Exemplar-RF による物体検出の高速化

山内 悠嗣 † 佐藤 慎也 † 黒川 貴都 † 山下 隆義 † 藤吉 弘亘 †
† 中部大学
E-mail: yuu@vision.cs.chubu.ac.jp, hf@cs.chubu.ac.jp

Abstract

事例ベースの学習法である Exemplar-SVM は，見えの変化に対して頑健に識別が可能である．識別時にテストサンプルと最も類似する学習用ポジティブサンプルを求めることで，学習サンプルに付与した物体の向きや姿勢等の属性情報をテストサンプルに転移させることができる．Exemplar-SVM は 1 つの学習用ポジティブサンプルに対して 1 つの識別器を学習する必要があり，識別時には全ての識別器のスコアを計算することになるため多大な計算を要するという問題がある．そこで，本研究では識別の高速化として，Exemplar-Random Forests を提案する．Exemplar-Random Forests は，Exemplar-SVM による識別処理に木構造を導入することで高速化を実現する．また，識別時に近似内積演算と早期分岐を導入することで，さらなる高速化を図る．評価実験の結果，提案手法は Exemplar-SVM より約 59.7 倍高速であることを確認した．

1 はじめに

顔検出や人検出に代表される物体検出は精力的に研究され，より多くのクラスに対応するために，クラス内の見えの変化が多様な物体の検出の実現が期待されている．検出対象クラスの多様な見えを吸収する手法として，パーツベースの物体検出法である Deformable Part Model(DPM)[1] が提案されている．DPM は，物体の姿勢変化による見えの違いを吸収するようなアプローチである．DPM で用いられている SVM[2] や Boosring[3] 等の 2 クラス識別器に基づく手法では，見えの変化が豊富なカテゴリを 1 つのカテゴリ識別器としてモデル化する．しかし，個々の見えの大きな変化に対応できない場合，識別器の汎化性能が低下する．

この問題を解決する手法として Exemplar-SVM[4] が提案されている．Exemplar-SVM は，ポジティブサンプル集合のある 1 つのサンプル[1]とネガティブサンプル集合から 1 つの識別器を学習する．この処理を全てのポジティブサンプルに対して行い，ポジティブサンプル数 N と同数の識別器を学習する．識別時には，入力サンプルに対して N 個の識別関数のスコアを算出し，最も高いスコアをしきい値処理することでクラスを判定する．

多様な見えを持つポジティブサンプルを扱う場合，特徴空間においてポジティブサンプルは広がりを持つ分布となり，ポジティブサンプルとネガティブサンプルは複雑に交じり合う．このようなサンプルを用いて Boosting や SVM 等でカテゴリ識別器を学習すると過学習を引き起こし，汎化性能が低下する．一方，Exemplar-SVM では，ポジティブサンプル集合のある一つのサンプルとネガティブサンプルから識別器を学習するため，クラスの異なるサンプルが交じり合う問題が発生しないために過学習することはない．Exemplar-SVM は多様な見えを持つ物体の検出に有効であり，三次元物体認識等にも利用されている [5]．また，識別時には類似した学習サンプルを探索するため，学習サンプルに付与した属性情報を転移することも可能である．

Exemplar-SVM は，上記のような利点がある一方で以下に示す 2 つの問題がある．1 つ目は，識別時に学習用ポジティブサンプルと同数の識別器に対する演算が必要となる点である．Exemplar-SVM の識別関数は，SVM と同様に特徴ベクトルと重みベクトルの内積により構成されるため，計算量が非常に大きくなるという問題がある．例えば，図 1(a) に示すように N 個の学習用ポジティブサンプルから識別器を学習した場合，識別時には N 回の内積を計算することになる．2 つ目は，各識別器のスコアのキャリブレーションである．識別時には，全ての識別器から計算されたスコアを比較する．Exemplar-SVM は，個々の識別器を独立に学習するため，スコアの範囲が識別器により異なる．これを解決するために，識別器のスコアの範囲を調整するキャリブレーションを必要とする．キャリブレーションでは，検証サンプルを用いたパラメータ調整が必要であり，識別時にも計算量が増加する．

そこで，本研究では上記の 2 つの問題を解決する手法として，Exemplar-Random Forests(E-RF) を提案する．Exemplar-Random Forests では，Exemplar-SVM の識別処理において木構造の Random Forests[6] を導入することで高速化を実現する．これにより，全ての

