B-005

マルチモーダル入力に対応した重み付き多数決による識別器の GPU による高速化
A Technique for GPU-Based Acceleration of Classifier with Weighted Majority Voting for Multimodal Input

佐々木 仁† 坂本 博之† 雫 譲†
Jin Sasaki Hiroyuki Sakamoto Yuzuru Shizuku
黒木 修隆† 廣瀬 哲也† 沼 昌宏†
Nobutaka Kuroki Tetsuya Hirose Masahiro Numa

1. はじめに

近年，防犯意識の高まりから，監視システムの利用が増加している。それらシステムには，画像認識技術と識別器を組み合わせた自動的な異常検出機能が求められる。また，利用対象の多様化にともない，汎用的に利用可能な監視システムの構築が必要と考えられる。システムの汎用性を向上させる手段として，複数の特徴量を用いることが考えられる。複数の特徴量を適切に扱うために，信頼度による重み付けと複数の SVM（Support Vector Machine）[1] の多数決方式を導入することにより，マルチモーダルな入力に対応した識別器がある [2]。この識別器では，各特徴量に学習データにより算出した信頼度を用いて不要な特徴量の影響を抑制し，複数の SVM を用いて判定ミスの悪影響を低減することで，マルチモーダルな入力に対応した識別器を構成している。しかし，信頼度算出処理や複数の SVM を導入することで，処理時間の増加が問題となる。

そこで本稿では，マルチモーダルな入力に対応した重み付き多数決による識別器の演算量を削減し，GPU（Graphics Processing Unit）による並列処理を導入することで，高速化を実現する手法を提案する。

2. マルチモーダル入力に対応した重み付き多数決による識別器の構成

マルチモーダル入力に対応した識別器の構成を図 1 に示す。特徴量ごとに識別器に入力し，各識別器を複数のカーネル関数からなる SVM 集合で構成する。各 SVM の出力の重みの値としては，2.1 節で述べる信頼度を用いる。

2.1 信頼度

識別に不要な特徴量の影響を抑えるため，SVM ごとに信頼度を算出し，その出力値に重み付けを行う。学習したサンプルデータごとに，与えられたクラスラベルと各 SVM の判定が一致しているか否かを確認し，

$$\text{信頼度 } \beta = \left|\text{正解率} - 0.5\right| \times 2 \tag{1}$$

$$\text{正解率} = \frac{\text{クラスラベルとSVMの判定の一致回数}}{\text{学習回数}} \tag{2}$$

によって信頼度の計算を行う。識別に有効な特徴量ほど 1 に近い値となり，有効でなければ値は 0 に近づく。

2.2 複数の SVM による重みつき多数決

信頼度によって大きい重みを付けられた場合，その特徴量での判定ミスは多数決後の結果へ大きな悪影響を与える。それを防ぐために，特徴量ごとに複数の SVM からなる SVM 集合を用いる。SVM 集合 i 内の $\mathrm{SVM}_{i,j}$ （j = 1, 2, ･･･, N）には，N 種の異なるカーネル関数を用いる。カーネル関数による識別境界の違いを利用することで，一部の SVM の判定ミスの影響を抑える。M 個の特徴量に対し，N 種のカーネル関数を利用した場合，$\mathrm{SVM}_{i,j}$ の出力を $c_{i,j} \in \{\pm 1\}$，それに対応する信頼度 $\beta_{i,j}$ とすると，

$$C = \sum_{i=1}^{M}\sum_{j=1}^{N} \beta_{i,j} c_{i,j} \tag{3}$$

として多数決後の出力 C 得る。得られた出力値 C の正負を

$$y = \mathrm{sign}(C) \tag{4}$$

により判定することで，2 クラスの識別を行う。カーネル関数には，線形 L，2 次多項式 Q，ガウシアン G の 3 種類を用いる。

3. 提案手法

3.1 学習処理

学習処理では，学習に用いたデータセットの各サンプルの重みとオフセット，信頼度を算出する。まず学習サンプルを正規化し，ノイズを低減する。次にカーネル計算を行い，後段の処理を高速化するためのルックアップ・テーブルを作成する。その後，分類処理に必要な各サンプルの重み，オフセット，信頼度を算出する。各カーネル関数について，独立して SVM を作成して学習すると，学習サンプルの正規化をそれぞれの SVM で重複して行うこととなり，不必要な演算が発生する。また，表 1 から 2 次多項式カーネルは線形カーネルの計算結果をルックアップ・テーブルとして用いることが可能である。また，ガウシアン・カーネルでは

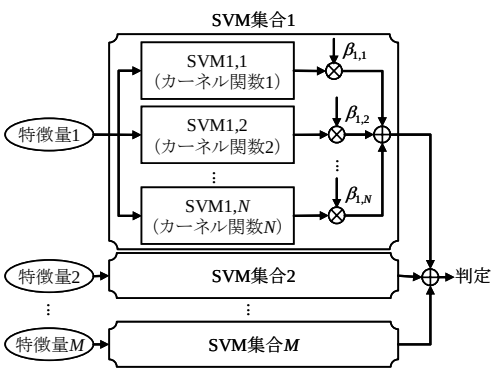

図 1 識別器の構成

†神戸大学大学院工学研究科,
Graduate School of Engineering, Kobe University

表 1　カーネル関数

カーネル関数	数式
線形	$\phi(x,x') = x \cdot x'$
2 次多項式	$\phi(x,x') = (1 + x \cdot x')^2$
ガウシアン	$\phi(x,x') = \exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)$

$$\|x-x'\|^2 = x \cdot x' - 2x \cdot x' + x' \cdot x' \quad (5)$$

とすることで，2 次多項式カーネルと同様に算出することが可能である。そこで，正規化処理を共通化し，線形カーネルの計算結果をルックアップ・テーブルとして利用することで，演算量を削減しつつ，並列処理することが可能であり，GPU を用いることで高速化を期待できる。

