卒業論文

SVMを用いた楽曲に対する感性情報の可視化

東海林研究室

5年情報工学科

14番　武井亨

目次

1.目的

近年，インターネットによる音楽配信サービスがさかんに行われている．多くの人に利用されている iTunes Music Store で用意されている楽曲数は100万曲以上であり，音楽に関する関心は高まってきている．しかし，楽曲の数は無数にあり，音楽を聴く人にとって好みの楽曲を探し出すのは困難である．

そこで様々な楽曲推薦システムが考えられている．例えば，ある楽曲をダウンロードした時，過去にその曲をダウンロードした人が多くダウンロードしている別の楽曲を新たに推薦するシステムである．また，新たな楽曲推薦システムとして，ユーザ個人の過去のダウンロードの履歴からそのユーザの嗜好を推測し，その結果に基づいて新たな楽曲を推薦するシステムが考えられる．

本研究では，そのような個人の嗜好に基づいた楽曲推薦システムの足掛かりとなるよう，楽曲の特徴と楽曲を聴いた時に感じる印象との関係を調べる．識別モデルのひとつである SVM(Support Vector Machine)を用いて楽曲の特徴と楽曲を聴いた時に感じる印象との関係を 2 次元画像として可視化し，どのような関係があるのかを調べる．さらに，楽曲の特徴からその曲を聴いた時に感じる印象を SVM によって識別する．

２．SVM

２－１　パターン認識における学習

パターン認識を実現するためには，まず，認識対象から何らかの特徴量を計測（抽出）する必要がある．一般には，特徴量は 1 種類だけではなく，複数の特徴 量を計測し，それらを同時に用いることが多い．そのような特徴量は，通常，ま とめて特徴ベクトル $x^T=(x_1,\cdots,x_M)$として表される．ここで，x^T は，ベクトル x の転置を表す．また，M は，特徴量の個数で ある．認識対象のクラスの総数を K とし，各クラスを $C_1,\cdots C_K$ と表す ことにする．

パターン認識における最も基本的な課題は，未知の認識対象を計測 して得られた特徴ベクトルからその対象がどのクラスに属するかを判定する識別 器を開発することである．そのためには，クラスの帰属が既知の訓練用のサンプル集合から特徴ベクトルとクラスとの確率的な対応関係を知識として学習するこ とが必要である．未知の認識対象の識別には，学習された確率的知識を利用して それがどのクラスに属していたかを推定（決定）する方式を指定しなければなら ない．その際，間違って識別する確率をできるだけ小さくすることが望ましい．

特徴ベクトルとクラスとの確率的な対応関係が完全にわかっている理想的な場合には，理論的に最適な識別方式（ベイズ識別方式）が存在する．しかし，実際の パターン認識問題では，特徴ベクトルとクラスとの確率的な対応関係が完全にわ かっていることは稀で，そのような確率的な関係を訓練サンプルから学習する必要 がある．

２－２　SVM について

SVM は未学習データに対して高い識別性能を得ることができ，現在知られている多くの手法の中でも認識性能の優れたクラス識別モデルの一つである．

SVM にデータの座標値とクラス値を学習させることにより，SVM は2つのクラスを分類する最適な識別関数を構成する．最適な識別関数は2つのクラスのデータからの距離が最も遠くなるように構成される．この識別関数とデータの距離をマージンという[1]．

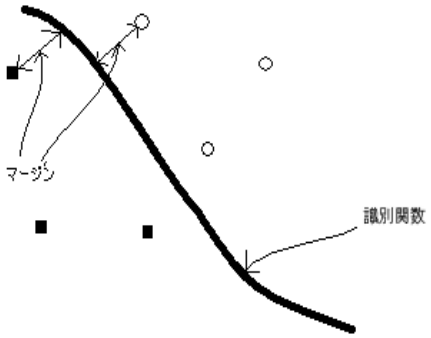

図 1　識別関数とマージン

カーネルトリックにより非線形の識別関数が構成で きるように拡張した SVM は，現在知られている手法の中でも最もパターン認識性能の優秀な学習モデルの一つである．ただし，サポートベク ターマシンは，基本的には 2 つのクラスを識別する識別器を構成するための学習 法であり，文字認識などの多クラスの識別器を構成するためには，複数のサポー トベクターマシンを組み合わせるなどの工夫が必要となる．一般に，カーネル学習法を用いて学習された識別器が，訓練サンプルに含まれていない未学習データに対しても高い 識別性能を発揮できるためには，汎化能力を向上させるための工夫が必要であるが SVM では，「マージン最大化」という基準を用いることでこ れを実現している．

2—3　SVM の原理

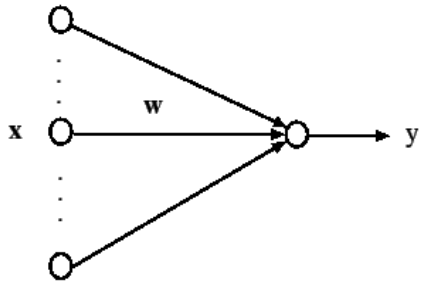

図 2　線形しきい素子

SVM は，線形しきい素子のモデルの中で最も単純な線形しきい素子を用いて，2 クラスのパターン識別器を構成する手法である．訓練サンプル 集合から，「マージン最大化」という基準で線形しきい素子のパラメータを学習 する．線形しきい素子は，図 2 に示すような線形しきい素子を単純化したモデルで，入力特徴ベクトルに対し，識別関数（線形識別関数）

$$y=sign(w^t x-h) \qquad \cdots(1)$$

により 2 値の出力値を計算する．ここで，w はシナプス荷重に対応するパ ラメータであり，h はしきい値である．また，関数 sign(u)は，u>0 のとき 1 をとり，　$u\leq 0$　のとき-1 をとる符号関数である．このモデルは，入力ベクトルとシナプス荷重の内積がしきい値を超えれば 1 を出力し，超えな ければ-1 を出力する．これは，幾何学的には，識別平面により，入力特徴空間 を 2 つに分けることに相当する．今，2 つのクラスを c_1,c_2とし，各クラス のラベルを 1 と-1 に数値化しておく．また，訓練サンプル集合として，N 個の特徴ベクトル $x_1,\cdots,x_N$ と，それぞれのサンプルに対 する正解のクラスラベル $t_1,\cdots,t_N$ が与えられているとする．また，この 訓練サンプル集合は，線形分離可能であるとする．すなわち，線形しきい素子のパラメータをうまく調整することで，訓練サンプル集合を誤りなく分けることが できると仮定する．

