 つの重要な観測について述べる．まず，L_2 正則化はベクトルとスカラーの積を要するため，$\boldsymbol{\alpha}$ を正則化するには，$\mathbf{w}$ に関する正則化と全く同じ更新を行えば良い．次に，カーネルパーセプトロンは特定の損失関数を使用しているものの，$\mathbf{w}$ に関する更新が加法的であるかぎり，劣微分可能なすべての凸損失関数に対して同じ方法を利用できる．例えば，$\mathbf{v}_i$ を $\mathbf{v}_i = -\nabla_{\mathbf{w}}\ell(y_i, f(\mathbf{x}_i;\mathbf{w})) = u_i\phi(\mathbf{x}_i)$ としたとき, $\mathbf{w}$ に関する更新が $\mathbf{w} \leftarrow \mathbf{w} + \eta_t\mathbf{v}_i$ の形をとれば，α_i に関する更新は $\alpha_i \leftarrow \alpha_i + \eta_t u_i$ となる．Algorithm 1 では，これらの 2 つの知見を用いている．一般の損失関数において，Algorithm 1 に現れる *update* 関数が出力する値を以下にまとめる．

- パーセプトロン: $u_i = y_i$ if $y_i\hat{y}_i < 0$ and 0 otherwise,
- ヒンジ: $u_i = y_i$ if $y_i\hat{y}_i < 1$ and 0 otherwise,
- 対数: $u_i = y_i/(1+\exp(y_i\hat{y}_i))$,
- 二乗: $u_i = y_i - \hat{y}_i$.

3.3 サポートベクトルの選択

Algorithm 1 の重要な要素として，*select* 関数がある．まず，サポートベクトルを選択するための既存の戦略を 4 つ紹介し，ラベルノイズに対して頑健な *Loss-Probabilistic* という新しい戦略を提案する．

Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1

Algorithm 1: Incremental optimization

Input: D
Parameters: B, m, λ
Initialize SV coefficients: $\boldsymbol{\alpha} \leftarrow \mathbf{0}$
Initialize SV candidates: $C \leftarrow \{1, \ldots, n\}$
Set: $t \leftarrow 1$
while $||\boldsymbol{\alpha}||_0 < B$ **do**
  Sample: $A = subsample(C, m)$
  Select SV: $i = select(A, \boldsymbol{\alpha})$
  Compute prediction: $\hat{y}_i = f(\mathbf{x}_i; \boldsymbol{\alpha})$
  Compute update: $u_i = update(y_i, \hat{y}_i)$
  Set: $\eta_t = learning\_rate(t, \lambda)$
  Regularize: $\boldsymbol{\alpha} \leftarrow (1 - \eta_t \lambda)\boldsymbol{\alpha}$
  if $u_i \neq 0$ **then**
    Remove from candidates: $C \leftarrow C - \{i\}$
    Set SV coefficient: $\alpha_i \leftarrow \alpha_i + \eta_t u_i$
  end if
  Set: $t \leftarrow t + 1$
  [Optional: Back-fit coefficients with Algorithm 2]
end while
Output: $\boldsymbol{\alpha}$

【Random】この戦略はもっとも単純であり，単に集合 A から事例 i を 1 つだけ無作為にサンプリングする．しかし，ラージマージン分類器の理論から，この戦略は最適ではない．

【Perceptron】カーネルパーセプトロンは，連続して訓練事例を 1 つずつ取得し，予測したラベルが不正解であった場合に，その事例をサポートベクトルとしてモデルに追加する．これに着想を得て，集合 A をスキャンし，損失が非ゼロであった最初の事例 i を選択する．この戦略によって，厳密な順序付きデータを扱うオンラインアルゴリズムの精度がある程度判断出来る．

【Loss】サポートベクトルを選択するもっとも自然な戦略は，集合 A の中で目的関数を最小化する事例を選択することである．すなわち，損失のもっとも大きい事例を選択することである．

$$i = \underset{k \in A}{\operatorname{argmax}}\, \ell(y_k, f(\mathbf{x}_k; \boldsymbol{\alpha}))$$