各処理の実装方法について述べる。正規化処理では，学習サンプル数分のスレッドを生成し，並列に処理を行う。カーネル計算処理では，まず CUDA [4]（Compute Unified Device Architecture）用に最適化された BLAS（Basic Linear Algebra Library）に含まれる SGEMM（Single General Matrix Multiply）関数を用いて，線形カーネルの計算を行う。その後，線形カーネルの計算結果を利用して，2 次多項式およびガウシアン・カーネルの算出を並列で処理する。重み算出処理では，最急降下法を用いて重みを算出する。このとき，勾配方向算出処理を GPU で行うことが考えられるが，学習サンプル数が少ない場合，データ転送時間の影響から高速化は困難であると考えられるので，学習サンプル数が少ない場合は GPU を用いない。オフセット算出処理は，十分に短時間に完了することが可能であるため，GPU は用いない。信頼度算出には，次節に述べる手法を用いて分類し，算出する。

3.2 分類処理

分類処理では，未知の入力ベクトル z を正規化後，

$$c = \mathrm{sign}\left\{b + \sum_i^n y_i \alpha_i \phi(x_i, z)\right\} \quad (6)$$

と表される識別関数を用いて各 SVM で分類する。その後，式 (3) を用いて信頼度に基づいた判定を行う。正規化処理はデータ転送時間を考慮すると GPU を用いた高速化は困難と考えられるので，CPU で処理する。学習処理と同様に線形カーネルの計算結果を利用して演算量を削減する。このときガウシアン・カーネルを

$$\|x-z\|^2 = x \cdot z - 2x \cdot z + z \cdot z \quad (7)$$

とする。式 (7) に必要な同一学習サンプル同士の内積は，学習処理中に算出されている結果を利用し，入力ベクトルの内積は正規化処理と同様に CPU で処理する。

4. 評価実験と考察

4.1 評価実験

提案手法を評価するために，図 2 に示す画像を用いて HOG（Histogram of Oriented Gradients）特徴量を用いた人物検出に適用し，評価を行った。従来手法として，SVM 集合の処理を CPU で行う手法を用いる。実験には，GPU には GeForce GT 240，CPU には Intel Core2 Duo 2.13GHz, RAM 2 GB を用いた。また，開発環境として Visual C++ 2008 Express Edition と CUDA 2.3 tool kit を用いた。

表 2　処理時間に関する比較評価結果

学習サンプル数	学習処理(ms)		認識処理(ms)	
	従来	提案	従来	提案
10	51	7	0.95	0.65
20	195	24	1.64	0.79
30	424	41	2.30	0.88
40	709	57	3.03	0.88
50	1,127	89	3.69	0.98
60	1,633	127	4.31	1.05
70	2,158	149	5.00	1.09
80	2,764	187	5.62	1.14
90	3,535	213	6.38	1.20
100	4,302	265	7.08	1.30

学習サンプル数については，学習処理と信頼度算出処理のために，各 10 枚から 100 枚まで変化させて学習を行い，学習処理に用いていない認識用画像 500 枚に対して認識処理を行った。評価項目は，学習処理と認識処理に要する処理時間とする。このとき，HOG 特徴量抽出処理時間を含まない識別器の処理時間のみを評価する。

4.2 評価実験結果

学習処理時間と認識処理時間に関する比較評価結果を表 2 に示す。表 2 から学習処理において最大 16.2 倍，認識処理において最大 5.4 倍の高速化を実現した。また，学習サンプル数の増加にともなう処理時間の増加を抑制している。この要因としては，学習サンプル数の増加にともなって並列度が高まるため，GPU を用いて並列処理することで，処理時間の増加を抑制することができたためと考えられる。

5. まとめ

本稿では，マルチモーダル入力に対応した重みつき多数決による識別器の演算量を削減し，GPU を用いることで高速化を実現する手法を提案した。提案手法により学習処理において 16.2 倍，認識処理において 5.4 倍の高速化を実現した。

今後の課題として，複数の特徴量を用いた評価実験を行い，提案手法を評価することが挙げられる。

参考文献

[1] 栗田多喜夫，“サポートベクターマシーン入門,” http://home.hiroshime-u.ac.jp/tkurita/lecture/svm.pdf
[2] 辻 亮弥, 西田喬士, 百崎将志, 廣瀬哲也, 黒木修隆, 沼 昌宏, “マルチモーダル入力に対応した重みつき多数決による識別器”, 第 10 回情報科学技術フォーラム（FIT2011）, H-010, 2011 年 9 月.
[3] CUDA, http://www.nvidia.co.jp/object/cuda_home_new_jp.html
[4] N. Dalal, B. Triggs, “Histogram of oriented gradients for human detection,” IEEE Computer Society Conference on Computer Vision and Pettern Recognition, pp. 886-893, 2005.

図 2　実験に用いた画像例