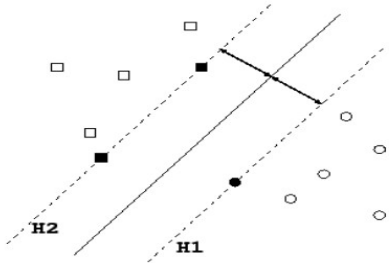

図 3　線形しきい素子の分離超平面とマージン

(◯がクラス 1 のサンプルで，□がクラス-1 のサンプルを示す．●と■はサポートベクターを示す．)

訓練サンプル集合が線形分離可能であるとしても，一般には，訓練サンプル集合を誤りなく分けるパラメータは一意には決まらない．サポート ベクターマシンでは，訓練サンプルをすれすれに通るのではなく，なるべく余裕をもって分けるような識別平面が求められる．具体的には，最も近い訓練サンプ ルとの余裕をマージンと呼ばれる量で測り，マージンが最大となるような識別平 面を求める．もし，訓練サンプル集合が線形分離可能なら，

$$t_i(w^T x_i-h)\geq 1 \quad i=1,\cdots,N \cdots(2)$$

を満たすようなパラメータが存在する．これは，H1:　$w^T x_i-h=1$　と H2:　$w^T x_i-h=-1$　の 2 枚の超平面で訓練サンプルが完全に分離されてお り，2 枚の超平面の間にはサンプルがひとつも存在しないことを示している．こ のとき，識別平面とこれらの超平面との距離（マージンの大きさ）は，　$\dfrac{1}{\|w\|}$　となる．したがって，マージンを最大とするパラメータ w と h を求める問題は，結局，制約条件

$$t_i(w^T x_i-h)\geq 1 \qquad \cdots(3)$$

の下で, 目的関数

$$L(w)= \frac{1}{2}\|w\|^2 \qquad \cdots(4)$$

を最小とするパラメータを求める問題と等価になる. ここで, 双対問題に帰着して解く方法を用いる. まず, Lagrange 乗数 $\alpha_i(\geq 0), i=1,\cdots,N$ を導入し, 目的関数を

$$L(w,h,\alpha)= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i\{t_i(w^t x_i - h)-1\} \qquad \cdots(5)$$

と書き換える. パラメータ w および h に関する偏微分から停留点では,

$$w= \sum_{i=1}^{N} \alpha_i t_i x_i \qquad \cdots(6)$$

$$0=\sum_{i=1}^{N} \alpha_i t_i \qquad \cdots(7)$$

という関係が成り立つ. これらを上の目的関数の式に代入すると, 制約条件,

$$\sum_{i=1}^{N} \alpha_i t_i=0 \qquad \cdots(8)$$

$$0\leq \alpha_i \qquad i=1,\cdots,N \qquad \cdots(9)$$

の下で, 目的関数

$$L_D(\alpha)=\sum_{i=1}^{N} \alpha_i-\frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j t_i t_j x_i^T x_j \quad \cdots(10)$$

を最大とする双対問題が得られる. これは, Lagrange 乗数 $\alpha_i(\geq 0)$に関する最適化問題となる. その解で α_i^*が 0 でない, すなわち, $\alpha_i^*>0$ となる訓練サンプル x_i は, 先の 2つの超平面か $W^Tx-h=1$ か $W^Tx-h=-1$ のどち らかにのっている. このことから, α_i^*が0でない訓練サンプル x_i は「サポートベクター」と呼ばれている. 直感的に理解できるように, 一般には, サポー トベクターは, もとの訓練サンプル数に比べてかなり少ない. つまり, 沢山の訓練サンプルの中から少数のサポートベクターを選び出し, それらのみを用いて 線形しきい素子のパラメータが決定される.

実際, 双対問題の最適解 $\alpha_i^*(i\geq 0)$および停留点での条件式か ら, 最適なパラメータ w^*は,

$$w^* \sum_{i\in S} \alpha_i^* t^i x^i \qquad \cdots(11)$$

となる. ここで, S はサポートベクターに対応する添え字の集合である. また, 最適なしきい値 h^*は, 2つの超平面 $w^T x-h=1$ か $w^T x-h=-1$ のどちらかにのっているという関係を利用して求め ることができる. すなわち, 任意のサポートベクター x_s, $s\in S$ から

$$h^*=w^{*T}x_s-t_s \qquad \cdots(12)$$

により求まる.

また, 最適な識別関数を双対問題の最適解 $\alpha_i^*(i\geq 0)$を用いて表現 すると

$$y = \mathrm{sign}(w^{*T}x-h^*)$$

$$= \mathrm{sign}\left(\sum \alpha_i^* t_i x_i^T x-h_*\right) \qquad \cdots(13)$$

となる. すなわち, $\alpha_i^*=0$ となる多くの訓練サンプルを無視し, $\alpha_i^*>0$ となる識別平面に近い少数の訓練サンプルのみを用いて識別関数 が構成される. ここで, 重要な点は, 「マージン最大化」という基準から自動的 に識別平面付近の少数の訓練サンプルのみが選択されたことであり, その結果として, 未学習データに対してもある程度良い識別性能が維持できていると解釈できる. すなわち, SVM は, マージン最大化という基準を用いて, 訓練サンプルを選択する.