[1] 事例のことをサンプルとも呼ぶ．

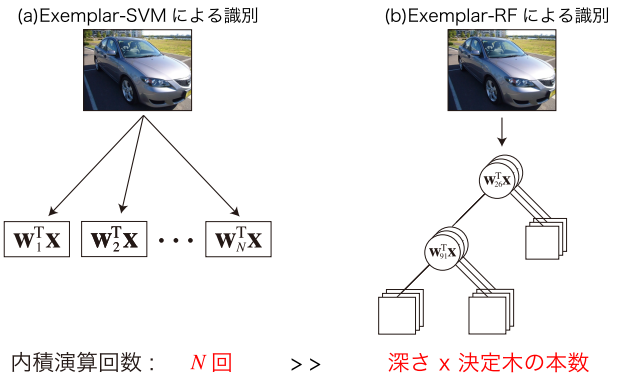

図 1 Exemplar-SVM と Exemplar-Random Forests における内積演算回数の比較．

識別器に対する内積計算をする必要がなくなる．例えば，図 1(b) に示す Random Forests を導入した提案手法では，決定木の深さ × 決定木の数の内積計算となる．Exemplar-SVM で取り扱う問題では，学習用ポジティブサンプルに数千枚を用いるため，木構造の Random Forests を導入することで識別の高速化が期待できる．さらに，本研究では識別時に近似内積計算 [7] と早期分岐を導入することでさらなる高速化を図る．また，木構造を導入した提案手法は，Exemplar-SVM の全識別器のスコアを一度に比較する必要がないため，スコアのキャリブレーションが不要となる．

1.1 関連研究

本研究と関連する最近傍探索の効率化と識別関数の高速化計算に関する先行研究について整理する．

1.1.1 最近傍探索の効率化

Exemplar-SVM における識別は，最も高いスコアを出力する識別器を求め，そのスコアをしきい値処理することでクラスを判定する．これは最近傍探索と似た問題である．Exemplar-SVM の識別関数は，SVM と同様に特徴ベクトルと重みベクトルの内積により構築されるため，全ての識別器からスコアを求める処理は非常に大きなコストを要する．この処理を効率化し，ある特定の識別器のスコアを求めるだけで処理できれば，計算コストを大幅に削減できる．最近傍探索の分野では，ハッシュを用いたアプローチ [8, 9] と木構造を用いたアプローチ [10, 11, 12] が提案されている．

ハッシュを用いたアプローチでは，ハッシュ関数を用いてハッシュ値を計算し，登録データとテストデータ間のハッシュ値を比較することで最近傍を探索できる．ハッシュを用いた代表的な手法として Locality Sensitive Hashing(LSH)[8] や Spectral Hashing[9] が挙げられる．ハッシュ値を元の特徴ベクトルよりも低次元かつ，2 値で表現することによりハッシュ値を高速に比較できる．一般的に用いられるハッシュ関数は，特徴ベクトルと基底ベクトルの内積により計算される．提案手法では，特徴ベクトルが識別器のスコアに該当するため，全次元の特徴ベクトルを計算するためには，全ての識別器のスコアを計算する必要がある．そのため，提案手法においてはハッシュを用いた最近傍探索は非効率である．

一方，木構造を用いて最近傍探索を効率化する代表的な手法として，kd-tree[10] や Fast Library for Approximate Nearest Neighbors(FLANN)[12]，Randomized kd-tree[11] が挙げられる．木構造を用いたアプローチでは，特徴空間を 2 分割し，左右の子ノードにサンプルを分岐することを繰り返すことで決定木を構築する．そして，探索時には木をトラバーサル[2]することで最近傍を探索できる．特徴ベクトルが高次元な場合においては，最近傍探索の性能が低下することから，バックトラックや複数の二分木を用いる等の工夫が必要となる．木構造を用いた最近傍探索は，事前に特徴ベクトルを計算するハッシュを用いたアプローチとは異なり，探索時の各分岐ノードにおいて特徴ベクトルを計算すれば良い．本研究においても効果的に働くことが期待できるため，木構造を用いたアプローチを採用する．

の問題設定においても効率的な探索が期待できるため，提案手法では木構造を用いたアプローチを採用する．

1.1.2 識別関数の高速な計算

識別関数の高速化手法として，特徴ベクトルと重みベクトルの内積を近似的に計算する手法 [7, 13, 14] が提案されている．Hare らは，SVM により学習した重みベクトルを二値の基底行列と実数のスケール係数に分解し，内積計算をハミング距離の計算と数回の実数の掛け算で近似する方法 [7] を提案した．この方法は，入力特徴ベクトルが 2 値という条件を満たせば適用できる．この手法を発展させ，より近似性能を高めた方法 [14, 13] も提案されている．

