カーネル PCA を用いたパターン認識

和田萌　(指導教員：吉田裕亮)

1　はじめに

パターン認識とは, いくつかの概念に分類出来る観測データが存在するとき, 観測されたパターンをそれらの概念のうちのひとつに対応させることである.

近年の社会の情報化は, 文字を介してコンピューターへ直接アクセスする用例や, データベースを構築する為に, 膨大な量の文字データを OCR で読み取る必要性を生み出した. 更に, ワークステーションやパソコンの普及は既に存在する文書を読み取り, 再利用を測る為のフレキシブルな入力を求めており, 手書きパターン認識技術は不可欠な技術となっている.

そこで, 本研究ではこのパターン認識をカーネル PCA を用いて, 機械学習により判別器の構成を行った.

2　PCA(主成分分析)

PCA(principal component analysis) とは, 多次元データの情報をその総合力の特性を保ちながら, より低い次元に縮約させる方法.

X を $p \times n$ のデータ行列とし, X の縦成分 (変数) ごとに標準化し, その行列を X_0 とする. この標準化されたデータの相関行列 R を求める.

$$R = \frac{1}{p}(X_0)^t X_0$$

行列 R の固有方程式を解き, n 個の固有値と, 各々の固有値に対応する固有ベクトルからなる行列 V を求める. 固有値は元データの情報保持率を表すことになる. 標準化されたデータ行列 X_0 を,

$$X_0 V = X^*$$

と変換する. X^* の第 1 列目は第 1 主成分, X^* の第 2 列目は第 2 主成分と呼ばれる.

3　カーネル法

カーネル法とは, (データ集合を X とする時) 2 つのデータの間にある種の類似度を表すカーネル関数, $K : X \times X \to \mathbb{R}$ を通じてデータにアクセスするような学習モデルである.

具体的には, n 個のデータ $x^{(1)}$, $x^{(2)}$, ..., $x^{(n)} \in X$ が与えられたときに, モデル $f(\xi)$ が,

$$f(\xi) = g\left(\sum_{i=-1}^{n} \alpha^{(i)} K\left(\xi, x^{(i)}\right)\right)$$

のような形, つまり, カーネル関数の線形結合の関数として表現されるようなモデルを扱うのがカーネル法である. なお, g はある関数, また $\{\alpha^{(i)}\}_{i=1}^{n}$ はモデルパラメータである.

カーネル関数はどのような関数でも良いという訳ではなく, 内積として書くことが出来る必要がある. つまり, データ $x^{(i)}$ の d 次元の特徴空間中でのベクトル表現を $\phi(x^{(i)})$ とすると, i 番目と j 番目のデータの間のカーネル関数が

$$K(x^{(i)}, x^{(j)}) = < \phi(x^{(i)}), \phi(x^{(j)}) >$$

のように定義されている必要がある.

4　カーネル PCA

PCA は線形の関係 (相関関係) を基礎とした分析であった. それをカーネル法を用いて非線形に拡張した手法がカーネル PCA である. この分析では, 非線形空間への写像を行い, その空間での PCA をカーネル法により行う分析である. カーネル法を用いることで, 実際に非線形な写像を計算することなく非線形な空間での PCA が行うことができる. 以下にその定化式を示す.

今, 共分散行列

$$S = \frac{1}{n}\sum_j x_j x_j^\tau$$

は非線形な空間への写像 ϕ とすると,

$$C = \frac{1}{n}\sum_j \phi(x_j)\phi(x_j)^\tau$$

となる. この時の固有値問題は, 同様に

$$Ca = \lambda a$$

となる. ここで

$$z_j = \phi(x_j)^\tau a$$

である. よって,

$$\begin{aligned} Ca =& \frac{1}{n}\sum_j \phi(x_j)\phi(x_j)^\tau \frac{1}{n\lambda}\sum_k z_k \phi(x_k) \\ =& \frac{1}{n^2\lambda}\sum_{j,k} z_k \phi(x_j) K(x_j, x_k) \end{aligned}$$

$$\lambda a = \frac{1}{n}\sum_j z_j \phi(x_j)$$

となる. ここで

$$K(x_j, x_k) = \phi(x_j)^\tau \phi(x_k)$$

はカーネル関数である. カーネルには Linear, Polynomial, Gaussian, Sigmoid 等が代表的である. よって,

$$\frac{1}{n}\sum_j z_k K(x_j, x_k) = \lambda z_j$$

となる.