２－４　ソフトマージン

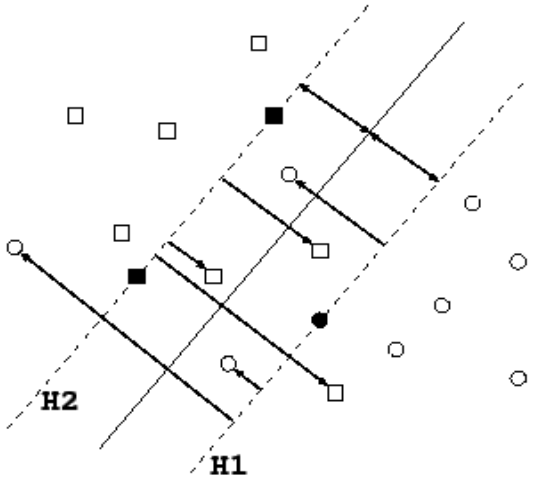

図 4 ソフトマージン
(◯がクラス 1 のサンプルで，□がクラス-1 のサンプルを示す．●と■はサポートベクターを示す．)

上述の SVM は，訓練サンプルが線形分離可能な場合につい ての議論であるが，パターン認識の実問題で線形分離可能な場合は稀である．したがって，実際的な課題に SVM を使うには，多少の識別誤りは許すように制約を緩める方法を用いる．これは，「ソフトマージン」と呼ばれている．[4]．

ソフトマージン法では，マージン $\frac{1}{\|w\|}$ を最大としながら，図4に示すように，幾つかのサンプルが超平面 H1 あるいは H2 を越 えて反対側に入ってしまうことを許す．反対側にどれくらい入り込んだかの距離 を，パラメータ $\varepsilon_i(\geq 0)$を用いて， $\frac{\varepsilon_i}{\|w\|}$ と表す とすると，その和

$$\sum_{i=1}^{N} \frac{\varepsilon_i}{\|w\|} \qquad \cdots(14)$$

をなるべく小さくする．これらの条件から最適な識別面を求める問題は，制約条件

$$\varepsilon_i \geq 0 \ , \ t_i(w^T x_i - h) \geq 1 - \varepsilon_i \quad (i=1,\ldots,N) \quad \cdots(15)$$

の下で，目的関数

$$L(w, \varepsilon) = \frac{1}{2}\|w\|^2 + \gamma \sum_{i=1}^{N} \varepsilon_i \qquad \cdots(16)$$

を最小とするパラメータを求める問題に帰着される．

この最適化問題の解法は，基本的には線形分離可能な場合と同様にふたつの制約 条件に対して，Lagrange 乗数 α_i，および，ν_i を導入し，目的関数 を

$$\begin{aligned} L(w,h,\alpha,\nu) = & \frac{1}{2}\|w\|^2 + \gamma \sum_{i=1}^{N} \varepsilon_i \\ & - \sum_{i=1}^{N} \alpha_i \{ t_i(w^T x_i - h) - (1 - \varepsilon_i) \} \\ & - \sum_{i=1}^{N} \nu_i \varepsilon_i \qquad \cdots(17) \end{aligned}$$

と書き換える．パラメータ w,h, ν_i に関する偏微分を 0 とする 停留点では，

$$w=\sum_{i=1}^{N} \alpha_i t_i x_i \qquad \cdots(18)$$

$$0=\sum_{i=1}^{N} \alpha_i t_i \qquad \cdots(19)$$

$$\alpha_i = \gamma - \nu_i \qquad \cdots(20)$$

という関係が成り立つ. これらを目的関数の式に代入すると, 制約条件

$$\sum_{i=1}^{N} \alpha_i t_i = 0 \qquad \cdots(21)$$

$$0 \le \alpha_i \le \gamma \ , \ i=1,\cdots,N \qquad \cdots(22)$$

の下で, 目的関数

$$L_D(\alpha)=\sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j t_i t_j x_i^T x_j \qquad \cdots(23)$$

を最大とする双対問題が得られる. 線形分離可能な場合には, 最適解 α_i^*の値により, 平面 H1 および H2 上の訓練サンプル(サポートベクター) とそれ以外のサンプルに分類されたが, ソフトマージンの場合には, さらに, H1 および H2 をはさんで反対側にはみ出すサンプルが存在する. それらは, 同様に, 最適解 α_i^*の値により区別することができる. 具体的には, $\alpha_i^*=0$ なら, 平面 H1 あるいは H2 の外側に存在し, 学習された識別器に よって正しく識別される. また, $0<\alpha_i^*<\gamma$ の場合には, 対応するサンプルは, ちょうど平面 H1 あるいは H2 の上に存在するサポートベクター となり, これも正しく識別される. $\alpha_i^*=0$ の場合には, 対応するサンプルはサポートベクターとなるが, $\varepsilon_i \neq 0$ となり, 平面 H1 あ るいは H2 の内側に存在する.

2—5　カーネルトリック

ソフトマージン法を用いることで, 線形分離可能でない場合に対しても線形しきい素子のパラメータを求められるが, ソフトマージン法を用いたとしても, 本質的に非線形で複雑な識別課題に対しては, 必ずしも良い性能の識別器を構成できるとは限らない. 本質的に非線形な問題に対応するための方法として, 特徴ベクトルを非線形変換して, その空間で線形の識別を行う「カーネルトリック」と呼ばれている方法を用いる[4].

