二次のメンバーシップ関数による超直方体領域を持つ ファジィクラシファイアの汎化能力の向上*

阿部 重夫[†]・Ruck THAWONMAS[‡]

Improvement of Generalization Ability of a Fuzzy Classifier with Hyperbox Regions by Quadratic Membership Functions*

Shigeo ABE[†] and Ruck THAWONMAS[‡]

In this paper we discuss a fuzzy classifier with hyperbox regions which has quadratic membership functions. First we define the classifier with quadratic membership functions and then discuss the training method that maximizes the recognition rate by counting the net increase in the recognition rate when the slope and the center of a membership function are separately changed. Finally, we demonstrate the recognition improvement by quadratic membership functions for four data sets.

1. はじめに

多層ニューラルネットと同等の学習機能を持つファジィクラシファイアの開発が進められている [1-9]. これらのクラシファイアは入力領域の近似の仕方から, 1) 入力軸に平行な面からなる超直方体領域を持つファジィクラシファイア [2-4], 2) 超多面体領域を持つファジィクラシファイア [5], および 3) 楕円領域を持つファジィクラシファイアに分類される [6-9]. 参考文献 [3] のクラシファイアは, データの分布が入力軸に平行なときは高い汎化能力を示すが, データの分布が入力軸に平行でないときは汎化能力が低下する. これに対して参考文献 [5] のクラシファイアでは, 入力の線形和の超平面で領域近似するため, 参考文献 [3] の問題は生じないが, 学習済みの多層ニューラルネットを用いて領域近似を行うために学習が遅いという問題がある. これに対して参考文献 [8,9] のクラシファイアは高速に学習でき, 入力連続量であるときは汎化能力も高いが, 離散量を含むときは汎化能力が低下する.

ここで超直方体領域を持つファジィクラシファイアのう

ち参考文献 [2] では, 同一クラスの超直方体の重なりは許すが, 異なるクラスの超直方体の重なりは許さない. これに対して参考文献 [3] では, データの存在を許す活性超直方体と存在を許さない禁止超直方体とを定義して, 活性超直方体が重なった部分を禁止超直方体として定義している. 参考文献 [2,3] では (活性) 超直方体の内部で成立度が 1 となるメンバーシップ関数を用いているが, 参考文献 [4] では, 角錐のメンバーシップ関数を用い, 異なるクラスの超直方体の重なりを許し, メンバーシップ関数の位置と傾きを解析的に調整することにより, 認識率を向上する方法をとっている.

本論文では参考文献 [4] に基づいて, 一次 (角錐) のメンバーシップ関数を二次のメンバーシップ関数に置き換えたファジィクラシファイアを提案する. さらに, 二次のメンバーシップ関数でも一次の場合と同様に位置と中心のチューニングが解析的に行えることを示す. すなわち, 多次元のメンバーシップ関数を二次関数で表現される一次元のメンバーシップ関数の平均値で定義すると, メンバーシップ関数の中心をある軸方向に動かしたときに, 影響を受けるのはその軸に対する一次元のメンバーシップ関数のみである. これより, 中心をある軸の方向に移動させたときに, どの位置で正しく認識されたデータが誤認識し, またどの位置で誤認識していたデータが正しく認識できるかを解析的に求めることができるために, 中心をどの位置に移動すれば, 認識率が最大になるかを求めることができる. これより, 各ファジィルールを

* 原稿受付 1995 年 8 月 1 日

[†] 神戸大学 大学院 自然科学研究科 Graduate School of Science and Technology, Kobe University; Rokkodai, Nada ward, Kobe city, Hyogo 657-8501, JAPAN

[‡] 高知工科大学 情報システム工学科 Department of Information Systems Engineering, Kochi University of Technology; Tosayamada-cho, Kochi city, Kochi 782-8502, JAPAN

Key Words: pattern recognition, fuzzy classifiers, quadratic membership functions.

入力軸に沿って認識率が最大になるようにチューニングする。メンバーシップ関数の傾きも同様に行うことができる。

以下では、2. で二次のメンバーシップ関数をもつファジクラシファイアを提案し、3. でファジールールの生成法を述べ、ついで4. でクラスに属するデータをクラスタリングする方法を述べる。5., 6. でおのおのメンバーシップ関数の中心と位置のチューニング法を述べ、7. で四つのデータを用いて、一次のメンバーシップ関数よりも二次のメンバーシップ関数の方が教師データ、テストデータの認識率ともに向上することを示す。

2. メンバーシップ関数

参考文献 [2,3] では超直方体の内部で1となるメンバーシップ関数を用いられているが、このメンバーシップ関数をチューニングするときは、クラスの異なる超直方体が重なったときに、重なりを解消する処理を行う必要がある。すなわち、傾きと位置のほかに超直方体そのものもチューニングする必要があり、チューニングが複雑になる。このため、角錐状のメンバーシップ関数を用いることにする。

m 次元入力空間においてクラス i ($i=1, \dots, n$) の j 番目の超直方体を、 A_{ij} としてつぎのように定義する。

$$A_{ij} = \{x | v_{ijk} \leq x_k \leq V_{ijk}, k=1, \dots, m\}, j=1, \dots, (1)$$

ここで x_k はベクトル x の k 番目の要素で、 v_{ijk} と V_{ijk} はおのおの A_{ij} の x_k 軸の最小値と最大値である。

このとき A_{ij} の中心 c_{ij} は次式で与えられる。

$$c_{ij} = \left(\frac{V_{ij1} + v_{ij1}}{2}, \dots, \frac{V_{ijm} + v_{ijm}}{2} \right) \quad (2)$$

ここで A_{ij} の中心 c_{ij} の成立度を1とし、 c_{ij} を頂点とする角錐状のメンバーシップ関数を定義する。このために入力 x に対して一次元の重み付きチューニング距離 $h_{ijk}(x)$ ($k=1, \dots, m$) を次式で定義する。

$$h_{ijk}(x)^s = \frac{d_{ijk}(x)^s}{\alpha_{ij}} \quad (3)$$

ただし s はメンバーシップ関数が一次のときは1で二次のときは2とする。また α_{ij} はメンバーシップ関数の傾きを決めるチューニングパラメータで初期値を1とする。 $d_{ijk}(x)$ は一次元の重み付き距離で次式で与えられる。

$$d_{ijk}(x) = \frac{|c_{ijk} - x_k|}{w_{ijk}} \quad (4)$$

ただし

$$w_{ijk} = \frac{V_{ijk} - v_{ijk}}{2} \quad (5)$$

である。ここで0での除算を避けるために ε を小さな値として $w_{ijk} < \varepsilon$ のとき、(4) 式において w_{ijk} を ε で置きかえる。