さらに，近似内積計算時に途中で計算を打ち切ることで高速化する手法 [15, 14] も提案されている．近似内積計算により得られる値は，ポジティブとネガティブを正確に分けられる程度の精度で十分である．そのため，不必要に精度の高い近似解を求める必要はなく，これ以上の計算が必要ないと判断した場合には計算を打ち切る．提案手法では，近似内積計算と計算の早期打ち切りによる高速化を取り入れる．

1.2 提案手法の概要

本研究では，Exemplar-SVM を高速化するために，下記の 2 点を導入する．

[2]ルートノードから末端ノードまでを操作する処理．

- 木構造の導入
  最も類似した事例を探索，すなわち最も高いスコアを出力する Exemplar-SVM の識別器に対応した事例を高速に探索するために，木構造の Random Forests を導入する．
- 近似内積計算と早期分岐
  Exemplar-SVM の識別関数を近似的に計算することにより高速化する．さらに，決定木の分岐関数において，識別関数の計算を早期に打ち切ることにより高速な計算を実現する．

上記の 2 点により高速な識別が期待できる．さらに，Exemplar-SVM の識別器のスコアを比較する必要がないため，パラメータの調整が難しいキャリブレーションが不要となる利点もある．

2 Exemplar-SVM

本章では，Exemplar-SVM の識別器の学習とスコアのキャリブレーション，識別について述べる．

2.1 学習

Exemplar-SVM(E-SVM) では，ポジティブサンプル集合 $\mathcal{P}$ のある 1 つのサンプル $\mathbf{x}_E$ とネガティブサンプル集合 $\mathcal{N}$ から識別器 $f(\cdot)$ を学習する．E-SVM の目的関数は，ソフトマージン SVM の目的関数を拡張した式 (1) で表され，目的関数 $\Omega_E(\cdot)$ を最小化するよう重みベクトル $\mathbf{w}$，バイアス項 b を最適化する．

$$\begin{aligned}\Omega_E(\mathbf{w}, b) = \|\mathbf{w}\|^2 &+ C_1 h(\mathbf{w}^T\mathbf{x}_E + b) \\ &+ C_2 \sum_{\mathbf{x}\in N} h(-\mathbf{w}^T\mathbf{x} - b) \quad (1)\end{aligned}$$

$h(\cdot)$ はヒンジ損失関数であり式 (2) で表される．

$$h(\alpha) = \max(0, 1-\alpha) \quad (2)$$

式 (1) で表される目的関数の第 1 項はマージンの大きさ，第 2 項はポジティブサンプルに関するペナルティ項，第 3 項はネガティブサンプル集合に関するペナルティ項を表す．C_1, C_2 は各項のバランスを調整するパラメータである．

2.2 識別

E-SVM の識別関数 $f(\cdot)$ は，式 (3) のように特徴ベクトル $\mathbf{x}$ と重みベクトル $\mathbf{w}$ の内積により計算される．

$$f_i(\mathbf{x}) = \mathbf{w}_i^{\mathrm{T}}\mathbf{x} + b_i \quad (3)$$

E-SVM では，学習用ポジティブサンプルと同数の識別関数 $f(x) \in \mathcal{F}$ を学習する．全ての識別関数のスコアを計算し，最も高いスコアを出力する識別器 $f_{max}(\mathbf{x})$ を求める必要がある．

$$f_{max}(\mathbf{x}) = \arg\max_{f_i(\mathbf{x})\in\mathcal{F}} f_i(x) \quad (4)$$

式 (4) をしきい値処理することでクラス $\hat{y}$ を判定することができる．

$$\hat{y} = \mathrm{sign}(f_{max}(\mathbf{x}) + th) \quad (5)$$

ここで，th はしきい値を表す．なお，識別器 $f_{max}(\cdot)$ の学習に用いたポジティブサンプルの属性を出力することで，属性を転移することができる．

2.3 スコアのキャリブレーション

E-SVM では，全ての識別器を独立に学習するため，各識別器のスコアの関係性は表現されない．そのため，Malisiewicz らは E-SVM により学習した識別器のスコアを調整する方法を提案した．

キャリブレーションを行うために，まず学習用サンプルを用いて E-SVM により識別器を学習する．検出対象物を含む検証用画像からラスタスキャン方式により物体検出を行う．式 (3) にて表される sign 関数により処理する前の識別関数 $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$ のスコアが -1 以上となる検出対象画像 $(y = 1)$ と背景画像 $(y = -1)$ を収集する．そして，識別関数のスコアとラベル y の関係を次式のシグモイド関数によりモデル化する．

$$f'(\mathbf{x}|\mathbf{w}_E, \alpha_E, \beta_E) = \frac{1}{1 + \mathrm{e}^{-\alpha_E(\mathbf{w}_E^T\mathbf{x} - \beta_E)}} \quad (6)$$