一般に, 線形分離可能性はサンプル数が大きくなればなるほど難しくなり, 逆に, 特徴空間ベクトルの次元が大きくなるほど易しくなる. しかし, 高次元への写像を行うと, 次元の増加に伴い汎化能力が落ちてしまう. また, 難しい問題を線形分離可能にするためには, 訓練サンプルと同程度の大きな次元に写像しなければならないので, 結果的に膨大な計算量が必要となってしまう.

元の特徴ベクトル x を非線形の写像 Φ(x)によって変換し, その空間で線形識別を行う. 例えば, 写像 Φ として, 入力特徴を2次の多項式に変換する写像を用いるとすると, 写像した先で線形識別 を行うことは, もとの空間で2次の識別関数を構成することに対応する. 一般には, こうした非線形の写像によって変換した特徴空間の次元は非常に大きくなる. しかし, SVM の場合には, 目的関数 L_Dや識別関数が入力パターンの内積のみに依存した形になっており, 内積が計算できれば最適な識別関数を構成することが可能である. つまり, もし非線形に写像した空間での二つの要素 $\Phi(x_1)$と $\Phi(x_2)$の内積が

$$\Phi(\mathrm{x}_1)^{\mathrm{T}}\Phi(\mathrm{x}_2)=\mathrm{K}(\mathrm{x}_1,\mathrm{x}_2) \qquad \cdots(24)$$

のように, 入力特徴 x_1と x_2のみから計算できるなら, 非線形写 像によって変換された特徴空間での特徴 $\Phi(x_1)$や $\Phi(x_2)$を 陽に計算する代わりに, $K(x_1,x_2)$から最適な非線形写像を構成 できる. ここで, このような K のことをカーネルと呼んでいる. このように高次元に写像しながら, 実際には写像された空間での特徴の計算を避けてカーネルの計算のみで最適な識別関数を構成することを「カーネルトリック」という.

実用的には, K は計算が容易なものが望ましい. 例えば,
多項式カーネル

$$\mathrm{K}(\mathrm{x}_1,\mathrm{x}_2)=(1+\mathrm{x}_1^{\mathrm{T}}\mathrm{x}_2)^{\mathrm{P}} \qquad \cdots(25)$$

Gauss カーネル

$$K(x_1,x_2)=\exp\left(\frac{-\|x_1-x_2\|^2}{2\sigma^2}\right) \quad \cdots(26)$$

シグモイドカーネル

$$K(x_1,x_2)=\tanh(ax_1^{T}x_2-b) \quad \cdots(27)$$

などが使われている．本研究ではこの中でガウスカーネルを用いた．

式(10)や式(23)の目的関数 L_D は，

$$L_D(\alpha)=\sum_{i=1}^{N}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j t_i t_j \Phi(x_i)^T\Phi(x_j)$$

$$=\sum_{i=1}^{N}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j t_i t_j K(x_i,x_j) \quad \cdots(28)$$

のように内積をカーネルで置き換えた形に書ける．また，式(13) から最適な識別関数は，

$$y=\mathrm{sign}(w^{*T}\Phi(x)-h^{*})$$

$$=\mathrm{sign}\left(\sum_{i\in S}\alpha_i^{*}t_i\Phi(x_i)^T\Phi(x)-h^{*}\right)$$

$$=\mathrm{sign}\left(\sum_{i\in S}\alpha_i^{*}t_iK(x_1,x)-h^{*}\right) \quad \cdots(29)$$

のように SVM の内積をカーネルで置き換えた形に書ける．ここ で，Gauss カーネルを用いると，構造的には従来のニューラルネッ トワークと同じになる．しかし，カーネルトリックを用いて非線形に拡張した SVM では，中間層から出力層への結合荷重のみが学習により決定され，前段の入力層から中間層への結合荷重は固定で，訓練サンプルから機械的 に求められる．また，中間層のユニット数が非常に大きく，訓練サンプル数と同 じになる．つまり，カーネルトリックを用いて非線形に拡張した SVM では，入力層から出力層への結合荷重を適応的な学習により求めない代わ りにあらかじめ中間層に非常に多くのユニットを用意することで複雑な非線形写像を構成する．

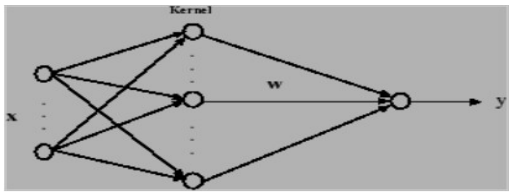

図 5 SVM の構造

３. 正準相関分析について

3変数以上が存在したとき, 2変数間同士の関係は相関係数によって知ることができる. しかし, いくつかの変数の組同士の関係を, 変数間同士の関 係だけから推論することは非常に難く, 例えば, q 個の変数が存在したとき, それらの変数の r 個の組と, 残りの (q - r) 個の組との関係を得たいような場合に正準相関分析を用いる.

今, q 個の変数,

$$x_1, x_2, \cdots, x_q$$

で表されるデータがあったとする. このとき,

$$r \text{ 個の変数の組}: x_1, x_2, \cdots, x_r$$

$$s(= q - r \geqq r) \text{個の変数の組}: x_{r+1}, x_{r+2}, \cdots, x_{r+s}$$

を考え, それらの線形結合から成る2つの変数,

$$y = a_1x_1 + a_2x_2 + \cdots + a_rx_r \quad \cdots(30)$$

$$z = b_1x_{r+1} + b_2x_{r+2} + \cdots + b_sx_{r+s} \quad \cdots(31)$$

を考え, y と z との間の関係によって, 2つの組間の関係を得る. ただし, y と z との間の相関係数 r_{yz} を最大にするように各係数を定める.