つぎに $x_k \geq c_{ijk}$ および $x_k < c_{ijk}$ でメンバーシップ関数の変曲点が生じないようにするために成立度が負になることも許すとする。すなわち

$$m_{ijk}(x) = 1 - h_{ijk}(x)^s \quad (6)$$

入力 x に対する多次元のメンバーシップ関数 $m_{ij}(x)$ を (6) 式で与えられる一次元のメンバーシップ関数の平均値で求めることとする。すなわち

$$m_{ij}(x) = \frac{1}{m} \sum_{k=1}^m m_{ijk}(x) \quad (7)$$

(6) 式を用いると (7) 式は次式のようにになる。

$$m_{ij}(x) = 1 - h_{ij}(x)^s \quad (8)$$

ここで $h_{ij}(x)$ はチューニングされた重み付き距離で次式で与えられる。

$$h_{ij}(x)^s = \frac{1}{m} \sum_{k=1}^m h_{ijk}(x)^s \quad (9)$$

なお、(8) 式のかわりに、以下に示す負の値をとらないメンバーシップ関数を用いてもよい。すなわち (6) 式のかわりにつぎの一次元のメンバーシップ関数を定義する。

$$m_{ijk}(x) = \exp(-h_{ijk}(x)^s) \quad (10)$$

(10) 式の積で多次元のメンバーシップ関数 $m_{ij}(x)$ を定義すると次式のようにになる。

$$m_{ij}(x) = \prod_{k=1}^m \exp(-h_{ijk}(x)^s) \quad (11)$$

$$= \exp\left(-\sum_{k=1}^m h_{ijk}(x)^s\right) \quad (12)$$

3. ファジールールの生成

ファジールールの生成は、まずクラス i に属するデータを4. で示すクラスタリング法によりクラス $ij, j=1, \dots$ に分割して、各クラス ij に属する教師データを用いて次のファジールールを定義する。

$$R_{ij}: \text{ If } x \text{ is } c_{ij} \text{ then } x \text{ belongs to class } i \quad (13)$$

次にメンバーシップ関数の傾きおよび位置を、教師データに対する認識率の改善がなくなるまで順次調整する。以下に学習の手順を示す。

ステップ 1 おのおののクラス $ij (i=1, \dots, n, j=1, \dots)$ に属するデータを用いて (13) 式 of ファジィルールを定義する。

ステップ 2 6. に示す手順でチューニングパラメータ $\alpha_{ij} (i=1, \dots, n, j=1, \dots)$ を順次増加あるいは減少させて、各クラスについて新たな誤認識の最大数を $l_M - 1$ として認識率が最大になるようにする。

ステップ 3 5. に示す手順で中心 $c_{ij} (i=1, \dots, m, j=1, \dots)$ を順次右あるいは左に移動させて、各クラスにおける新たな誤認識の最大数を $l_{Mc} - 1$ として認識率が最大になるようにする。

ステップ 4 ステップ 2, 3 の処理を繰り返し、認識率の改善がなくなった時点で処理を終了する。

4. クラスタリング

データをクラスタリングする方法は種々の方法が提案されているが、ここでは参考文献 [2] の動的にファジィルールを抽出する方法をベースにして、教師データを最初から順次一個づつ処理して、超直方体の生成、拡張により各クラスの教師データのクラスタリングを行う。すなわち、あるデータを処理するときは、そのクラスに属する拡張可能な超直方体が存在しないときにそのデータのみを含む超直方体を生成する。そうでないときは、そのデータを含むように超直方体を拡張する。超直方体の生成が終了すると、超直方体に属するデータが一つのクラスを構成することになる。なお、クラスタリングにおいて超直方体の内部にあるか否かを判定するために角錐台状のメンバーシップ関数を用いる。クラスタリングではクラス間の重なりは解消は行わないので、角錐台状のメンバーシップ関数を用いても問題は生じない。

クラス i の j 番目の超直方体 A_{ij} に角錐台状のメンバーシップ関数を定義するために、まず k 番目の入力に対する一次元のメンバーシップ関数を次式で定義する。

$$m_{ijk}(\mathbf{x}) = \begin{cases} 1 - (x_k - V_{ij}) & x_k \geq V_{ij} \\ 1 & V_{ij} > x_k \geq v_{ij} \\ 1 - (v_{ij} - x_k) & v_{ij} > x_k \end{cases} \quad (14)$$

これを用いて m 次元のメンバーシップ関数を (7) 式で与えられるとする。ここで超直方体の一つの頂点から出る辺の和は $m\theta$ 以下であるという制限を与えておく。

教師データの最初のデータについては、そのデータがクラス i に属するとするとそのデータのみを含む超直方体 A_{i1} を定義する。それ以降のクラス i のデータについてはすでに定義された超直方体 $A_{ij} (j=1, \dots)$ がそのデータを含めるように拡張可能かを調べる。すなわち次式が

成り立てば拡張可能である。

$$m\theta \geq \sum_{k=1}^m (\max(V_{ijk}, x_k) - \min(v_{ijk}, x_k)) \quad (15)$$

もし拡張可能な超直方体が存在すれば、(7), (14) 式によりこれらの超直方体に対する成立度を計算し、 A_{ij} に対する成立度が最も大きいとすると、次式により A_{ij} を拡張する。

$$\begin{aligned} v_{ijk} &\leftarrow \min(v_{ijk}, x_k) \\ V_{ijk} &\leftarrow \max(V_{ijk}, x_k) \end{aligned} \quad k=1, \dots, m \quad (16)$$