ここで，α_E, β_E はシグモイド関数のパラメータである．この処理を全ての識別器に対して行う．識別時には，識別関数の内積計算の他に，式 (6) のキャリブレーションの計算も必要となるため，計算コストが増加する．

3 Exemplar-Random Forests

本章では，提案する Exemplar-Random Forests(E-RF) の学習と識別について述べる．提案手法では，図 2 に示すように学習した E-SVM の識別器を用いて Random Forests(RF)[6] を構築する．本研究では，RF を構成する決定木の分岐関数に E-SVM の識別関数を使用する．E-SVM の識別関数のスコアをしきい値処理することで，サンプルを左右の子ノードに分岐する．末端ノードには各クラスの事後確率を保持し，これを基にクラスを判定する．RF を用いることで，分岐ノードにて選択された E-SVM の識別器のみの計算で済むため，識別時の高速化が期待できる．

3.1 学習

3.1.1 Exemplar-SVM による事例型識別器の学習

2.1 にて述べた E-SVM により識別器を学習する．なお，本研究では E-SVM の識別関数の計算に近似計算法を導入するため，入力特徴ベクトル $\mathbf{x}$ を 2 値で表現する必要がある．本研究では HOG 特徴量 [16] を複数のしきい値より二値化した B-HOG 特徴量 [14] を用いる．

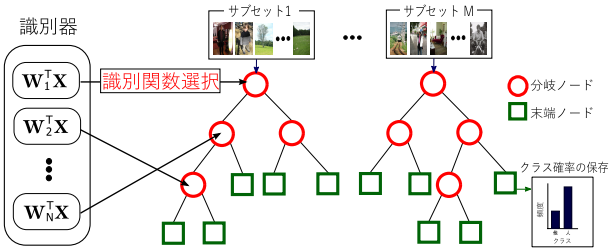

図 2　Exemplar-Random Forests の分岐関数．分岐関数では E-SVM の識別器のスコアをしきい値処理することによりサンプルを左右の子ノードに分岐する．

3.1.2　Exemplar-Random Forests による識別器の学習

RF は，ランダムな要素を取り入れて構築した決定木の集合であり，マルチクラス分類器を学習するアンサンブル学習アルゴリズムである．ランダム要素を取り入れることにより，高い汎化性能を持つ識別器を学習できる．本研究では 2 クラスを分類する Classification Forests を用いるが，非常に汎用性が高い手法であることから回帰分析や密度推定等にも利用されている．

RF は，複数の決定木により構成されている．決定木は，サンプルを左右の子ノードに分岐させる分岐ノードと，クラスの事後確率を記憶する末端ノードにより構成される．1 つの決定木を学習するために，教師信号付きサンプル集合 I の中からランダムにサンプルを選択し，複数のサンプルで構成されるサブセットを作成する．そして，1 つのサブセットを用いて 1 つの決定木を学習する．決定木の学習は，分岐ノードの作成と末端ノードの事後確率の保存の 2 つの処理に分けられる．

分岐ノード

分岐ノードでは，何らかの指標によりサンプルを 2 つに分割し，分けたサンプルを左右の子ノードに分岐する．本研究では，E-SVM の識別関数により計算されたスコアをしきい値処理し，左右の子ノードにサンプルを分岐する．

分岐ノード n における入力サンプルの集合を I_n，左右に分岐するサンプルの集合をそれぞれ I_l, I_r とすると，分岐条件は式 (7)，(8) で表される．

$$I_l = \{j \in I_n | f(\mathbf{x}_j) < t\} \tag{7}$$

$$I_r = I_n \setminus I_l \tag{8}$$

ここで，t はしきい値を表す．サンプルを分岐した後，式 (9) により情報利得 ΔE を求め，分岐関数を評価する．

$$\Delta E = -\frac{|I_l|}{|I_n|}E(I_l) - \frac{|I_r|}{|I_n|}E(I_r) \tag{9}$$

ここで，$E(I)$ とは，情報エントロピーであり次式により求める．

$$E(I_n) = -\sum_y \mathrm{P}(I_y)\log \mathrm{P}(I_y) \tag{10}$$

E-SVM の全ての識別関数及び幾つかのしきい値を試行し，情報利得が最も大きくなる組み合わせを分岐ノードの分岐関数として採用する．

末端ノード

決定木構築の終了条件は，分岐ノードの深さが設定した値の深さになるか，情報利得が 0 になるまでとする．末端ノード l では，クラスの確率分布 $\mathrm{P}(y)$ が作成される．