各係数を求めるのに先立ち, 以下に示すようなベクトル, 及び, 行列を定義する.

$$\mathbf{X}_1 = \begin{bmatrix} x_1 \\ x_2 \\ \cdot\cdot \\ x_r \end{bmatrix} \quad \mathbf{X}_2 = \begin{bmatrix} x_{r+1} \\ x_{r+2} \\ \cdots \\ x_{r+s} \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \cdot\cdot \\ a_r \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \cdot\cdot \\ b_s \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} \\ \mathbf{C}_{21} & \mathbf{C}_{22} \end{bmatrix}$$

$$E[\mathbf{X}_1] = E[\mathbf{X}_2] = \mathbf{0}, \quad V[y] = \mathbf{a}^T\mathbf{C}_{11}\mathbf{a} = 1, \quad V[z] = \mathbf{b}^T\mathbf{C}_{22}\mathbf{b} = 1 \quad \cdots(32)$$

ただし, $\mathbf{C}_{11}$, および, $\mathbf{C}_{22}$ は, $\mathbf{X}_1$, および, $\mathbf{X}_2$ の分散共分散行列, また, $\mathbf{C}_{12} = \mathbf{C}_{21}{}^T$ は, $\mathbf{X}_1$ と $\mathbf{X}_2$ の分散共分散行列とする.

y と z の相関係数 r_{yz} は, 上記の条件（y と z の分散が 1）より,

$$r_{yz} = \mathbf{a}^T\mathbf{C}_{12}\mathbf{b} \quad \cdots(33)$$

となる.

結局, y と z の分散が 1 であるという条件の下で, 上式を最大にする **a**, **b** を求めることになる. Lagrange の未定乗数法により, 以下の式を最大にする.

$f = \mathbf{a}^T\mathbf{C}_{12}\mathbf{b} - 0.5\ \lambda\ (\mathbf{a}^T\mathbf{C}_{11}\mathbf{a} - 1) - 0.5\ \mu\ (\mathbf{b}^T\mathbf{C}_{22}\mathbf{b} - 1)$

上式を, **a**, **b** で偏微分して 0 とおくと,

$$\frac{\partial f}{\partial a} = \mathbf{C}_{12}b - \lambda\mathbf{C}_{11}a = 0 \quad \cdots(34)$$

$$\frac{\partial f}{\partial b} = \mathbf{C}_{21}a - \mu\mathbf{C}_{22}b = 0 \quad \cdots(35)$$

となる. これらを変形して2つの式を加えると,

$$(\mathbf{C}_{11}^{-1}\mathbf{C}_{12}\mathbf{C}_{22}^{-1}\mathbf{C}_{21}\mathbf{a} - \lambda\mu\,\mathbf{I})\,\mathbf{a} = \mathbf{0} \quad \cdots(36)$$

となる.

上式より, $\lambda\mu$ は, $\mathbf{C}_{11}^{-1}\mathbf{C}_{12}\mathbf{C}_{22}^{-1}\mathbf{C}_{21}$ の固有値であり, $\mathbf{a}$ は, 対応する固有ベクトルとなる.

次に, $\lambda\mu$ について考える. (34) 式より、

$$\mathbf{a}^{T}\mathbf{C}_{12}\mathbf{b} = r_{yz} = \lambda \quad \cdots(37)$$

となり、また (35) 式より、

$$\mathbf{a}^{T}\mathbf{C}_{12}\mathbf{b} - \mu\,\mathbf{b}^{T}\mathbf{C}_{22}\mathbf{b} = 0 \quad \cdots(38)$$

となる。これらより、$\lambda = \mu$ であり, $\lambda\mu = \lambda^2$ となる.

また, $\mathbf{b}$ は, 得られた固有値 λ^2, 固有ベクトル $\mathbf{a}$ と, (35) 式より, 以下のようにして計算できる.

$$\mathbf{b} = \mathbf{C}_{22}^{-1}\mathbf{C}_{21}\,\mathbf{a} / \lambda \quad \cdots(39)$$

以上の結果をまとめると,

$$(\mathbf{C}_{11}^{-1}\mathbf{C}_{12}\mathbf{C}_{22}^{-1}\mathbf{C}_{21} - \lambda^2\,\mathbf{I})\,\mathbf{a} = \mathbf{0} \quad \cdots(40)$$

(λ^2 は, $\mathbf{C}_{11}^{-1}\mathbf{C}_{12}\mathbf{C}_{22}^{-1}\mathbf{C}_{21}$ の固有値, $\mathbf{a}$ は, 対応する固有ベクトル)

$$\mathbf{b} = \mathbf{C}_{22}^{-1}\mathbf{C}_{21}\,\mathbf{a} / \lambda \quad \cdots(41)$$

$$r_{yz} = \lambda \quad \cdots(42)$$

となる.

以上のようにして得られた変量 y, z を正準変量, また, その各係数を正準相関係数と呼ぶ. 絶対値が最も大きな固有値に対応する正準変量, 正準相関係数を第1正準変量, 第1正準相関係数, また, 2番目に大きな固有値に対応するものを第2正準変量, 第2正準相関係数, ・・・と呼ぶ.

本研究では, 楽曲の特徴量の第1正準変量, 第 2 正準変量を用いて 2 次元画像として表した.