もし拡張できる超直方体がないときはつぎのようにそのデータのみを含む超直方体を定義する。

$$v_{ijk} = V_{ijk} = x_k \quad k=1, \dots, m \quad (17)$$

上記の方法では教師データを一度処理すればクラスタリングができるが、処理順序によりクラスタリングの結果が異なる可能性がある。

5. 中心のチューニング

5.1 考え方

以前正しく認識していたデータが誤認識することも許して、超直方体 A_{ij} を移動して教師データの認識率の向上を図る。ここで議論を簡単にするために、一度の移動では A_{ij} を一つの軸方向に移動することにする。中心のチューニングは A_{11} の 1 番目の入力から順次 m 番目の入力について行い、その後残りの超直方体の位置のチューニングを行う。これを 1 回の位置のチューニングとよぶことにする。位置のチューニングは教師データの認識率が 100% になるか教師データの認識率の向上がなくなるまで行う。この方法では処理順序により結果が変わる可能性があるが、以下に述べる手順にしたがえば十分な認識性能が得られることを 7. で示す。

ここで中心 c_{ij} を k 番目の入力軸で移動することを考える。このとき正しく認識されたクラス i に属するデータを誤認識させないで c_{ijk} を右または左に移動させることが可能である。もし c_{ijk} を x_k まで動かすと、 $m_{ij}(\mathbf{x})$ は c_{ijk} が x_k に達するまで増加する。そして $m_{ij}(\mathbf{x})$ は c_{ijk} が x_k から離れてゆくと減少する。したがってクラス i に属し正しく認識されたデータ $\mathbf{x}$ に対して $\mathbf{x}$ が正しく認識される c_{ijk} の範囲 $(I_{ijk}(\mathbf{x}), D_{ijk}(\mathbf{x}))$ が存在する。同様にクラス i に属しない、正しく認識されたデータ $\mathbf{x}$ に対して $\mathbf{x}$ が正しく認識される c_{ijk} の範囲 $(-\infty, D_{ijk}(\mathbf{x})), (I_{ijk}(\mathbf{x}), \infty)$ が存在する。

したがって、正しく認識されたデータに誤認識が生じない上限 $D_{ijk}(1) = \min D_{ijk}(\mathbf{x}) (\geq c_{ijk})$ および下限

$I_{ijk}(1) = \max I_{ijk}(\mathbf{x}) (< c_{ijk})$ が存在する. 同様に正しく認識されたデータのうちに $l-1$ 個のデータが誤認識する上限 $D_{ijk}(l)$ および下限 $I_{ijk}(l)$ が存在する.

誤認識しているデータに関しては, c_{ijk} を右あるいは左に動かすことにより, 誤認識が解消する場合がある. すなわち誤認識しているクラス i のデータ $\mathbf{x}$ を正しく認識させる c_{ijk} の区間 $(I_{ijk}(\mathbf{x}), D_{ijk}(\mathbf{x}))$ が存在する. さらに, クラス i に属さない, 誤認識したデータ $\mathbf{x}$ に対して, ある条件のもとで, 正しく認識させる c_{ijk} の区間 $(I_{ijk}(\mathbf{x}), \infty)$ および $(-\infty, D_{ijk}(\mathbf{x}))$ が存在しう.

したがって正しく認識されたデータのうちに $l-1$ 個のデータが誤認識する c_{ijk} の上限 $D_{ijk}(l)$ および下限 $I_{ijk}(l)$ を用いて, 最も高い認識率を実現する c_{ijk} の区間を決定できる.

以下では正しく認識されたデータのうちに $l-1$ 個のデータが誤認識する c_{ijk} の上限 $D_{ijk}(l)$ および下限 $I_{ijk}(l)$ の決定法, 誤認識したデータを正しく認識する区間の計算法, および $l_{Mc}-1$ 個の誤認識を許しながら, 最大の認識率を実現する方法について述べる.

5.2 c_{ijk} の上下限

$l-1 (\geq 0)$ 個の正しく認識されたデータを誤認識させる c_{ijk} の上限 $D_{ijk}(l)$ および下限 $I_{ijk}(l)$ を求めるために, 教師データをファジイルール $\{R_{ij}\}$ を用いて正しく認識された教師データの集合 X と誤認識された教師データの集合 Y とに分割する. つぎにクラス i に属する入力データ $\mathbf{x} (\in X)$ で次式を満たすものを求める.

$$h_{ij}(\mathbf{x}) \leq \min_{k \neq j} h_{ik}(\mathbf{x}) \quad (18)$$

もし (18) 式が成立しないときは, 成立度 $m_{ij}(\mathbf{x})$ は最小ではないから, c_{ijk} を変更しても $\mathbf{x}$ は正しく認識されたままである. もし $\mathbf{x}$ が次式を満たすとすると

$$h_{ij}(\mathbf{x}) < h_{op}(\mathbf{x}) = \min_{\substack{o \neq i, \\ q=1, \dots}} h_{oq}(\mathbf{x}) < \min_{k \neq j} h_{ik}(\mathbf{x}) \quad (19)$$

$\mathbf{x}$ を正しく認識させたままにする下限 $I_{ijk}(\mathbf{x})$ および上限 $D_{ijk}(\mathbf{x})$ が存在する.

上下限を計算するために, まず c_{ijk} が動いたときのチューニングされた重み付き距離の変化を計算する. 一次元のチューニングされた重み付き距離 $h_{ijl}(\mathbf{x}) (l=1, \dots, m)$ のうち, (4) 式より $h_{ijk}(\mathbf{x})$ のみが変化することが分かる. ここで $h_{ij}^k(\mathbf{x})$ を c_{ij} が k 番目の入力軸上で c'_{ijk} まで移動したときのファジイルール R_{ij} の入力 $\mathbf{x}$ に対するチューニングされた重み付き距離とする. このとき差 $\Delta h_{ij}^k(\mathbf{x}) = h_{ij}^k(\mathbf{x})^s - h_{ij}(\mathbf{x})^s$ は次式で与えられる.

$$\Delta h_{ij}^k(\mathbf{x})^s = h_{ij}^k(\mathbf{x})^s - h_{ij}(\mathbf{x})^s$$

$$= \frac{|c'_{ijk} - x_k|^s}{m\alpha_{ij}w_{ijk}^s} - \frac{|c_{ijk} - x_k|^s}{m\alpha_{ij}w_{ijk}^s} \quad (20)$$

したがって

$$h_{ij}^k(\mathbf{x})^s - h_{ij}(\mathbf{x})^s \geq - \frac{|c_{ijk} - x_k|^s}{m\alpha_{ij}w_{ijk}^s} \quad (21)$$

ここで等式は $c'_{ijk} = x_k$ のときに成立する.