ただし，この戦略はラベルノイズに敏感であるという欠点がある[5]．

【Active】能動学習は，アルゴリズムが特定の事例に対してのラベル情報を能動的に要求する方法であり，機械学習で近年注目されている．これは，本研究におけるサポートベクトル選択の問題と非常に類似している．能動学習でよく使われている戦略[15] として，集合 A の中で，現在の超平面にもっとも近い事例 i を選択する方法がある．

$$i = \underset{k \in A}{\operatorname{argmin}}\, |f(\mathbf{x}_k; \boldsymbol{\alpha})|$$

一般的に，$|f(\mathbf{x}_k)|$ は分類器の信頼度として見なされるので，この戦略は分類器の不確実性が一番高い事例を選択することを意味する．なお，この戦略は教師なしのため（上式に y_k は存在しない），ラベルノイズに対して頑健であるという特長がある．

【Loss-Probabilistic】教師ありの目的関数である式 (7) を最小化するためには，ラベル情報を利用することが重要である．この目的を達成し，かつラベルノイズに対する影響を抑えるため，クラスタ中心初期化のアルゴリズムである k-means++[3] から着想を得て，新しいサポートベクトル選択の戦略を提案する．この戦略の重要な点は，損失の最も大きい事例のラベル情報は誤りであるという可能性を考慮することである．もし，損失が最大の事例だけを常にサポートベクトルとして選択すれば，上述の Loss と同様に，精度に悪影響を及ぼす可能性がある．

この問題を解決するため，損失の高い事例ほど選択されやすいような確率的な選択を行う．その結果，損失が低い（最大ではない）事例も選択される余地が生まれる．たとえばヒンジ関数を使うと，現在の分類器が正しく予測した事例（$0 < y_k f(\mathbf{x}_k) < 1$）であってもサポートベクトルとして選ばれる可能性がある．具体的には，一つの事例 $\mathbf{x}_k$ の損失を次のように定義する．

$$\ell_k = \ell(y_k, f(\mathbf{x}_k; \boldsymbol{\alpha}))$$

この損失を用いて，多項分布のパラメタ $\boldsymbol{\theta}$ を定義する．

$$\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k, \ldots, \theta_{|A|})$$
$$\theta_k = \frac{\ell_k}{\sum_{k' \in A} \ell_{k'}}$$

上記の正規化は損失を確率として扱うためであり，損失が大きい事例ほど，サポートベクトルとして選択される確率が高くなる．最後に，事例 i を以下のように A からサンプリング

Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1

する．

$$i \sim Multinomial(\boldsymbol{\theta})$$

3.4 重みバックフィッティング

Algorithm 1 によって，式 (7) に従ったスパース性の制約を守る解 $\boldsymbol{\alpha}$ を見つけることができ，未知のデータに対しての分類が可能となる．しかし，Algorithm 1 が反復回数 B で終了すると仮定すれば，たかだか $B \times m$ 個の訓練事例しか考慮していないことになる．この数は，通常，全データセットの事例数 n のごく一部でしかない．サポートベクトルの数に制限を加えるとしても，良いアルゴリズムは訓練事例の大部分を考慮するべきである．

この問題を解決するため，事例とサポートベクトルの間のカーネルの出力値を類似度と見なし，重みバックフィッティングと呼ぶステップを導入する．これにより，サポートベクトル数を増やすことなく，モデルを改善することが可能となる．Algorithm 1 の各反復の時点において，サポートベクトル集合 S は不完全ではあるものの，2 章で述べたように式 (3) を内積形式で書き直すことができ，確率的劣勾配降下法で $\boldsymbol{\alpha}$ を改善できる．この過程を Algorithm 2 に示す．アルゴリズムが訓練事例を 1 つサンプリングし，その事例に対する予測を計算する．そして，予測が不正解であった場合，$\boldsymbol{\alpha}$ を更新する．この過程を T 回繰り返す．重みバックフィッティングという名前は orthogonal matching pursuit（OMP）[11] から由来する．OMP では二乗損失関数を用い，標準的な勾配降下法で重みバックフィッティングを行うのに対して，本研究では確率的劣勾配降下法を用い，任意の損失関数が利用できる．

