顔検出における識別コストの削減および検出精度向上のための特徴分割 SVM

小野　友己†　　黒田　忠広††

† 慶應義塾大学理工学研究科
〒 223–8522 神奈川県横浜市港北区日吉 3–14–1
†† 慶應義塾大学理工学部電子工学科
〒 223–8522 神奈川県横浜市港北区日吉 3–14–1
E-mail: †ono@kuro.elec.keio.ac.jp, ††kuroda@elec.keio.ac.jp

あらまし　顔検出は近年パターン認識の分野で注目を集めている技術であり、様々な手法が提案されてきた。しかし、いずれの手法も誤検出率を低くするにつれて検出率も低下する。SVM をカスケード最終段に接続することで検出率の低下を抑えることが可能だが、識別にかかる演算時間および識別器に必要なデータサイズ、すなわち識別コストが大幅に増大する。そこで、本稿では識別コストを低減した SVM として特徴分割 SVM を提案する。提案手法により RBF kernel SVM と比べて 54 倍高速で、1/146 倍のデータサイズの SVM を実現した。さらに提案手法をカスケード最終段に接続することでデータセットにおいて検出率 7.7%の向上を達成した。

キーワード　特徴分割 SVM, 識別コスト, 1 次識別器, 2 次識別器, HOG, Soft Cascade, 顔検出

1. はじめに

顔検出は近年パターン認識の分野で注目を集めている技術であり、デジタルカメラのオートフォーカスや、顔認証、HCI など様々なアプリケーションに応用されている。こうした技術を実現する顔検出アルゴリズムは数多く提案されてきた。Viola ら [1] は Haar-like 特徴を用いた弱識別器を AdaBoost により選択し、強識別器を構成するアルゴリズムを提案した。さらに強識別器をカスケードに接続することで、リアルタイム動作を実現した。この手法は従来手法に比べて精度、速度の両面において優れており、以降多くの研究の基盤となっている。Wu ら [2] は Look-up-table 型の弱識別器を Real AdaBoost により選択するアルゴリズムを提案し、より精細な検出を可能にした。また、Huang ら [3] は木構造に識別器を構築する Vector Boosting を提案し、多方向の顔検出を可能にした。

一方で、識別にかかる演算時間および識別器に必要なデータサイズ、すなわち識別コストを小さくするためのアルゴリズムも数多く提案されている。Bourdev ら [4] はカスケード 1 段に 1 特徴を割り当てる Soft Cascade を提案した。これにより [1] よりも少ない特徴数で識別を行うことができ、識別の高速化およびデータサイズの削減を実現した。また、花井ら [5] は Look-up-table 型弱識別器のビン幅を適応的にに変化させることで少ないビン数での識別を可能にした。

いずれの手法にも検出率と誤検出率のトレードオフの関係が存在する。すなわち、誤検出率を低くすると検出率も低くなり、検出率を高くすると誤検出率もまた高くなる。既存の識別器は自身の閾値と顔である確信度との比較で識別を行っているため、このトレードオフは必然的に生じてしまう。しかし、閾値調整による検出率の変化率を小さくすることは可能である。図 1 に示したように、従来の識別器は誤検出を低くするほど、検出率は急激に低下する傾向にある。したがって、このときの低下率を小さくすることで、誤検出率を低くしても検出率は高く保つことが可能である。これは既存の識別器の最終段に新たな識別器を接続することで実現できる。既存の識別器を 1 次識別器、新たに接続する識別器を 2 次識別器とすると、1 次識別器の閾値を低く設定し、検出率、誤検出率を高くした状態から 2 次識別器で非顔のみを選択的に削除する。1 次識別器のみでもカスケード段数を増やすことで誤検出率を小さくすることは可能だが、深

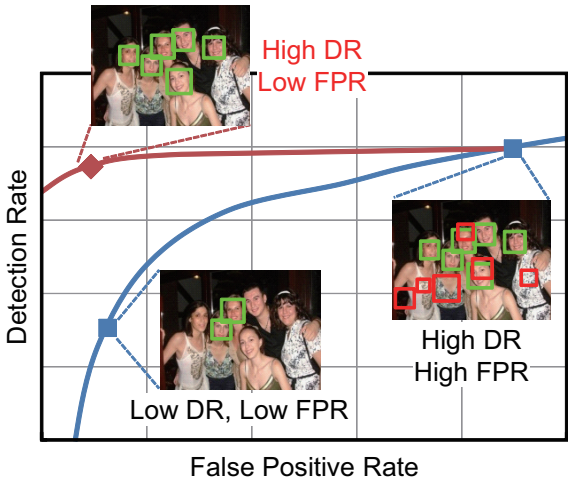

図 1　ROC の変化率の違いと識別精度

いステージになるにつれてその削減率は飽和してくる。これは、各ステージが特徴量のパラメータこそ異なるものの、特徴の種類および識別器そのものは本質的には等価であることに起因すると考えられる。そこで、1 次識別器とは異なる特徴量および識別関数をもつ 2 次識別器を最終段に接続することで効率的な誤検出削減が可能になる。