４．実験

4ー1　識別関数の構成と感性情報の可視化

4ー1ー1　実験の目的

楽曲の特徴と楽曲を聴いた時に感じる印象から SVM により識別関数を構成し，楽曲の特徴と楽曲を聴いた時に感じる印象との関係を２次元画像として可視化する．

4ー1ー2　入力データについて

・楽曲の特徴量について

本研究では楽曲の特徴量としてコードとメロディの音階の構成比率を用いた．調による違いを無くすために，一つ一つのメロディの音階について，スケールと呼ばれるその楽曲の基となる音から何度離れているかを求め，その構成率を用いた．スケールと同じ音ならば０度差，半音高い音なら１度差となる．オクターブによる違いは無視するため，音階は12音階あり，スケール音からの差は最大11度差となる．また，コードについても同じ方法を用いた．

例えば，メロディの音階でスケールが G（ソ）の楽曲ならば，ソは0度差，ソ＃は1度差，ラは2度差，…，ファ＃は11度となる．また，コードの場合なら，スケールが G の楽曲の場合，G は0度差，A は2度差，…となる．

また，この楽曲の特徴量は24次元のデータとなるが，特徴量を２次元画像として可視化するために24次元から2次元にしなければならない．そのために24次元の特徴量を主成分分析によって9次元まで減らし，さらに正準相関分析によって得られた第1正準変量，第２正準変量を用いて2次元のデータとした．主成分による次元数を9次元としたのはデータの寄与率（元のデータと比べどのくらいの情報量を含んでいるか）を95％以上とするためである．

・感性情報について

特徴量を抽出した楽曲を実際に被験者に聴いてもらい，その曲を好きだと感じたかそうでないかのアンケートをとる．好きだと感じた場合は0とし，そうでなければ1とし，このデータを感性情報とした．

4ー1ー3　実験の手順

(1)楽曲の特徴量として，コードとメロディの音階の構成比率を抽出する．
(2)楽曲を実際に被験者に聴いてもらい，その曲を好きだと感じたかそうでないかのアンケートをとる．好きならば0，そうでなければ1とする．このアンケート結果を感性情報とする．
(3)主成分分析を用いて24次元の特徴量の次元を減らす．
(4)正準相関分析を用いて特徴量と感性情報を結びつける．
(5)特徴量に対する第１正準変量，第２正準変量を用いて SVM により識別関数を構成する．
(6)構成した識別関数と楽曲の特徴量を２次元画像として可視化する．

4ー1ー4　実験環境

以下の条件で実験を行った．
被験者…2 人
楽曲数…30曲

また，特徴量の寄与率は95％以上とするために，主成分分析によって24次元のデータを9次元まで減らして実験を行った．

4ー1ー5　作成したプログラムソース（一部抜粋）

※実際に画像を表示するプログラムは省略

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
double siguma2; //分散
double C;  //上限値
int l;  //データ数
double f;                 //マージン
double a[100];        //識別関数のパラメータ
const double TYNY_NUM=0.00001;  //収束値

//入力データの型
struct data
{
        double x1; //x 座標
        double x2; //y 座標
        int y;     //クラス値
};

data x[100];  //入力データ

int main( int argc, char*argv[])
{
        int seikai=0,i,j;

        FILE *fp;

        void chouheimen(data x[],double a[]);
        double K(data x,data z);
        int get_class( double x1, double x2 );

        //ファイルからデータの座標値とクラス値を入力
        fp=fopen(argv[1],"r");
        i=0;
        while(!feof(fp)&& !ferror(fp))
        {
                fscanf(fp,"%lf,%lf,%d",&x[i].x1,&x[i].x2,&x[i].y);
                i++;
        }
        l=i-2;
        fclose(fp);

        //クラスを1と−1に分類
        for(i=0;i<l;i++)
                if(x[i].y!=1)
                        x[i].y=-1;

        //分散と上限値を決定
        siguma2=atof(argv[2]);
        C=atof(argv[3]);
```

```
	//識別関数のパラメータ a を決定
	chouheimen(x,a);

	for(i=0;i<l;i++)
	{
		//識別関数を求める
		f=0;
		for(j=0;j<l;j++)
			f+=x[j].y*a[j]*K(x[j],x[i]);

		//正解数をカウント
		if(f>=0)
		{
			if(x[i].y==1)
				seikai+=1;
		}
		else
		{
			if(x[i].y==-1)
				seikai+=1;
		}



	}

	//正解率を表示
	printf("正解率は　%d パーセント\n",seikai*100/l);

	show_graph( argc, argv );

	return 0;
}




//識別関数のパラメータを求める
void chouheimen(data x[],double a[])
{
	double n[100]		//学習率
		,s;
	int i,j;
	double current_w=0,pre_w=0;
	double K(data x,data z);	//ガウスカーネル

	//学習率を決定
	for(i=1;i<=l;i++)
	{
		a[i]=0;
		n[i]=0.2/K(x[i],x[i]);
	}
```

```
	//パラメータaを決定
	for(;;)
	{
		for(i=0;i<l;i++)
		{
			s=0;
			for(j=0;j<l;j++)
				s=s+a[j]*x[j].y*K(x[i],x[j]);
			a[i]=a[i]+n[i]*(1-(x[i].y*s));
			if(a[i]<0)a[i]=0;		//a>0(非負条件)
			else if(a[i]>C)a[i]=C;		//a<C(上限値)
		}

		//収束条件
		double t1=0;
		for(i=0;i<l;i++)
			t1+=a[i];
		double t2=0;
		for(i=0;i<l;i++)
		{
			for(j=0;j<l;j++)
				t2+=x[i].y*x[j].y*a[i]*a[j]*K(x[i],x[j]);
		}
		current_w=t1-t2/2;
		if(fabs(current_w-pre_w)<TYNY_NUM)break;
		pre_w=current_w;
	}


}

//ガウスカーネル
double K(data x,data z)
{
	double A=-( (x.x1-z.x1)*(x.x1-z.x1) + (x.x2-z.x2)*(x.x2-z.x2) ) /siguma2;
	return exp(A);
}

//データのクラス値を返す関数(画像表示に用いる)
int get_class( double x1, double x2 )
{
	data xx;
	int j;
	xx.x1=x1-idou_x;
	xx.x2=x2-idou_y;
	f=0;
	for(j=0;j<l;j++)
		f+=x[j].y*a[j]*K(x[j],xx);
	if(f>=0)
		return 1;
	else
		return 0;