3.5 計算コスト

提案手法における計算コストの大部分は，サポートベクトルを選択するためのカーネル計算に起因する．まず，$\boldsymbol{\alpha}$ に関する更新と正則化などの計算は，Shalev-Shwartz ら[14] が述べているように，$\boldsymbol{\alpha}$ のスケールと二乗ノルムを分割して保存するととても効果的に計算できるので，無視することができる．Algorithm 1 の一部であるサポートベクトル選択は，各反復ごとに $m|S|$ 回のカーネル計算を要するので，アルゴリズムが B 反復で終わると仮定すれば，合わせて $mB(B+1)/2$ 回のカーネル計算が必要となる．同様に，Algorithm 2 が $T|S|$ 回のカーネル計算を要するので，重みバックフィッティングは合わせて $TB(B+1)/2$ 回のカーネル計算が必要となる．比較として，一般的な SVM ソルバーは n^2 から n^3 の複雑性を持つ（$n \gg B$）．

Algorithm 2: Support vector coefficient back-fitting

Input: D, C, $\boldsymbol{\alpha}$, t
Parameters: λ, T
repeat
  Sample: $i = sample(C)$
  Compute prediction: $\hat{y}_i = f(\mathbf{x}_i; \boldsymbol{\alpha})$
  Compute update: $u_i = update(y_i, \hat{y}_i)$
  Set: $\eta_t = learning\_rate(t, \lambda)$
  Regularize: $\boldsymbol{\alpha} \leftarrow (1 - \eta_t \lambda)\boldsymbol{\alpha}$
  if $u_i \neq 0$ **then**
    Update SV coefficients: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} + \eta_t u_i \psi(\mathbf{x}_i)$
  end if
  $t \leftarrow t + 1$
until T instances have been sampled
Output: $\boldsymbol{\alpha}$

4. 評価実験

本章では，特に以下の点に注目して評価実験を行った．

- 異なるサポートベクトル選択方法の有効性評価．
- 重みバックフィッティングの効果．
- ラベルノイズに対する提案アルゴリズムの頑健性．
- 従来手法と提案手法との比較．

実験には，USPS，Adult，MNIST データセットを用いた．これらのデータセットは，線形カーネルより非線形カーネルの方が優れた精度を示しているので，カーネル分類の文献でよく使われている．USPS と MNIST は手書き数字認識の多クラスデータセットであり，ノイズが少ない．Adult は 2 値分類データセット（年棒が 50 万ドルを超えているかどうか）であり，非常にノイズが多い．前者の USPS と MNIST を用いた実験については，従来の研究でよく行われているように，それぞれ 0 番目のクラスかそれ以外（USPS0），8 番目のクラスかそれ以外（MNIST8）の 2 値分類問題として扱った．さらに，MNIST に人工的に

Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1

表 1 Datasets

Name	Train	Test	Feat	#SV
MNIST8	60,000	10,000	780	760
MNIST8+N	60,000	10,000	780	11,642
Adult	32,561	16,281	123	6,650
USPS0	7,291	2,007	256	228

ラベルノイズを 10%加えたデータ（MNIST8+N）を用いた実験も行った．データセットの概要を表 1 に示す．なお，#SV はサポートベクトル数を制限しない（無限の予算を用いた）カーネル pegasos[14] を適用して得られたサポートベクトル数である．

カーネルとしては，$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-0.01||\mathbf{x}_i - \mathbf{x}_j||^2)$ を USPS0，MNIST8，MNIST8+N で使い，$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (1 + \langle \mathbf{x}_i, \mathbf{x}_j \rangle)^4$ を Adult で用いた．全ての実験に関して，サンプル A の大きさ m を 60 とし，学習率を pegasos と同じ $\eta_t = 1/(\lambda t)$ とした．正則化のハイパーパラメータである λ は，訓練データの 10%を検証データとして決定した．特に明記しない限り，すべての実験でヒンジ損失を利用した．本節で使われているすべてのアルゴリズムは確率的であるため，実験の度に異なる結果が得られる．そのため，データセットをシャッフルし，各実験を 30 回繰り返して平均を求めた．計算機環境としては，Intel(R) Core(TM) i5 CPU750 @ 2.67GHz（4Gb RAM）を利用し，コードは C++で実装した．