2 次識別器として、Zhang ら [6] は PCA による特徴量を用いることで局所的、大域的両方の面から顔を検出することを可能にした。また、山下ら [7] は検出窓を Gabor ウェーブレット変換し、得られた特徴量に対してサポートベクタマシーン (SVM) で識別を行うことで 1pt の検出率の低下で 87.7%の誤検出数の削減に成功した。

ここで用いられている SVM とは、Vapnik らによって提案された 2 値分類問題に対する識別器である。学習サンプルのクラス間のマージンを最大にする境界を識別に用いることで高い汎化性能を保ち、顔検出に限らず人体検出や一般物体認識など幅広い応用がなされている。SVM の識別関数は、

$$f(\mathbf{x}) = \mathrm{sign}\left(\sum_{i=1}^{N_{SV}} w_i K(\mathbf{x}, \mathbf{z}_i) - \theta\right) \qquad (1)$$

で表される。ただし、w_i は学習で決定されるパラメータであり、$\mathbf{z}_i$ はサポートベクタと呼ばれる識別境界の最近傍にある学習サンプルを表す。また、N_{SV} はサポートベクタ数とする。$K(\mathbf{x}, \mathbf{z})$ はカーネル関数と呼ばれ、一般的には式 (2) の Polynomial カーネルや、式 (3) の RBF カーネルなどが用いられる。

$$K_{\mathrm{Poly}}(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z})^d \qquad (2)$$

$$K_{\mathrm{RBF}}(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2) \qquad (3)$$

SVM の欠点として、識別コストが他の識別器に比べて膨大であることが挙げられる。今、SVM で用いる特徴量の次元数を D としたとき、SVM にかかる識別コストは $\mathcal{O}(N_{SV}D)$ で与えられる。

そこで、本稿では高速かつ省メモリな SVM として、特徴分割 SVM を提案し、それをカスケード識別器最終段に接続することで、既存の識別器の精度向上を図る。

以降では次のようにまとめる。2 章では、従来の SVM の識別コスト削減手法について述べる。3 章では提案手法である特徴分割 SVM について述べる。4 章では特徴分割 SVM の分割手法に関して議論する。5 章では識別精度、演算時間、データサイズを各 SVM に関して比較し、6 章において結論とする。

2. SVM の識別コスト削減手法

SVM の識別コスト削減手法として特徴量次元数の削減や、サポートベクタの削減が挙げられる。特徴量次元数削減は、Forward Feature Selection などの特徴選択やサンプリングなどで行うことができる。山下ら [7] はサンプリング方法として Retinal サンプリングを用いることで、Gabor 特徴を 136 次元まで削減している。一方、サポートベクタ削減手法として、Burges ら [8] は Reduced Set SVM(RSVM) を提案した。RSVM では、元のサポートベクタからより少数の新たなデータセット (reduced set) を構成する。識別関数は、reduced set を $\mathbf{z}'$ としたとき、

$$f_{\mathrm{RSVM}}(\mathbf{x}) = \mathrm{sign}\left(\sum_{i=1}^{N_r} w_i K(\mathbf{x}, \mathbf{z}'_i) - \theta\right) \qquad (4)$$

で表される。ただし、N_r を reduces set 数とする。RSVM による識別コストの削減効果は、N_r/N_{SV} である。また、Ben-Hur ら [9] はガウシアンカーネル SVM においてサポートベクタのクラスタリングを行うことで識別コストを削減した。Maji ら [10] はカーネルにヒストグラムインタセクションを用いて、サポートベクタ数に依らない一定時間での識別を可能にした。