3.2　識別

構築した E-RF を用いてクラスを分類する．各決定木にテストサンプルを入力しトラバーサルする．たどり着いた末端ノードの確率分布の平均を式 (11) より求める．

$$\mathrm{P}(y|l) = \frac{1}{T}\sum_{t=1}^{T}\mathrm{P}(y|l_t) \tag{11}$$

T は決定木の数である．テストサンプルのクラスの推定には，次式で表す単純ベイズ分類器を用いる．

$$\hat{y} = \arg\max_y \mathrm{P}(y|L) \tag{12}$$

3.2.1　内積近似計算法の導入による高速化

RF を用いることにより，E-SVM の全ての識別器を計算する必要がなくなるため高速化が期待できる．さらなる高速化を実現するために，識別器の内積演算を近似的に計算する．

Hare らは，実数ベクトルを実数のスケール係数と二値基底行列に分解する方法 [7] を提案した．本手法では特徴ベクトルを二値のベクトルにすることで，内積計算をハミング距離の計算に置き換えることができるため，高速な近似内積計算が可能となる．本研究では Hare らの近似内積計算法 [7] を導入する．E-SVM により学習した重みベクトル $\mathbf{w}_i$ を式 (13) に示すようにスケール係数 $\mathbf{c}_i$ と二値基底行列 $\mathbf{M}_i$ に分解する．

$$\mathbf{w}_i \approx \mathbf{M}_i\mathbf{c}_i \tag{13}$$

ここで，$\mathbf{c}_i = (c_{i1}, c_{i2}, \cdots, c_{ik})^{\mathrm{T}} \in \mathbb{R}^k$ はスケール係数，$\mathbf{M} = \{\mathbf{m}_{i1}, \mathbf{m}_{i2}, \cdots, \mathbf{m}_{ik}\} \in \{-1, 1\}^{L\times k}$ は二値基底行列，$\mathbf{m}_{ij} \in \{-1, 1\}^k$ は k 個の二値基底ベクトルを表す．

式 (3) で示した重みベクトルと特徴ベクトルの内積の計算は，分解したスケール係数 $\mathbf{c}_i$ と二値基底行列 $\mathbf{M}_i$

を用いると下記のように表せる.

$$\mathbf{w}_i^{\mathrm{T}}\mathbf{x} \approx \mathbf{c}_i^T\mathbf{M}_i^T\mathbf{x} = \sum_{j=1}^{k} c_{ij}\mathbf{m}_{ij}^{\mathrm{T}}\mathbf{x}. \quad (14)$$

式 (14) における $\mathbf{m}_{ij}^{\mathrm{T}}\mathbf{x}$ の計算は，式 (15) のようにハミング距離の計算に置き換えられる.

$$\mathbf{m}_{ij}^{\mathrm{T}}\mathbf{x} = L - Hammaing\ Distance(\mathbf{m}_{ij}, \mathbf{x}) \quad (15)$$

ハミング距離は，対応する各ビットで XOR 演算を行った後，その結果に対して 1 となるビットをカウントするだけで計算可能であるため，極めて高速に計算できる. 式 (13) で示した重みベクトル $\mathbf{w}_i$ を分解する方法としていくつかの手法 [7, 13, 14] が提案されている．本研究では，Hare らの分解アルゴリズム [7] を採用する.

3.2.2 早期分岐の導入による高速化

内積の近似精度は基底数 k に依存する．しかし，サンプルによっては少ない基底数で近似計算しても十分な精度が得られる場合がある．また，RF の各ノードでは内積の近似値をしきい値処理してサンプルを左右の子ノードに分岐するが，分岐先を誤ることがなければ高い近似精度は不要である．既に近似計算の打ち切りを導入することで，計算を高速化するアプローチ [14, 15, 17] が採用されている．本研究においても，近似的に計算する過程でサンプルを早期に子ノードに分岐するアルゴリズムを導入する．早期分岐のアルゴリズムを **Algorithm**1 に示す.

Algorithm1 では，基底の数を 1 個ずつ増やし，設定したしきい値を用いて処理した結果，早期分岐が可能であるかを判定する．なお，しきい値 th は任意に設定するパラメータ，th_i^{pos}, th_i^{neg} は学習サンプルを用いて決定する．th_i^{pos} は学習用ポジティブサンプルを入力した際に最も小さなスコア，th_i^{neg} は学習用ネガティブサンプルを入力した際に最も大きなスコアとする.

Algorithm 1 早期分岐のアルゴリズム.

Require: 特徴ベクトル $\mathbf{x}$，スケール係数ベクトル $\mathbf{c}_i$，二値基底行列 $\mathbf{M}_i$，しきい値 th, th_i^{pos}, th_i^{neg}

(1) 初期化

$sum = 0$

for $j = 1 : k$ **do**

(2) 識別関数 $f(\mathbf{x})$ の近似計算.