5　提案手法

本研究ではカーネル PCA を用いて, 手書き文字 (数字) の判別を以下のように行うことを提案する.

1. カーネル選択ではガウスカーネルを使用する.

$$K(x,y) = \exp(-||x-y||^2/\sigma^2)(\sigma > 0)$$

2. それぞれのデータ間距離を計算する.

$$||\vec{y} - \frac{1}{n}(\vec{x_1} + \vec{x_2} + \cdots + \vec{x_n})||^2$$
$$= ||\vec{y}||^2 - \frac{2}{n}\sum_{i=1}^{n}(\vec{y}, \vec{x_i}) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\vec{x_i}, \vec{x_j})$$

この値が一番小さくなる数字グループに属すると判断する.

6 数値実験

6.1 手書き数字データ

ここでは, 公開されているアメリカ合衆国郵便公社が業務で得た実際の手書き数字データを使用する. 数字 10 種類の手書きデータを各 100 個用意する. 画像は 16×16 画素の 256 次元を使用する. 各画素値は 256 階調の離散値である.

1. 0〜9 の判別

0〜9 を一度に判別を試みたが, 数字が多すぎるため判別は不可能であった.

そこで, 似ているもの同士を同じグループとしてまとめ 2 群にわけたあと, それぞれのグループ内で各数字を判別する階層的方法をとった.

0, 2, 3, 5, 6 と 1, 4, 7, 8, 9 の 2 群に分けたときが, 最も判別正答率が高いためこれを使用する.

この段階における判別の正答率は, 学習データでは 96%, また学習データとは違うデータを使って汎化性能を調べた場合は 92%であった.

2. 0, 2, 3, 5, 6 の判別

0, 2, 3, 5, 6 のデータを各 50 個ずつ用意し, 250 個のデータに対して改めてパラメータを選択し, 判別機にかける. 以下の表からわかるように, どの判別においても, 100%に近い高い正答率を得ることができた.

次に, 学習データとは異なるデータをテストデータとして 50 個用意し, 判別を行い汎化性能を調べた. 平均正答率は 96.4%と, 学習データ判別の正答率と比較すると低い数値ではあるが, かなり高い正答率を得られた.

n	学習データ正答率	汎化性能正答率
0	100%	100%
2	100%	100%
3	98%	92%
5	100%	96%
6	100%	94%

3. 1, 4, 7, 8, 9 の判別

同じ方法で 1, 4, 7, 8, 9 の学習データの判別正答率, 汎化性能を調べた結果は以下の表のようになった.

n	学習データ正答率	汎化性能正答率
0	92%	80%
2	94%	86%
3	94%	90%
5	92%	76%
6	90%	88%

0, 2, 3, 5, 6 のときと比較すると, どちらも低い数値であるが, 高い正答率を得られた.

ここで, 誤判別された画像データが以下のようなものであった.

図 1: (誤判別例) 1

図 2: (誤判別例) 7

3. ノイズを除いた 1, 4, 7, 8, 9 の判別

そこで, 上のような人の目でみても明らかに判別が難しい数字データを学習データから除き, 改めて 50 個用意し, 判別を行った. すると, 下の表のようにすべての数字に対し, ほぼ 100%の高い正答率を得ることができた.

n	改良前正答率	改良後正答率
0	92%	100%
2	94%	96%
3	94%	100%
5	92%	98%
6	90%	94%

7 まとめと今後の課題

昨年の研究では, 256 次元のデータに対して PCA により次元を下げた後, 2 群の判別 (例:3 であるか, そうでないか) を行った. その拡張として, 本研究ではカーネル PCA を用いて, 次元を落とす手間を省き, 256 次元のデータをそのまま 2 段階で 10 通りの文字の識別を行った. この手法はかなり単純な計算法で判別を行うことができるため, 有効であると思われる. さらに, 学習に用いるデータ数を増やすことで, 判別における正答率を高めたい. また, 数字判別より複雑なアルファベットやカタカナなどにおいてもこの手法が有効であるか確認したい.

参考文献

[1] B.Schlkoph and A.J.Smola, Learning With Kernel, MIT Press, 2002.

[2] 石村貞夫, すぐわかる多変量解析, 東京図書 (1992)