Heisele ら [11] は、入力画像サイズおよびカーネルの異なる SVM をカスケードに接続することで高精度かつ高速な識別を可能にした。

3. 特徴分割 SVM

3.1 Approximate RBF Kernel SVM

ここでは、提案する特徴分割 SVM の基礎となる Approximate RBF Kernel SVM(Approx–SVM) について述べる。

Approx–SVM は Cao ら [12] により提案された手法であり、SVM のカーネル関数である RBF カーネルに対して 2 次のテイラー展開を適用する。すなわち、

$$\begin{aligned} K_{\mathrm{RBF}}(\mathbf{x}, \mathbf{z}) &= \exp(-2\gamma)\exp(2\gamma \mathbf{x}^T \mathbf{z}) \\ &\approx a + b(2\gamma \mathbf{x}^T \mathbf{z}) + c(2\gamma \mathbf{x}^T \mathbf{z})^2 \end{aligned} \qquad (5)$$

とすると、式 (1) は次のように変形可能である。

$$f_{\mathrm{Approx}}(\mathbf{x}) = \mathrm{sign}\left(\boldsymbol{\beta}^T \mathbf{x} + \mathbf{x}^T \boldsymbol{\Omega} \mathbf{x} - \theta\right) \qquad (6)$$

ただし、

$$\beta_j = \sum_{i=1}^{N_{SV}} 2\gamma b w_i z_{ij} \qquad (7)$$

$$\omega_{jk} = \sum_{i=1}^{N_{SV}} 4\gamma^2 c w_i z_{ij} z_{ik} \qquad (8)$$

とする。

Approx–SVM の識別コストは、$\boldsymbol{\Omega}$ が対称行列であるから、$\mathcal{O}(D(D+3)/2)$ となり、SVM と比べて $(D+3)/2N_{SV}$ の削減が可能となる。

Approx–SVM の利点は特徴次元数やサポートベクタ数の削減を行っていない点にある。つまり、近似が十分に妥当であれば識別精度を落とさずに識別コストの削減可能である。

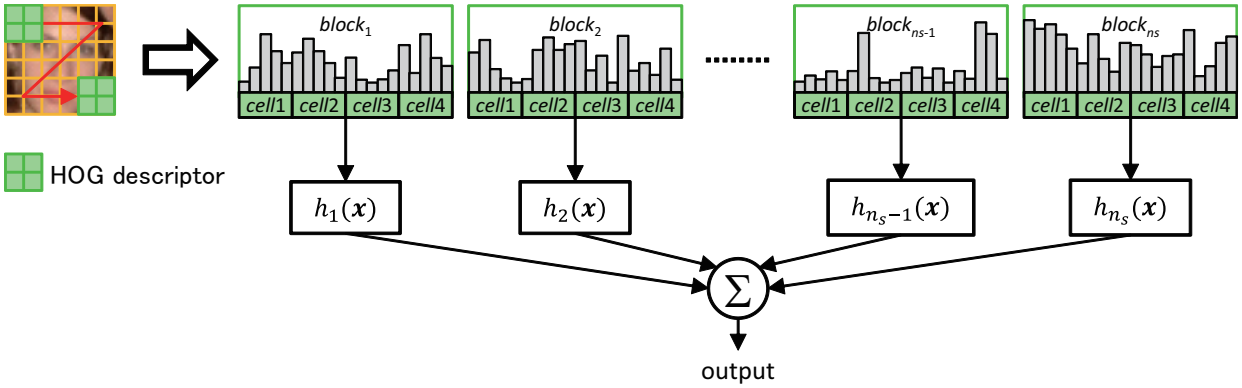

図 2　ブロック分割による FDSVM

input:
- 学習サンプル $\mathcal{S} = \{(\mathbf{X}_1, y_1), \ldots, (\mathbf{X}_m, y_m)\}$
- 学習サンプル数 m
- 特徴分割数 n

for $i \leftarrow 1$ **to** n **do**
（1） $\mathbf{s}_i = \{(\mathbf{x}_{i,1}, y_1), \ldots, (\mathbf{x}_{i,m}, y_m)\}$
（2） $\mathbf{s}_i$ から RBF kernel SVM h'_i を学習
（3） h'_i を Approx–SVM h_i に変換
（4） $\alpha_i = \log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$
ただし、ε_i は h_i の訓練誤差とする
end

output:
- $f_{\mathrm{FD}}(\mathbf{X}) = \mathrm{sign}\left(\sum_{i=1}^{n} \alpha_i h_i(\mathbf{x}_i) - \theta\right)$

アルゴリズム 1: 特徴分割 SVM の学習

3.2　特徴分割 SVM

Approx–SVM の欠点として、識別に高次元特徴を用いた場合、依然として識別コストは膨大であるということが挙げられる。これは、識別コストが特徴量次元数の 2 乗に比例することに起因する。そこで、本稿では Approx–SVM を元に、高次元特徴を利用しつつも低識別コストな SVM として、特徴分割 SVM(Feature Division SVM: FDSVM) を提案する。