上限あるいは下限において

$$h_{ij}^k(\mathbf{x})^s = \Delta h_{ij}^k(\mathbf{x})^s + h_{ij}(\mathbf{x})^s = h_{op}(\mathbf{x})^s \quad (22)$$

が成立する. (21) 式と (22) 式より,

$$h_{op}(\mathbf{x})^s - h_{ij}(\mathbf{x})^s \geq - \frac{|c_{ijk} - x_k|^s}{m\alpha_{ij}w_{ijk}^s} \quad (23)$$

となる. したがって $h_{op}(\mathbf{x}) - h_{ij}(\mathbf{x}) > 0$ より, 上下限は常に存在して次式で与えられる.

$$D_{ijk}(\mathbf{x}) = x_k + \text{sign}(h_{op}(\mathbf{x}) - h_{ij}(\mathbf{x}))a_k \quad (24)$$

$$I_{ijk}(\mathbf{x}) = x_k - \text{sign}(h_{op}(\mathbf{x}) - h_{ij}(\mathbf{x}))a_k \quad (25)$$

ここで

$$\text{sign}(x) = \begin{cases} 1 & x \geq 0, \\ 0 & x < 0, \end{cases} \quad (26)$$

$$a_k = (m\alpha_{ij}w_{ijk}^s (h_{op}(\mathbf{x})^s - h_{ij}(\mathbf{x})^s) + |c_{ijk} - x_k|^s)^{1/s} \quad (27)$$

である.

もし (19) 式が満たされないとき, すなわち

$$h_{ij}(\mathbf{x}) < \min_{k \neq j} h_{ik}(\mathbf{x}) < \min_{o \neq i, p=1, \dots} h_{op}(\mathbf{x}) \quad (28)$$

のとき, $\mathbf{x}$ が誤認識することなく c_{ijk} を移動できる.

つぎにクラス $o (\neq i)$ に属する $\mathbf{x} (\in X)$ を選び, クラス o が最小のチューニングされた重み付き距離 $h_{op}(\mathbf{x})$

$$h_{op} = \min_q h_{oq}(\mathbf{x}) \quad (29)$$

を持つとする.

チューニングされた重み付き距離 $h_{ij}(\mathbf{x})$ は $h_{op}(\mathbf{x})$ より大きいから, もし (23) 式が満たされると (24) 式の $D_{ijk}(\mathbf{x})$ と (25) 式の $I_{ijk}(\mathbf{x})$ が存在する.

正しく認識されたデータの誤認識がない下限 $I_{ijk}(1)$ は次式で定義される.

$$I_{ijk}(1) = \max_{\mathbf{x} \in X} I_{ijk}(\mathbf{x}) \quad (30)$$

ここで, 議論を簡単にするために, 異なる $\mathbf{x}$ に対して $I_{ijk}(\mathbf{x})$ が異なると仮定する. このとき (30) はひとつの $\mathbf{x}$ によって満たされる. 一つの正しく認識されるデータ

の誤認識を許す下限 $I_{ijk}(2)$ は, $I_{ijk}(\mathbf{x})$ の二番目の最小値であり, 次式で与えられる.

$$I_{ijk}(2) = \max_{\mathbf{x} \in X, I_{ijk}(\mathbf{x}) \neq I_{ijk}(1)} I_{ijk}(\mathbf{x}) \quad (31)$$

一般に $I_{ijk}(l)$ を次式で定義する.

$$I_{ijk}(l) = \max_{\substack{\mathbf{x} \in X, \\ I_{ijk}(\mathbf{x}) \neq I_{ijk}(1), \dots, I_{ijk}(l-1)}} I_{ijk}(\mathbf{x}) \quad (32)$$

c_{ijk} が $I_{ijk}(l)$ まで移動したときに, c_{ijk} が $I_{ijk}(o)$ ($1 \leq o < l$) まで移動したときに誤認識したデータが正しく認識される可能性があるが, $I_{ijk}(l)$ の計算においてこのことは考慮に入れないことにする.

つぎに正しく認識されたデータの誤認識を生じない上限 $D_{ijk}(1)$ は次式で与えられる.

$$D_{ijk}(1) = \min_{\mathbf{x} \in X} D_{ijk}(\mathbf{x}) \quad (33)$$

ここでも異なる $\mathbf{x}$ に対して $D_{ijk}(\mathbf{x})$ の値が異なるとすると, (33) 式が一つの $\mathbf{x}$ に対して満たされる. 同様に一つのデータの誤認識を許す上限 $D_{ijk}(2)$ は $D_{ijk}(\mathbf{x})$ の二番目の最小値で次式で与えられる.

$$D_{ijk}(2) = \min_{\mathbf{x} \in X, D_{ijk}(\mathbf{x}) \neq D_{ijk}(1)} D_{ijk}(\mathbf{x}) \quad (34)$$

一般に $D_{ijk}(l)$ を次式で定義する.

$$D_{ijk}(l) = \min_{\substack{\mathbf{x} \in X, \\ D_{ijk}(\mathbf{x}) \neq D_{ijk}(1), \dots, D_{ijk}(l-1)}} D_{ijk}(\mathbf{x}) \quad (35)$$

c_{ijk} が $D_{ijk}(l)$ まで移動したときに, c_{ijk} が $D_{ijk}(o)$ ($1 \leq o < l$) まで移動したときに誤認識したデータが正しく認識されるようになる可能性があるが, このようなことが生じてても $D_{ijk}(l)$ の計算には考慮しないことにする.