}
```

```
//データを描画領域にあわせる（画像表示に用いる）
#define idou_x 0.6
#define idou_y 0.5
int get_inputdata( int n, double& xx, double& yy )
{
	int c;
	if( (n < 0)  ||  (n  >  l) ) return -1;
	xx=x[n].x1+idou_x;
	yy=x[n].x2+idou_y;
	if(x[n].y==1)
		c=1;
	else
		c=0;

	return c;
}
```

4―1―6　実験結果

横軸，縦軸にそれぞれ特徴量と感性情報の第 1 正準変量，第 2 正準変量を用いてそれぞれの楽曲の特徴量と，SVM によって構成された好きだと感じた曲とそうでない曲を識別する識別関数を 2 次元画像として図6に示す．

図中の0の点は被験者が好きだと感じた曲の特徴量，1の点はそうでない曲の特徴量を示している．また，灰色と白色の境界が SVM によって構成された識別関数を表している．

(1)被験者1　　(2)被験者2

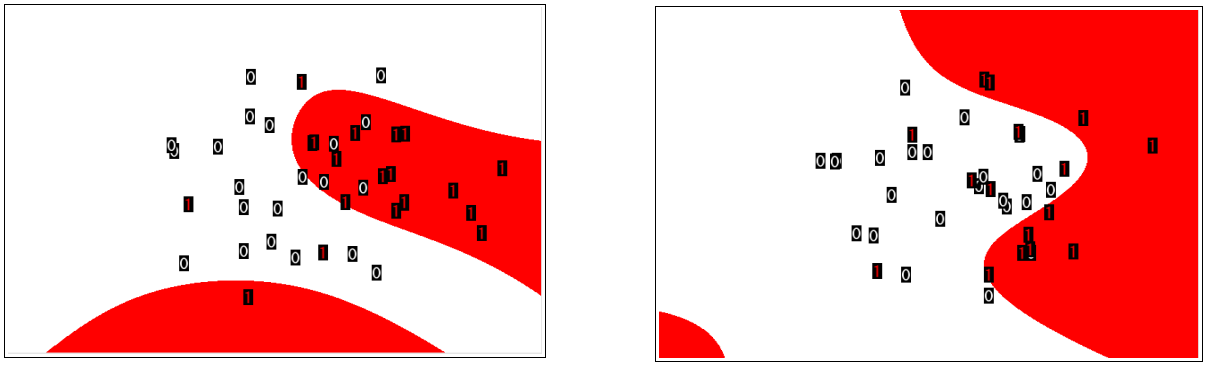

図6　SVM を用いた楽曲の特徴空間

4―1―7　考察

全体的に楽曲の特徴量はクラスごとにまとまっているが，1曲だけ他の同じクラスの楽曲と離れた特徴量（外れ値）があると，その 1 曲を識別するだけの識別関数が構成されてしまう．このような外れ値はできるだけ取り除くようにすればさらに識別の精度は上がると思われる．

4―2　識別関数を用いた感性情報の識別

4―2―1　実験の目的

楽曲から特徴量を抽出し識別関数を構成する. 構成した識別関数を用いて他の楽曲について識別を行う.

4―2―2　実験手順

(1)特徴量として, 40曲の楽曲からコードとメロディの音階の構成比率を抽出する（特徴量は24次元となる).
(2)被験者に特徴量を抽出した40曲の楽曲を聴いてもらい感性情報のアンケートをとる. 0なら好きな曲, 1ならそうでない曲とする.
(3)40曲中30曲の特徴量と感性情報を用いて識別関数を構成する. なお, 可視化はしないため, 主成分分析, 正準相関分析は行わず24次元の特徴量のまま識別関数を構成する.
(4)識別関数の構成には用いなかった残りの10曲の楽曲に対して構成した識別関数によりそれぞれの楽曲の特徴量から感性情報を識別する.
(5)識別結果が(2)で実際にアンケートをとった結果とどのくらい一致しているか確認する.

4―2―3　実験環境

以下の条件で実験を行った.
被験者…6人
楽曲数…40曲（30曲を識別関数の構成に用い, 10曲を識別する）

4―2―4　作成したプログラムソース

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
int l,ll;          //データ数
double f;          //マージン
double a[100];     //識別関数のパラメータ
double siguma2;    //分散
double C;          //上限値
const int jigen=24;          //データの次元数

const double TYNY_NUM=0.00001;     //収束値
struct data
{
        double x[256];
        int y;
};

data input[100];

int main( int argc, char*argv[])
{
        int seikai=0,i,j,k,hanbetu,w1,w2;

        FILE *fp1,*fp2;
```

```
	void chouheimen(data input[],double a[]);
	double K(data x,data z);

	//ファイルからデータの座標値とクラス値を入力
	fp1=fopen(argv[1],"r");
	fp2=fopen(argv[4],"r");
	k=0;
	while(!feof(fp1)&& !ferror(fp1))
	{
		for(i=0;i<jigen;i++)
		{
			fscanf(fp1,"%lf,",&input[k].x[i]);
		}
		fscanf(fp2,"%d,%d",&w1,&w2);
		if(w1==1)
			input[k].y=1;
		else
			input[k].y=-1;
		k++;
	}
	l=k-1;
	fclose(fp1);
	fclose(fp2);

	//分散と上限値を決定
	siguma2=atof(argv[2]);
	C=atof(argv[3]);

/*----------40曲中30曲で識別関数を構成----------*/
	ll=l;
	l=30;
	//識別関数のパラメータ a を決定
	chouheimen(input,a);


	for(i=0;i<l;i++)
	{
		//識別関数を求める
		f=0;
		for(j=0;j<l;j++)
			f+=input[j].y*a[j]*K(input[j],input[i]);
		//正解数をカウント
		if(f>=0)
		{
			if(input[i].y==1)
				seikai+=1;
		}
		else
		{
			if(input[i].y==-1)
				seikai+=1;
		}
	}

	//正解率を表示
```

```
	printf("正解率は　%d パーセント\n",seikai*100/l);
/*----------40曲中30曲で識別関数を構成----------*/



/*----------残りの10曲の感性情報を識別する----------*/
	int sikibetu=0;
	for(i=30;i<40;i++)
	{
		f=0;
		for(j=0;j<l;j++)
		{
			f+=input[j].y*a[j]*K(input[j],input[i]);
		}
		if(f>=0)
		{
			if(input[i].y==1)sikibetu++;
		}
		else
		{
			if(input[i].y==-1)sikibetu++;
		}

	}
	printf("正解率は　%d パーセント\n",sikibetu*100/10);
/*----------残りの10曲の感性情報を識別する----------*/



	return 0;
}



//識別関数のパラメータを求める
void chouheimen(data input[],double a[])
{
	double n[100]		//学習率
		,s;
	int i,j;
	double current_w=0,pre_w=0;
	double K(data x,data z);	//ガウスカーネル


	//学習率を決定
	for(i=0;i<l;++i)
	{

		a[i]=0;
		n[i]=0.2/K(input[i],input[i]);
	}

	//パラメータ a を決定
	for(;;)
```

```
	{
		for(i=0;i<l;i++)
		{
			s=0;
			for(j=0;j<l;j++)
			{
				s=s+a[j]*input[j].y*K(input[i],input[j]);
			}
			a[i]=a[i]+n[i]*(1-(input[i].y*s));
			if(a[i]<0)a[i]=0;			//a>0(非負条件)
			else if(a[i]>C)a[i]=C;			//a<C(上限値)
		}

		//収束条件
		double t1=0;
		for(i=0;i<l;i++)
		{
			t1+=a[i];
		}
		double t2=0;
		for(i=0;i<l;i++)
		{
			for(j=0;j<l;j++)
			{
				t2+=input[i].y*input[j].y*a[i]*a[j]*K(input[i],input[j]);
			}
		}
		current_w=t1-t2/2;
		if(fabs(current_w-pre_w)<TYNY_NUM)break;
		pre_w=current_w;
	}

}

//ガウスカーネル
double K(data x,data z)
{
	int i;
	double s=0;
	for(i=0;i<jigen;i++)
		s+=(x.x[i]-z.x[i])*(x.x[i]-z.x[i]);
	return exp(-s /siguma2);
}
```