FDSVM では前処理として特徴量を n 個のセグメントに分割する。今、検出窓より抽出される D 次元特徴量を $\mathbf{X} = \{f_1, f_2, \ldots, f_D\}$ とすると、特徴量は $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ に分割される。ただし、$\mathbf{x}_i = \left\{f_{\frac{D}{n}(i-1)+1}, \ldots, f_{\frac{D}{n}i}\right\}$ とする。アルゴリズム 1 により、各セグメントの特徴量から Approx–SVM を生成し、重みを割り当てる。最終的な識別結果は、各 Approx–SVM の識別結果の重みつき加算により行われる。

FDSVM は [1] における強識別器と同じ構造をとっている。特徴量を分割したことで各 $h(\mathbf{x})$ は弱識別器として低精度の識別を行うが、それらを組み合わせることで高い精度の識別を可能にしている。

次に、FDSVM の識別コストについて議論する。前述したように、$h(\mathbf{x})$ には D/n 次元の特徴量が入力される。仮に $h(\mathbf{x})$ に、SVM を用いると、FDSVM 全体として必要な識別コストは $\mathcal{O}(N_{SV}\frac{D}{n} \cdot n) = \mathcal{O}(N_{SV}D)$ であり、識別コストは SVM と同一である。しかし、Approx–SVM を用いることで、FDSVM の識別コストは $\mathcal{O}\left(\frac{\frac{D}{n}\left(\frac{D}{n}+3\right)}{2} \cdot n\right) = \mathcal{O}\left(\frac{D\left(\frac{D}{n}+3\right)}{2}\right)$ となる。近似的には $D/n \gg 1$ であるとき、識別コストは Approx–SVM と比べて $1/n$ になる。また、SVM と比べると、$\frac{1}{n} \cdot \frac{D}{2N_{SV}}$ にまで削減可能である。

4.　特徴量の分割手法

4.1　Histograms of Oriented Gradients

FDSVM で用いる特徴量として本稿では Histograms of Oriented Gradients(HOG) [13] を採用した。HOG では、検出窓に対して設定された n_s 個のサンプリング点を中心とする局所領域 (ブロック) から特徴量が抽出される。各ブロックはさらに $n_c \times n_c$ 個のセルと呼ばれる小領域に細分化され、各セルごとに輝度勾配ヒストグラムが生成される。得られたヒストグラムはブロックごとに正規化され、最終的に、すべてのヒストグラムを連結して特徴ベクトルが形成される。

今、ヒストグラムのビン数を n_b とすると、HOG の次元数は

$$D_{\mathrm{HOG}} = n_c^2 n_b n_s \tag{9}$$

で与えられる。

4.2　HOG の分割単位

HOG は、ヒストグラムという意味のあるまとまりで構成されているため、それを無視して分割を行うことはできない。さらに、各サンプリング点ごとに独立して抽出されるため、分割の最小単位はブロックとなる。図 2 に、ブロック分割を行ったときの FDSVM を示した。すなわち、FDSVM は n_s 個の Approx–SVM から形成され、識別コストは、

$$c = \mathcal{O}\left(\frac{n_c^2 n_b n_s (n_c^2 n_b + 3)}{2}\right) \tag{10}$$

で表される。

4.3 AdaBoost 選択

HOG のサンプリング点の組み合わせは無数存在するため、各組み合わせごとに比較し、最適解を求めることは困難である。そのため、一般的に HOG はグリッド状に配置されたサンプリング点より抽出される。一方、FDSVM では、サンプリング点ごとに独立して識別を行うため、サンプリング点の逐次的な選択が可能となる。そこで、本稿ではサンプリング選択手法として AdaBoost を採用した。学習において、まず、弱識別器候補の作成を行う。検出窓サイズを $w \times w$ pixel とすると、位置 (x, y) をサンプリング点として学習サンプルより HOG を抽出し、Approx–SVM $h_{x,y}$ を生成する。ただし、$0 \leqq x, y < w$ とする。次に、得られた $w \times w$ 個の弱識別器候補から AdaBoost による選択を行う。各ステップごとに、$h_{x,y}$ の閾値を再調整し、訓練誤差を最小にする弱識別器を選択した後、学習サンプルの重みを更新する。

AdaBoost 選択を用いることによって、サンプリング点を最適化するだけでなく、サンプリング点数の調整も可能になる。これにより冗長な点を排除し、識別精度と識別コストの最適化が可能となる。