以上の議論より, c_{ijk} はつぎのように制限される.

$$\dots < I_{ijk}(l) < I_{ijk}(l-1) < \dots < I_{ijk}(1) < c_{ijk} \\ < D_{ijk}(1) < \dots < D_{ijk}(l-1) < D_{ijk}(l) < \dots \quad (36)$$

もし c_{ijk} を $(I_{ijk}(1), D_{ijk}(1))$ の区間で移動させたとしても, 正しく認識されたデータは正しく認識されたままである. そして c_{ijk} を区間 $[D_{ijk}(l-1), D_{ijk}(l)]$ あるいは区間 $(I_{ijk}(l), I_{ijk}(l-1))$ の中に移動させたときは, $l-1$ 個の正しく認識されたデータが誤認識を起こす.

5.3 c_{ijk} の移動による誤認識の解消

クラス i に間違ったデータ $\mathbf{x} (\in Y)$ あるいはクラス i に属するデータがクラス o ($\neq i$) に間違ったデータ $\mathbf{x} (\in Y)$ が c_{ijk} を移動することにより正しく認識されるか調べる.

クラス i に属するデータ $\mathbf{x}$ がクラス o に誤認識してそのときのチューニングされた重み付き距離の最小値が

$h_{op}(\mathbf{x})$ であるとする. もし (23) 式が満たされるとすると, 区間 $(I_{ijk}(\mathbf{x}), D_{ijk}(\mathbf{x}))$ が存在し, 中心がその中に移動すると $\mathbf{x}$ が正しく認識される. ここで区間 $I_{ijk}(\mathbf{x})$ および $D_{ijk}(\mathbf{x})$ はおのおの (24) 式および (25) 式で与えられる.

クラス o に属するデータ $\mathbf{x}$ がクラス i に誤認識されたとする. このとき上と同様の議論により, 中心 c_{ijk} が移動するとデータ $\mathbf{x}$ が正しく認識される区間 $(-\infty, D_{ijk}(\mathbf{x}))$ および $(I_{ijk}(\mathbf{x}), \infty)$ を計算する. まずクラス o のチューニングされた重み付き距離が n 個のクラスの中で二番目の最小値である必要がある. すなわち

$$\min_k h_{ik}(\mathbf{x}) < h_{op}(\mathbf{x}) = \min_{q \neq i, r=1, \dots} h_{qr}(\mathbf{x}) \quad (37)$$

つぎに $h_{ij}(\mathbf{x})$ はクラス i で最小で, クラス i で二番目に小さいチューニングされた重み付き距離はクラス o のチューニングされた重み付き距離よりも大きい必要がある. すなわち

$$h_{ij}(\mathbf{x}) < h_{op}(\mathbf{x}) < \min_{k \neq j} h_{ik}(\mathbf{x}) \quad (38)$$

以上より $h_{op}(\mathbf{x}) > h_{ij}(\mathbf{x})$ であるから, $D_{ijk}(\mathbf{x})$ と $I_{ijk}(\mathbf{x})$ が存在して, おおのは (24) 式および (25) 式で与えられる.

5.4 認識率の増加の計算

c_{ijk} を動かすことによる認識率の向上を計算するために, c_{ijk} が移動したときの認識率の増減を示すステップ関数を求める. 正しく認識されていたデータは c_{ijk} が左に $I_{ijk}(1), \dots, I_{ijk}(l_{Mc}-1)$ あるいは $I_{ijk}(l_{Mc})$ に移動したときに, また右方向に $D_{ijk}(1), \dots, D_{ijk}(l_{Mc}-1)$, あるいは $D_{ijk}(l_{Mc})$ に移動したときに誤認識し, ステップ関数はこれらの点で1だけ減少する. 誤認識していたデータは c_{ijk} が右に $I_{ijk}(\mathbf{x})$ に動くと正しく認識され, さらに $D_{ijk}(\mathbf{x})$ まで移動すると再び誤認識する. また左に $D_{ijk}(\mathbf{x})$ まで移動すると正しく認識され, $I_{ijk}(\mathbf{x})$ まで移動すると再び誤認識する. したがってステップ関数はこれらの点で一つだけ増加あるいは減少する. これより, 区間 $[c_{ijk}, D_{ijk}(l_{Mc})]$ で増加方向に, また区間 $[I_{ijk}(l_{Mc}), c_{ijk}]$ で減少方向にステップ関数が増減する点を検出することにより, ステップ関数を生成できる.

このため増加方向に区間 $[c_{ijk}, D_{ijk}(l_{Mc})]$ に属する $I_{ijk}(\mathbf{x})$, $D_{ijk}(\mathbf{x})$ および $D_{ijk}(l)$ をソートしてソートした点を $\beta_{ijk}^u(o)$ ($o=1, \dots, o_{Mu}$) と定義する. ここで o_{Mu} は区間 $[c_{ijk}, D_{ijk}(l_{Mc})]$ に属する点の数である. $\beta_{ijk}^u(o)$ に数値 $\text{Inc}^u(o)$ をつぎのように定義する.

$$\text{Inc}^u(o) = \begin{cases} 1 & \beta_{ijk}^u(o) = I_{ijk}(\mathbf{x}) \\ -1 & \beta_{ijk}^u(o) = D_{ijk}(l), D_{ijk}(\mathbf{x}) \end{cases} \quad (39)$$

同様に, $[I_{ijk}(l_{Mc}), c_{ijk}]$ に含まれる $D_{ijk}(\mathbf{x}), I_{ijk}(\mathbf{x})$ および $I_{ijk}(l)$ を減少方向にソートして, ソートした点を $\beta_{ijk}^l(o) (o=1, \dots, o_{M_l})$ と定義する. ただし, o_{M_l} は $[I_{ijk}(l_{Mc}), c_{ijk}]$ に属する点の数である. $\beta_{ijk}^l(o)$ に対する数値 $\text{Inc}^l(o)$ をつぎのように定義する.

$$\text{Inc}^l(o) = \begin{cases} 1 & \beta_{ijk}^l(o) = D_{ijk}(\mathbf{x}) \\ -1 & \beta_{ijk}^l(o) = I_{ijk}(l), I_{ijk}(\mathbf{x}) \end{cases} \quad (40)$$

つぎに $\beta_{ijk}^u(o)$ におけるステップ関数 $f^u(o)$ の値は c_{ijk} から右方向に $\text{Inc}^u(o)$ を加算することにより決めることができる. また $\beta_{ijk}^l(o)$ におけるステップ関数 $f^l(o)$ の値も c_{ijk} から左方向に $\text{Inc}^l(o)$ を加算することにより決めることができる.