【サポートベクトル選択戦略の比較】この実験では，重みバックフィッティングを使用せずに Algorithm 1 を実行し，3.3 節で述べたサポートベクトル選択の戦略を比較した．図 1 に，サポートベクトル数 B を変化させたときの結果を示す．ラベルノイズを含んでいる Adult と MNIST8+N データセットに対して，*Active* がもっとも良い精度を示している．これは予想された通りの結果であり，*Active* がラベル情報に依存しないことによる．ただし，クリーンなデータセットに対しては，あまり良い精度を示さなかった．対照的に，*Loss* はクリーンなデータセットに対して最良の精度を示しているものの，ラベルノイズを含んでいるデータセットに対しては非常に精度が低い．一方，*Loss-Probabilistic* はクリーンなデータセットに対して *Loss* と同等の精度を保ちつつ，ラベルノイズを含んでいるデータセットに対しても *Loss* 以上の精度を示した．また，*Perceptron* と *Random* は，どのデータセットに対しても不適であった．

【重みバックフィッティングの影響】この実験では，Algorithm 2 で述べた重みバックフィッティングも行った上で，サポートベクトル選択の戦略を比較した．結果を図 2 に示す．累計で n 個程度の訓練事例が考慮されるように，つまりデータセットと同程度の事例を考慮するように，反復回数 T の値を n/B とした．この場合，重みバックフィッティングの計算コストは $n(B+1)/2$ 回のカーネル計算となる．結果から，2 つの傾向が観察できる．1 つは，重みバックフィッティングを行わなかった場合と比較すると，重みバックフィッティングがサポートベクトル選択戦略間のギャップ（違い）を縮めることである．もう 1 つは，特にデータセットがラベルノイズを含んでいる場合やサポートベクトル数が小さい場合，重みバックフィッティングが精度を向上させることである．

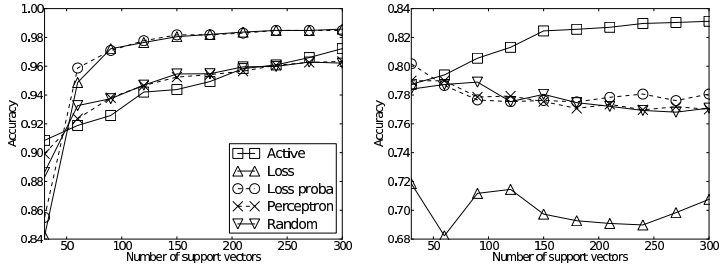
【従来手法との比較】この実験では，budget perceptron[6] および LaSVM[5] を提案手法と比較した．budget perceptron は，定義された最大のサポートベクトル数（予算）に達すると，既存のサポートベクトルをランダムで 1 つ選択・除去する．実験を行うため，データセットをシャッフルし，訓練事例を 1 つずつ利用した．LaSVM は，新しいサポートベクトルと既存のサポートベクトルに関する SMO[12] ステップ，および不必要なサポートベクトルを除去するステップを定義されたサポートベクトル数に達するまで繰り返す．LaSVM において，サポートベクトル選択戦略は *Active* と *Gradient* を使用した．ヒンジ損失を使った際，後者は本質的に *Loss* と同じである．結果を表 3 に示す．

USPS0 以外では，budget perceptron と LaSVM より，提案アルゴリズムが高い精度を示した．比較のため，サポートベクトル数を制限しないカーネル pegasos を用いて実験を行ったところ，USPS0，Adult，MNIST8，MNIST8+N に対して，それぞれの精度が 98.05%，80.87%，99.37%，89.71%であった．クリーンなデータセットの場合，カーネル pegasos の精度が最適であったものの，ラベルノイズが含まれているデータセットの場合，提案アルゴリズムと LaSVM のようなスパースカーネル分類器の方が良い精度を示している．これは，スパース性の制約は正則化と見なすことができ，過学習を制限して汎化能力を上げる効果があることによる．