$sum = sum + \mathbf{m}_{ij}^{\mathrm{T}}\mathbf{x}$

(3) 早期分岐の判定

if $sum > th_i^{pos} || sum < th_i^{neg}$ **then**

break

end for

(4) しきい値処理による分岐

if $sum > th$ **then**

右の子ノードへ分岐

else if $sum < th$ **then**

左の子ノードへ分岐

end if

4 評価実験

提案手法の有効性を確認するために評価実験を行う.

4.1 実験概要

提案手法の検出性能を確認するために，以下の項目について実験する.

- 各データベースにおける性能の違い
  見えの変化が小さい INRIA Person Dataset[16] と，見えの変化が大きい Pascal VOC 2007[18] の Car の 2 つのデータセットを用いる．それぞれのデータセットを用いて従来法ならびに提案手法の有効性を評価する.
- B-HOG 特徴量の影響
  近似計算法を導入するためには，特徴ベクトルを二値で表す必要がある．本研究では HOG 特徴量 [16] を複数のしきい値より二値化した特徴量 (B-HOG)[14] を用いる．特徴量を量子化すると量子化誤差の影響により性能が低下する．本実験では二値で表す影響を確認する.
- キャリブレーションの有無による性能差
  E-SVM は，スコアのキャリブレーションをしなければ性能が低下すると報告されている．各識別器を独立に学習するためであり，各識別器のスコアを直接的に比較できないことを示唆する．E-RF では，RF の分岐関数にて E-SVM の識別関数のスコアをしきい値処理することで左右の子ノードに分岐する．提案手法では，各識別器のスコアを比較する必要がないため，パラメータ調整が手間なキャリブレーションを省略できる可能性がある．本実験では，提案手法においてキャリブレーションの有無の影響を確認する.

4.2 検出性能の評価

実験結果の DET カーブを図 3 に示す．下記に各実験項目の結果を述べる.

4.2.1 各データベースにおける性能

図 3(a) に示す INRIA Person Dataset を用いた際の各手法の結果を比較する．まず，“HOG+SVM”と“HOG+E-SVM(キャリブレーション有)”を比較すると性能はほぼ同等であることが確認できた．見えの変化が