5.5 c_{ijk} の修正

$f^u(o^u) = \max_{o=1, \dots, o_{M_u}} f^u(o)$ を満たす o^u を求める. 同様に $f^l(o^l) = \max_{o=1, \dots, o_{M_l}} f^l(o)$ を満たす o^l を求める. ここで複数の o が存在するときは, 最小の o にする. もし $f^u(o^u) \geq f^l(o^l)$ のとき, c_{ijk} を右に移動する. c_{ijk} を区間 $(\beta_{ijk}^u(o^u), \beta_{ijk}^u(o^u+1))$ 内の点に移動すれば, 認識率が最大化される. したがって, c_{ijk} を次式のように設定する.

$$c_{ijk} = \beta_{ijk}^u(o^u) + \delta^c (\beta_{ijk}^u(o^u+1) - \beta_{ijk}^u(o^u)) \quad (41)$$

ここで δ^c は $0 < \delta^c < 1$ を満たす定数である.

同様に, もし $f^u(o^u) < f^l(o^l)$ のときは, c_{ijk} を左に移動する. c_{ijk} を区間 $(\beta_{ijk}^l(o^l+1), \beta_{ijk}^l(o^l))$ に設定すると, 認識率が最大化される. したがって, c_{ijk} を次式のように設定する.

$$c_{ijk} = \beta_{ijk}^l(o^l) - \delta^c (\beta_{ijk}^l(o^l) - \beta_{ijk}^l(o^l+1)) \quad (42)$$

6. メンバシップ関数の傾きのチューニング

メンバシップ関数の傾きのチューニングは位置のチューニングと同様の考え方で行うことができる. すなわちチューニングは A_{11} の α_{11} について行い, その後残りの超直方体のチューニングパラメータ α_{ij} のチューニングを順次行う. これを1回の傾きのチューニングとよぶことにする. 傾きのチューニングは教師データの認識率が100%になるか教師データの認識率の向上がなくなるまで行う. 以下ではその概要を示す. 詳細は参考文献[8]を参考にしていただきたい.

もし α_{ij} を増加させると(8)式で与えられるメンバシップ関数の成立度は増加し, 減少すると成立度は低下する. ここで $U_{ij}(l)$, $L_{ij}(l)$ を正しく認識されたデータのうちの $l-1$ 個のデータが誤認識する上限および下限とす

る. たとえば, α_{ij} の値を $[L_{ij}(2), L_{ij}(1)]$ の区間内の値に設定すると正しく認識されていたデータの 하나가誤認識を起こす. α_{ij} を増加あるいは減少させると正しく認識されていたデータが誤認識されることもあれば誤認識されていたデータが正しく認識される可能性もある. そこで $\beta_{ij}(l)$ を $U_{ij}(l)$ より小さく, 以前に誤認識されていたデータを正しく認識させる α_{ij} の上限としさらに $\gamma_{ij}(l)$ を $L_{ij}(l)$ より小さく, 以前誤認識されていたデータを正しく認識させる α_{ij} の下限とする.

次に $(L_{ij}(l), \gamma_{ij}(l))$ と $(\beta_{ij}(l), U_{ij}(l))$ ($l=1, \dots$) の区間の中でどの区間に α_{ij} を設定すると認識率が最大になるかを調べる. ここで調べる領域を制限するために l の最大値, すなわち l_M をパラメータとして設定する. このとき $(L_{ij}(k), \gamma_{ij}(k))$ および $(\beta_{ij}(k), U_{ij}(k))$ ($k=1, \dots, l_M$) の区間の中で $(L_{ij}(l), \gamma_{ij}(l))$ の区間で認識率が最大になるとすると, この区間中どこに α_{ij} を設定しても教師データの認識率は変わらないがテストデータの認識率は変わる. そこで汎化能力を調整するために $(\beta_{ij}(l), U_{ij}(l))$ の区間中で α_{ij} を次のように設定する.

$$\alpha_{ij} = \beta_{ij}(l) + \delta(U_{ij}(l) - \beta_{ij}(l)) \quad (43)$$

ただし, δ は $0 < \delta < 1$ を満足するパラメータである. 次に $(L_{ij}(l), \gamma_{ij}(l))$ の区間で認識率が最大となるときは α_{ij} を次のように設定する.

$$\alpha_{ij} = \gamma_{ij}(l) - \delta(\gamma_{ij}(l) - L_{ij}(l)) \quad (44)$$

7. シミュレーションによる評価

本論文で提案したファジィクラシファイアの性能をあやめのデータ[10], 数字データ[11], 甲状腺データ[12]および血球データ[13]を用いてメンバシップ関数が一次の場合[4]と比較した. Table 1に各データの入力数, クラス数, 教師データ数, テストデータ数を示す.

Table 1 Benchmark data specifications

Data	Inputs	Classes	Train.	Test
Iris	4	3	75	75
Numeral	12	10	810	820
Thyroid	21	3	3772	3428
Blood Cell	13	12	3097	3100

パラメータは $\alpha_{ij} = 1$, $\delta = 0.1$, $\delta^c = 0.1$ とし, 甲状腺データを除いては $\varepsilon = 0.01$ とした. 甲状腺データでは小さな値のときに認識率がよかったために, $\varepsilon = 1.0 \times 10^{-5}$ とした. またあやめのデータと数字のデータにおいては

$l_M=10$, $l_{M_c}=10$ とした。その他のデータにおいては、 $l_M=l_{M_c}$ として、 $l_M=l_{M_c}$ の値を変えて教師データの認識率を求め、認識率が最大になるときの値に設定した。認識率のよさはテストデータでの認識率で比較した。またあやめのデータと数字データでは各クラスのデータを分割しなくても十分な性能が得られたので、クラスタリングは行わなかった。計算には Sun UltraSPARC IIi 333[MHz] を用い、以下で述べる時間は学習時間とテストデータでの認識も含めた時間である。