5. 実　　験

提案する顔検出器では、1 次識別器として、350 段からなる Soft Cascade 識別器を用いた。各段では、Real AdaBoost により決定された Look-up-table 型弱識別器が識別を行う。

Soft Cascade 全段を通過した検出窓は続く 2 次識別器で識別が行われる。2 次識別器には RBF kernel SVM、Approx–SVM、RSVM、そして FDSVM を用いて比較を行った。なお、SVM の実装には LIBSVM [14] を用いた。

特徴量は、1 次識別器には Haar-like 特徴を、2 次識別器には $n_c = 2$、$n_b = 9$ の HOG を利用している。1 セルの大きさを 4×4pixel とし、サンプリング点は AdaBoost FDSVM 以外は検出窓に対して 5×5 のグリッド状に配置した。2 次識別器を学習するにあたり、顔サンプルは 1 次識別器と同じものを用いたが、非顔サンプルに関しては、1 次識別器において誤検出として識別された非顔サンプルのみを用いた。

まず、FDSVM においてブロック分割と AdaBoost 選択の訓練誤差の比較を図 3 に示す。AdaBoost においてサンプリング数が増えるにつれて訓練誤差は減少し、$n_s \geqq 13$ でブロック分割の訓練誤差を下回った。これより、AdaBoost FDSVM のサンプリング数を 13 と定めた。

次にそれぞれの識別器のデータサイズおよび 1 検出窓あたりの識別時間を表 1 にまとめた。ただし、1 データあたりのビット幅は 32bit とする。

表 1 より、識別器データサイズは、RBF kernel SVM

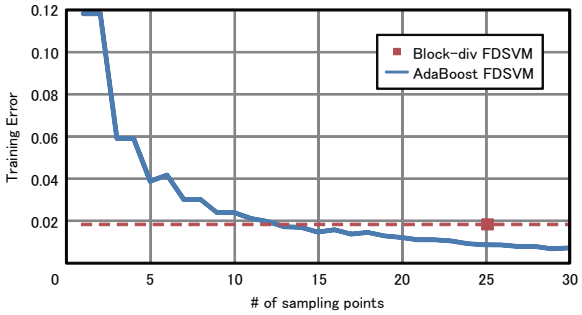

図 3　AdaBoost FDSVM の訓練誤差

表 1　データサイズおよび識別時間

classifier	data size (kB)	time (ms/window)
Soft Cascade(350 stages)	5.6	0.04
RBF kernel SVM	5346	10.43
Approx–SVM	1625	1.98
RSVM($N_r = 200$)	720	1.66
RSVM($N_r = 20$)	72	0.43
Block-div FDSVM	70.2	0.38
AdaBoost FDSVM	36.5	0.19

と比較したとき、ブロック分割 FDSVM では約 1/76 倍に、AdaBoost FDSVM では約 1/146 倍に減少している。識別速度も、それぞれ約 28 倍、54 倍で実行可能となった。

さらに、2 次識別器接続による 1 画像 (320×240 pixel) あたりの識別時間の増加を図 4 に示した。RBF kernel SVM では、識別時間が 1 次識別器のみの場合に比べて約 2 倍に増加するのに対し、ブロック分割 FDSVM では、1.04 倍、AdaBoost FDSVM では 1.02 倍と、従来とほぼ同速度での識別が可能となった。

最後に、識別精度を図 5 に示した。評価は web より収集した正面顔 1154 個を含む 600 枚の画像セットにより行った。ブロック分割 FDSVM および AdaBoost FDSVM は他手法に比べて高い検出率を維持することができた。1 次識別器のみの場合、検出率 86.2%のとき、139 個の誤検出が生じていたものを、AdaBoost FDSVM により、検出率 85.4%で誤検出数 10 個まで削減することができた。すなわち、検出率 0.8pt の低下で 93%の誤検出を削減している。また、このとき 1 次識別器のみの場合に比べて検出率は 7.7pt 向上している。

6. 結　　論

本稿では特徴分割 SVM を提案し、高速かつ省メモリな SVM を実現した。HOG 特徴においてブロック分割と、AdaBoost 選択を提案し、RBF kernel SVM に比べてブロック分割では識別速度を約 28 倍、AdaBoost 選択では 54 倍向上した。また、データサイズにおいても、ブロック分割では約 1/76 倍、AdaBoost 選択では約 1/146 倍に削減することができた。さらに特徴分割 SVM を Soft Cascade 識別器の最終段に接続することで検出速度を犠牲にすることなく検出率を 7.7pt 向上させた。