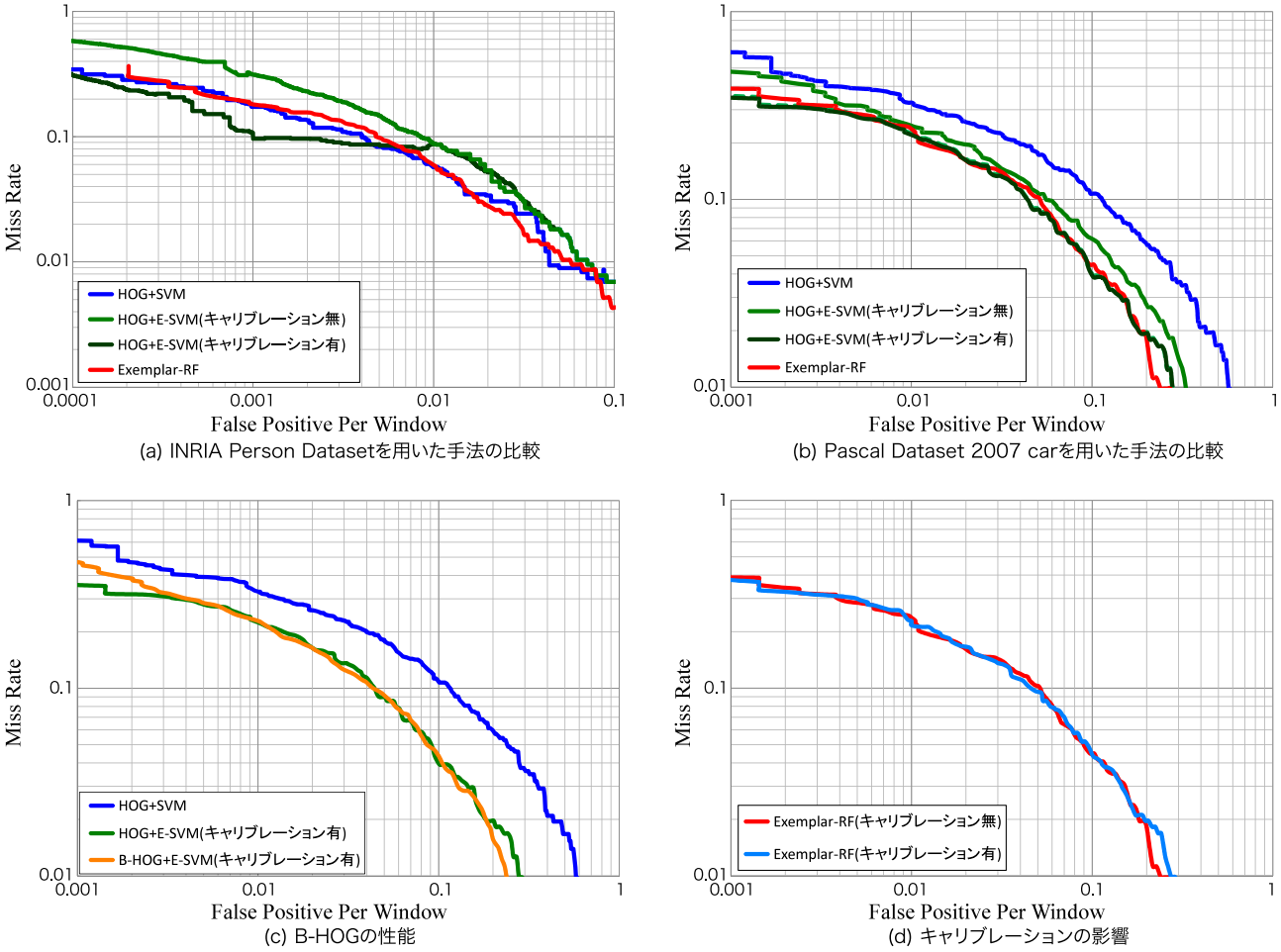

図 3　DET カーブによる実験結果．

小さい人検出用データセットでは，SVM と E-SVM は同等の性能を発揮することがわかる．次に，“HOG+E-SVM(キャリブレーション有)”と “HOG+E-SVM(キャリブレーション無)”を比較すると，キャリブレーションをしない場合は大きく性能が低下していることがわかる．E-SVM においては，キャリブレーションが必須であることを意味する．最後に，“HOG+E-SVM(キャリブレーション有)”と “Exemplar-RF”を比較すると，両手法ともに同等の性能であることがわかる．提案手法では，全ての E-SVM の識別器を計算していないが，検出性能を維持できることが確認できる．

次に，図 3(b) に示す Pascal VOC 2007 の car を用いた各手法の結果を比較する．見えの変化が大きいデータベースを用いた場合には，“HOG+SVM”の性能が低いことがわかる．“HOG+SVM”以外は，INRIA Person Dataset を用いた際の結果と同様の傾向であった．

4.2.2　B-HOG 特徴量の影響

図 3(c) に “HOG+E-SVM(キャリブレーション有)”と “B-HOG+E-SVM(キャリブレーション有)”の結果を示す．両手法の性能はほぼ同等であることが確認できる．本実験で用いた文献 [14] の特徴量は，実数で表現される特徴量を 4 つのしきい値より 5 つの二値符号に変換する．1 つのしきい値により二値符号に変換するよりも量子化誤差が少ないため，本手法により得られた B-HOG は人検出性能を維持できていることがわかる．

4.2.3　キャリブレーションの有無による性能差

図 3(d) に”Exemplar-RF(キャリブレーション有)”と”Exemplar-RF(キャリブレーション無)”の結果を示す．提案手法ではキャリブレーションをしない場合においても性能が低下しないことを確認した．キャリブレーションは，パラメータ最適化の手間と識別時の計算量の増加を招くため，キャリブレーションの省略可能な点は大きな利点である．

4.3　処理時間の評価

図 4 に未検出と処理時間の関係を表す．なお，処理時間は 1 ウィンドウあたりの画像を識別する時間であり，Pascal VOC 2007 の car の全てのテスト用のサンプルを処理した際の処理時間を平均し，さらにこれを 10 回繰り返した際の平均処理時間の平均を平均識別時間として採用した．

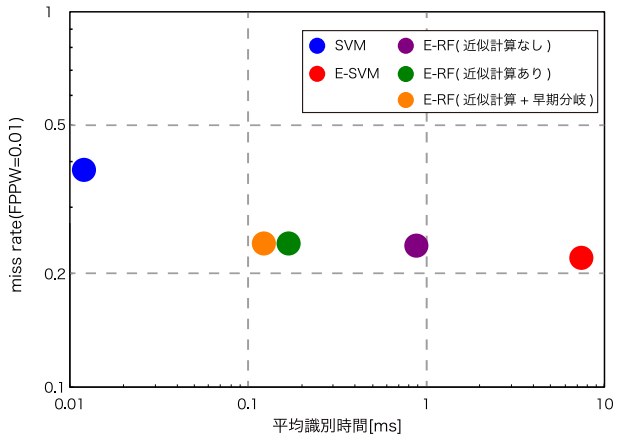

図 4　各手法の未検出と処理時間.