7.1 あやめのデータ

Table 2 に各クラスに一つのファジールールを定義したときに、パラメータのチューニングを行う前 (Initial) と後 (Final) のテストデータと教師データの認識率を示す。括弧内の数字が教師データの認識率で、これ以降の表でも同じとする。最初の欄で L と Q はおのおの一次および二次のメンバーシップ関数が使われたことを示す。

一次のメンバーシップ関数で傾きと中心の両方をチューニングすると教師データの認識率は 100% になったが、テストデータの認識率は 97.33% から 94.67% に下がった。すなわち過学習が生じた。これに対して二次のメンバーシップ関数を用いたときは、テストデータの認識率は 97.33% のままで教師データの認識率が 100% となり過学習は生じなかった。

Table 2 Performance for the iris data

Method	Initial (%)	Final (%)
L	97.33 (93.33)	94.67 (100.00)
Q	97.33 (93.33)	<u>97.33 (100.00)</u>

7.2 数字データ

Table 3 に各クラスに一つのファジールールを定義したときのパラメータをチューニングする前と後の認識率を示す。二次のメンバーシップ関数を用いた方が一次のメンバーシップ関数よりも認識率がよかった。

Table 3 Performance for the numeral data

Method	Initial (%)	Final (%)
L	99.15 (99.63)	99.27 (99.88)
Q	99.51 (99.75)	<u>99.63 (100.00)</u>

7.3 甲状腺データ

Table 4 に一次のメンバーシップ関数を用いて $l_M=l_{M_c}$ として値を変えたときの、パラメータをチューニングする前と後の認識率を示す。このときにクラスタリングは行っていない。初期の認識率は低いがチューニングによ

り認識率が大幅に向上している。 $l_M(=l_{M_c})$ を変えても認識率はあまり変わっていないが、 $l_M=l_{M_c}=20$ のときに教師データで最大の認識率が得られているので、クラスタリングをするときに $l_M=l_{M_c}=20$ とした。

Table 4 Performance for the thyroid data changing l_M and l_{M_c} (with linear membership functions)

l_M, l_{M_c}	Initial (%)	Final (%)
20	9.27 (9.14)	93.49 (95.07)
40	9.27 (9.14)	93.52 (94.43)
60	9.27 (9.14)	<u>93.99 (94.94)</u>

Table 5 に θ を変えてクラスタリングをしたときのルール数とパラメータをチューニングした後の認識率を示す。 $\theta=0.4$ のときにテストデータの認識率が 96.94% で最大となった。クラスタリングをしないときに比べてルール数が 3 から 20 に増え、テストデータ (教師データ) の認識率が 3.0(3.7)% 向上している。

Table 5 Performance for the thyroid data by clustering ($l_M=l_{M_c}=20$)

θ	Rules	Rate (%)	Time [s]
0.4	14	<u>96.94</u> (98.65)	20
0.3	27	96.44 (98.81)	40
0.2	49	96.79 (98.97)	55
0.1	105	96.53 (98.81)	116

二次のメンバーシップ関数を用いて、クラスタリングを行わないで $l_M=l_{M_c}$ としたときのパラメータのチューニング前と後の認識率を Table 6 に示す。テストデータの最大の認識率は 95.74% で、一次のメンバーシップ関数でクラスタリングをしないときより認識率が向上している。 $l_M(=l_{M_c})$ を変えたときに、 $l_M=l_{M_c}=60$ で教師データの認識率が最大になったので、クラスタリングを行うときに $l_M=l_{M_c}=60$ として評価した。なおこのときテストデータでも認識率が最大になっている。

Table 6 Performance for the thyroid data by changing l_M and l_{M_c} (with quadratic membership functions)

l_M, l_{M_c}	Initial (%)	Final (%)
20	21.73 (19.35)	94.08 (95.78)
40	21.73 (19.35)	95.19 (96.53)
60	21.73 (19.35)	<u>95.74 (96.77)</u>

Table 7 にクラスタリングを行ったときのルール数と認

識率とを示す。クラスタリングをしたときは、メンバーシップ関数が一次のときと同じく $\theta=0.4$ でテストデータの認識率 98.05% と最大となり、一次のときよりテストデータでの認識率で 1.1% 向上した。二次でクラスタリングをしないとときと比較するとテスト（教師）データの認識率が 2.3(2.7)% 向上した。

Table 7 Performance for the thyroid data by clustering ($l_M = l_{M_c} = 60$)

θ	Rules	Rate (%)	Time [s]
0.4	14	<u>98.05</u> (99.50)	17
0.3	27	97.52 (99.26)	40
0.2	49	97.11 (<u>99.55</u>)	51
0.1	105	96.09 (<u>99.55</u>)	111

7.3.1 血球データ

Table 8 に、一次のメンバーシップ関数を用い、クラスタリングを行わないで l_M と l_{M_c} を同じ値で変更したときの結果を示す。 l_M と l_{M_c} が 60 のときに教師データの認識率が最大で 92.09% となった。そこで $l_M = l_{M_c} = 60$ としてクラスタリングを行った。

Table 8 Performance for the blood cell data changing l_M and l_{M_c} (with linear membership functions)

l_M, l_{M_c}	Initial (%)	Final (%)
20	75.84 (77.78)	84.16 (90.57)
40	75.84 (77.78)	84.74 (91.35)
60	75.84 (77.78)	<u>85.48</u> (92.09)

Table 9 にクラスタリングを行ったときの結果を示す。 $\theta=0.21$ でテストデータ、教師データとも認識率が最大になった。このときに 51 個のファジイルールが生成され、クラスタリングをしないととき比べてテスト（教師）データで 1.4(2.6)% 認識率が向上した。

Table 9 Performance for the blood cell data by clustering ($l_M = l_{M_c} = 60$)

θ	Rules	Rate (%)	Time [s]
0.4	14	85.81 (92.54)	14
0.3	20	85.42 (92.86)	18
0.21	51	<u>86.90</u> (94.67)	46
0.2	60	86.48 (94.32)	53