4―2―5　実験結果

識別結果と実際のアンケート結果が10曲中どのくらい一致したかを正解率として表1で示す.

	正解率
被験者1	60%
被験者2	70%
被験者3	50%
被験者4	30%
被験者5	40%
被験者6	40%

表1　識別結果の正解率

4—2—6　考察

感性情報の識別は精度に個人差があり，全体的に低い正解率となった．これを改善するには，今回用いた楽曲の特徴量に加えてリズムやテンポなどの特徴量を採り入れたり，外れ値の曲を除いて識別関数を構成すればさらに精度が上がると思われる．

5．まとめ

本研究では，個人の嗜好に基づいた楽曲推薦システムの足掛かりとなるよう，楽曲の特徴と楽曲を聴いた時に感じる印象との関係を調べるため，SVM により楽曲の特徴量と感性情報を識別する識別関数を構成し，楽曲の特徴量と感性情報を 2 次元画像として表示した．さらに，構成した識別関数を利用し，感性情報が未知の楽曲に対して感性情報の識別を行った．

今後の課題として，楽曲の特徴量を，今回用いたコードとメロディの音階の構成比率の他にリズムやテンポなどの特徴量を採り入れ，外れ値の楽曲は取り除いて識別するようにすればさらに精度のよい識別が可能になると思われる．

これを発展，応用させれば個人的な嗜好に基づいた楽曲推薦システムに役立てることができると思われる．

7.参考文献

[1] Nello Cristianini,John Shawe-Taylor 著, 大北剛 訳: サポートベクターマシン入門
[2]田中豊,垂水共之,脇本和昌 編:パソコン統計解析ハンドブック2多変量解析編
[3]武井亨,エドワード・パック・エンフィ,東海林智也:感性情報に基づいた楽曲の特徴空間の可視化(info 北海道シンポジウム 2006）
[4]栗田多喜夫 :サポートベクターマシン入門　http://www.neurosci.aist.go.jp/~kurita/lecture/svm/svm.html
[5]Tomoyuki Tarumi :正準相関分析　http://case.f7.ems.okayama-u.ac.jp/statedu/hbw2-book/node80.html
[6]北川裕:絶対わかる!コード理論
[7]飯塚広:日本唱歌童謡集
[8]御池鮎樹:裏口からの MIDI 入門
[9]水谷友香,キッシー岸田:CD で覚えるやさしい楽譜の読み方
[10]市川裕也,田村智嗣,速水悟:印象語のグループ化を用いた楽曲推薦システム（The 20th Annual Conference of the Japanese Society for Artificial Intelligence,2006）
[11]梶克彦,平田圭二,長尾確:状況と嗜好に関するアノテーションに基づくオンライン楽曲推薦システム（音楽情報科学研究会(SIGMUS) ）
[12]黒瀬崇弘,梶川嘉延,野村康雄:感性情報を用いた楽曲推薦システム
[13]岐阜大学速水研究室:音楽推薦システム　http://hym.info.gifu-u.ac.jp/recomm.html
[14]ユーザの気分に合った曲の自動選曲・推薦法　http://www.hit.dj.kit.ac.jp/research/music/
[15]渡辺澄夫:学習システムの理論と実現