5. 関連研究と議論

訓練データの一部を参考事例（reference training instances）とし，それらの事例の集合との類似度を素性として使用する概念は非常に有効である．実際に SVM は，そのような参考事例，すなわちサポートベクトルのスパースな集合を理論的な教師付きの方法で選択する枠組を与えている．Balcan ら[4] は，類似度関数を用いた学習理論を発展させ，訓練セットから参考事例を十分サンプリングし， 勾配降下法でそれらの事例の重みを最適化すれば，良い分類器を学習できることを証明した．本研究による実験的な観測（図 2）によると，たしかに重みのバックフィッティングを行うことで精度は向上し，結果的にサポートベクトル

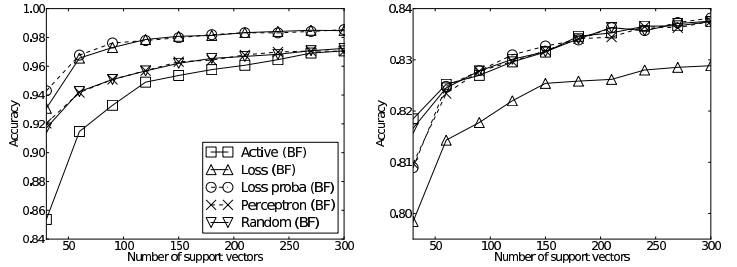
Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1



(a) USPS0 (b) Adult

(c) MNIST8 (d) MNIST8+N

図 1 Accuracy comparison of support vector selection strategies (without coefficient back-fitting)



(a) USPS0 (b) Adult

(c) MNIST8 (d) MNIST8+N

図 2 Accuracy comparison of support vector selection strategies (with coefficient back-fitting)

の選択方法の違いによる精度の差は小さくなった．しかし，同じ実験結果から，ランダム選択よりも良い選択方法が存在することは疑う余地がない．

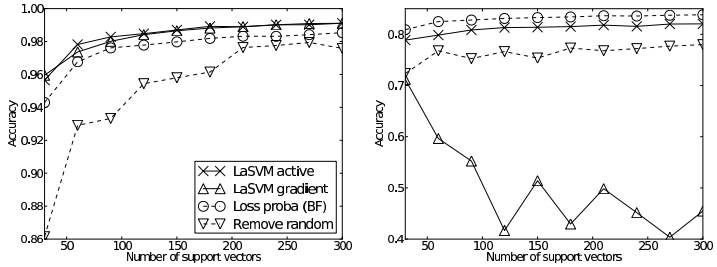
重みベクトルにスパース性の制約を与えることは，特徴選択を含んだ分類器の学習と見なすことができる．そのような従来研究の例としては，Schölkopf ら[13] と Weston ら[18] の研究がある．これらはそれぞれ，L_1 ノルムと L_0 擬似ノルムの近似を用いたバッチアルゴリズムである．サポートベクトル数の上限（予算）を考慮した分類は，オンライン学習の分野でも盛んに研究されてきている．そこでは主に，除去すべきサポートベクトルを選択する方法についての研究がなされている[6),8),9)]．

本研究と関係が一番深い既存手法は，おそらく kernel matching pursuit（KMP）[16] と LaSVM[5] である．1 節で述べたとおり，提案アルゴリズムは KMP と類似した点がいくつかある．ただし，KMP は全データセットを用いた積算損失によってサポートベクトルを選択するので，計算コストが非常に高いという問題がある．提案手法は，LaSVM と同様に，ランダム探索を使うことで効果的にサポートベクトルの選択が行える．しかし，LaSVM は SVM の双対問題を解き，提案アルゴリズムは主問題を解く．また提案アルゴリズムでは，SVM の損失関数であるヒンジ関数だけでなく，任意の損失関数が利用できる．