二次のメンバーシップ関数を用い、クラスタリングを行わないときの結果を Table 10 に示す。 $l_M = l_{M_c} = 60$

のときに、テストデータ、教師データともに最大の認識率が得られた。これは、一次のメンバーシップ関数でクラスタリングを行ったときよりも高い認識率となっている。 $l_M = l_{M_c}$ を最大の認識率となった $l_M = l_{M_c} = 60$ として、クラスタリングを行った。

Table 10 Performance for the blood cell data changing l_M and l_{M_c} (with quadratic membership functions)

l_M, l_{M_c}	Initial (%)	Final (%)
20	78.87 (81.53)	86.81 (93.19)
40	78.87 (81.53)	<u>88.52</u> (93.87)
60	78.87 (81.53)	87.61 (93.74)

Table 11 にクラスタリングを行った結果を示す。 $\theta=0.21$ のときにテストデータ、教師データともに最大性能が得られ、クラスタリングをしない場合に比べてテスト（教師）データで 1.5(2.0)% の認識率の向上が得られた。

Table 11 Performance for the blood cell data by clustering ($l_M = l_{M_c} = 40$)

θ	Rules	Rate (%)	Time [s]
0.4	14	87.87 (93.57)	11
0.3	20	87.19 (94.51)	19
0.21	51	<u>90.03</u> (95.87)	42
0.2	60	88.74 (95.83)	55

7.4 考察

四つのベンチマークデータに対する性能評価で一次のメンバーシップ関数に対して二次のメンバーシップ関数を用いることにより、認識率が向上することが確かめられた。Table 12 は二次のメンバーシップ関数で最大の認識率となったときのルール数と処理時間をまとめたものである。これを参考文献 [8] における性能評価結果と比較すると、処理時間はすべて一分以内で多層ニューラルネットに比較して格段に早いことが分かる。またテストデータの認識率も多層ニューラルネットの平均の認識率以上の値となっている。

8. おわりに

本論文では超直方体領域上に二次のメンバーシップ関数を定義したファジイクラシファイアについて述べた。まず最初にファジイクラシファイアの構成を述べた後、ファジイルールの抽出法とメンバーシップ関数の傾きと位置のチューニング法について述べた。すなわち、まず各クラスのデータをクラスタリングして、そのおのおののクラ

Table 12 Best performance

Data	Rules	Rate (%)	Time [s]
Iris	3	97.33 (100.00)	0.01
Numeral	12	99.63 (100.00)	0.3
Thyroid	14	98.05 (99.50)	17
Blood Cell	17	90.03 (95.87)	42

スタを超直方体で近似し、その上に二次のメンバーシップ関数を定義したファジィルールを生成する。さらに一つのメンバーシップ関数の傾きあるいは位置を変化させたときの教師データの認識率の変化を解析的に計算して、認識率が最大になるように調整して、これをすべてのメンバーシップ関数に対して行う。以上の方式の有効性をあやめのデータ、血球データ等の四つのベンチマークデータで評価して、一次のメンバーシップ関数よりも汎化能力が向上することを示した。

本論文では超直方体上のメンバーシップ関数の形状の差異による汎化能力の差を検討したが、異なる形状の領域上に定義されたメンバーシップ関数が汎化能力にどのような影響を及ぼすかは重要な問題であり、今後の検討課題としたい。

謝 辞

本研究を進めるにあたり、血球データを提供していた川崎医科大学松田信義教授、数字およびひらがなデータを提供していただいた(株)日立製作所日立研究所小林芳樹技術主幹、およびサイロイドデータ(ics.uci.edu: pub/machine-learning-databases)を提供いただいたカリフォルニア大学アーバイン校 P. M. Murphy 氏および D. W. Aha 氏に感謝の意を表する。

参 考 文 献

- [1] 阿部: ニューラルネットとファジィシステム: 理論と応用, 近代科学社 (1995)
- [2] P. K. Simpson: Fuzzy min-max neural networks - part 1: classification; *IEEE Trans. Neural Networks*, Vol. 3, No. 5, pp. 776-786 (1992)
- [3] S. Abe and M. -S. Lan: A method for fuzzy rules extraction directly from numerical data and its application to pattern classification; *IEEE Trans. Fuzzy Systems*, Vol. 3, No. 1, pp. 18-28 (1995)
- [4] S. Abe: Fast training of a fuzzy classifier with pyramidal membership functions; *Proc. SCI '99/ISAS '99*, Vol. 3, pp. 487-492 (1999)
- [5] F. Uebele, S. Abe and M. -S. Lan: A neural network-based fuzzy classifier; *IEEE Trans. Syst., Man, Cybern.*, Vol. 25, No. 2, pp. 353-361 (1995)
- [6] 中森: ファジィモデリング, オーム社 (1994)
- [7] 井上, 亀井, 井上: 遺伝的アルゴリズムによる超楕円錐形メンバーシップ関数を用いたファジィルールの自動生成; 第13回ファジィ・システム・シンポジウム講演論文集, pp. 341-344 (1997)
- [8] S. Abe and R. Thawonmas: A fuzzy classifier with ellipsoidal regions; *IEEE Trans. Fuzzy Systems*, Vol. 5, No. 3, pp. 358-368 (1998)
- [9] S. Abe: Dynamic cluster generation for a fuzzy classifier with ellipsoidal regions; *IEEE Trans. Systems, Man, and Cybernetics-Part B*, Vol. 28, No. 6, pp. 869-876 (1998)
- [10] R. Fisher: The use of multiple measurements in taxonomic problems; *Annals of Eugenics*, Vol. 7, Part II, pp. 179-188 (1936)
- [11] 武長, 阿部, 高藤, 鹿山, 北村, 奥山: 感度解析を用いたニューラルネットの入力層の最適化とその数字認識への適用; 電学論 D, Vol. 111-D, No. 1, pp. 36-44 (1991)
- [12] S. M. Weiss and I. Kapouleas: An empirical comparison of pattern recognition, neural nets, and machine learning classification methods; *Proc. International Joint Conference on Artificial Intelligence*, pp. 781-787 (1989)
- [13] A. Hashizume, J. Motoike and R. Yabe: Fully automated blood cell differential system and its application; *Proc. IUPAC 3rd International Congress on Automation and New Technology in the Clinical Laboratory*, pp. 297-302 (1988)