“E-SVM”は “SVM”と比較して，検出性能が高いものの識別時間が大きいことがわかる．提案手法である“E-RF”は “E-SVM”の検出性能を維持しながらも，大幅に計算コストを削減することができ，約 8.5 倍の高速化を実現した．さらに，近似計算法と早期分岐を導入することで，さらなる高速化を図ることができ，約 59.7 倍の高速化を達成した．

5　おわりに

本稿では，Exemplar-SVM の高速化として Random Forest を導入した Exemplar-Random Forests を提案した．Exemplar-SVM の識別処理において木構造を導入することで高速化を実現した．これにより，全ての識別器に対する計算が不要となり，ある幾つかの識別器に対しての演算のみで識別可能となる．さらに，識別時に近似内積計算と決定木における早期分岐を導入することでさらなる高速化を図った．評価実験より，提案手法は検出性能を維持しつつ，Exemplar-SVM よりも約 59.7 倍の高速化を実現したことを確認した．また，提案手法は Exemplar-SVM の識別器の各スコアを比較する必要がないため，キャリブレーションが不要であることを確認した．

Exemplar-SVM では，最近傍の事例を探索することにより属性を転移させることができる．しかしながら，提案手法では最近傍の事例を探索していないため，同じアプローチでは属性を転移させることができない．今後は，提案手法をベースとした属性の転移方法について検討する予定である．

参考文献

[1] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part-based models”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.32, no.9, pp.1627–1645, 2010.

[2] V. N. Vapnik, Statistical Learning Theory, Wiley, 1998.

[3] Y. Freund, and R. E. Schapire, “A Decision-theoretic Generalization of On-line Learning and an Application to Boosting”, European Conference on Computational Learning Theory, pp.23–37, 1995.

[4] T. Malisiewicz, A. Gupta, and A. A. Efros, “Ensemble of Exemplar-svms for Object Detection and Beyond”, IEEE International Conference on Computer Vision, pp.89–96, 2011.

[5] S. Song, and J. Xiao, “Sliding Shapes for 3D Object Detection in Depth Images”, European Conference on Computer Vision, 2014.

[6] L. Breiman, “Random Forests”, Machine Learning, vol.45, pp.5–32, 2001.

[7] S. Hare, A. Saffari, and P. H. S. Torr, “Efficient Online Structured Output Learning for Keypoint-Based Object Tracking”, CVPR, pp.1894-1901, 2012.

[8] P. Indyk, and R. Motwani, “Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality”, Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, pp.604–613, 1998.

[9] Y. Weiss, R. Fergus, and A. Torralba, “Multidimensional Spectral Hashing”, European Conference on Computer Vision, pp.340–353, 2012.

[10] J. L. Bentley, “Multidimensional binary search trees used for associative searching”, Communications of the Association for Computing Machinery, vol.18, no.9, pp.509–517, 1975.

[11] C. Silpa-Anan, and R. Hartley, “Optimised kd-trees for fast image descriptor matching.”, Computer Vision and Pattern Recognition, 2008.

[12] M. Muja, and D. Lowe, “Scalable Nearest Neighbor Algorithms for High Dimensional Data”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.36, no.11, pp.2227-2240, 2014.

[13] Y. Yamauchi, M. Ambai, I. Sato, Y. Yoshida, and H. Fujiyoshi, “Distance computation between binary code and real vector for efficient keypoint matching”, IPSJ Transactions on Computer Vision and Applications, vol.5, pp.124–128, 2013.

[14] M. Ambai, and I. Sato, “Spade: Scalar product accelerator by integer decomposition for object detection”, European Conference on Computer Vision, vol.8693, pp.267–281, Springer, 2014.

[15] 後藤雄飛，土屋成光，山内悠嗣，藤吉弘亘，“近似計算を導入した線形識別器の早期判定による高速な識別”，電子情報通信学会論文誌 D，vol.97，no.2，pp.294-302，2014.

[16] N. Dalal, and B. Triggs, “Histograms of oriented gradients for human detection”, Computer Vision and Pattern Recognition, vol.1, pp.886–893, 2005.

[17] 黒川貴都，山内悠嗣，安倍満，山下隆義，藤吉弘亘，“行列分解と早期棄却による多クラス物体検出の高速化”，コンピュータビジョンとイメージメディア研究会，vol.2015-CVIM-195，no.2，2015.

[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results”, http://www.pascal-network.org/challenges/VOC/voc2007/ workshop.