6. おわりに

本研究では，スパースカーネル分類器を学習するため，主問題による統合アルゴリズムを提案した．提案アルゴリズムは有効に訓練データを利用でき，任意の劣微分可能な凸損失関数が利用できる．また，ラベルノイズに頑健であり，実装し易いという特長を持つ．今後の課題として，まず，不必要となったサポートベクトルを除去するためのステップを研究する予定である．また，学習の際のアルゴリズムの終了条件をさらに検討することを計画している．他の拡張としては，構造化予測とマルチカーネル学習への適応がある．さらに，

Vol.2011-MPS-86 No.2
Vol.2011-BIO-27 No.2
2011/12/1



(a) USPS0 (b) Adult

(c) MNIST8 (d) MNIST8+N

図 3 Accuracy comparison with existing algorithms

ベクトルデータと非ベクトルデータ（例えば時系列データ）について，大規模なデータセットを用いて実験を行う予定である．

参 考 文 献

1) Aizerman, A., Braverman, E.M. and Rozoner, L.I.: Theoretical foundations of the potential function method in pattern recognition learning, *Automation and Remote Control*, Vol.25, pp.821–837 (1964).
2) Amaldi, E. and Kann, V.: On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems, *Theoretical Computer Science*, Vol.209, No.1-2, pp.237 – 260 (1998).
3) Arthur, D. and Vassilvitskii, S.: k-means++: The advantages of careful seeding, *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, Society for Industrial and Applied Mathematics, pp.1027–1035 (2007).
4) Balcan, M. and Blum, A.: On a theory of learning with similarity functions, *Proceedings of the 23rd international conference on Machine learning*, ACM, pp.73–80 (2006).
5) Bordes, A., Ertekin, S., Weston, J. and Bottou, L.: Fast Kernel Classifiers with Online and Active Learning, *Journal of Machine Learning Research*, Vol. 6, pp. 1579–1619 (2005).
6) Cavallanti, G., Cesa-Bianchi, N. and Gentile, C.: Tracking the best hyperplane with a simple budget Perceptron, *Machine Learning*, Vol.69, pp.143–167 (2007).
7) Chapelle, O.: Training a support vector machine in the primal, *Neural Computation*, Vol.19, No.5, pp.1155–1178 (2007).
8) Crammer, K., Kandola, J.S. and Singer, Y.: Online Classification on a Budget, *Proceedings of Neural Information Processing Systems (NIPS)* (2003).
9) Dekel, O., Shalev-Shwartz, S. and Singer, Y.: The Forgetron: A kernel-based perceptron on a fixed budget, *Proceedings of Neural Information Processing Systems (NIPS)* (2005).
10) Freund, Y. and Schapire, R.E.: Large Margin Classification Using the Perceptron Algorithm, *Machine Learning*, Vol.37, pp.277–296 (1999).
11) Pati, Y. C., Rezaiifar, R. and Krishnaprasad, P. S.: Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition, *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, pp.40–44 vol.1 (1993).
12) Platt, J. C.: *Fast training of support vector machines using sequential minimal optimization*, pp.185–208, MIT Press (1999).
13) Schölkopf, B. and Smola, A.J.: *Learning with Kernels*, MIT Press (2002).
14) Shalev-Shwartz, S., Singer, Y., Srebro, N. and Cotter, A.: Pegasos: primal estimated sub-gradient solver for SVM, *Mathematical Programming*, pp.1–28 (2010).
15) Tong, S. and Koller, D.: Support vector machine active learning with applications to text classification, *Journal of Machine Learning Research*, Vol.2, pp.45–66 (2002).
16) Vincent, P. and Bengio, Y.: Kernel Matching Pursuit, *Machine Learning*, Vol.48, pp.165–187 (2002).
17) Wang, Z., Crammer, K. and Vucetic, S.: Multi-Class Pegasos on a Budget, *Proceedings of the twenty-seventh international conference on Machine learning* (2010).
18) Weston, J., Elisseeff, A., Schölkopf, B. and Tipping, M.: Use of the zero norm with linear models and kernel methods, *Journal of Machine Learning Research*, Vol.3, pp.1439–1461 (2003).
19) Zhang, T.: Solving large scale linear prediction problems using stochastic gradient descent algorithms, *Proceedings of the twenty-first international conference on Machine learning*, ACM, p.116 (2004).