論　文

競合型ニューラルネットを用いたオンライン準教師付能動学習手法

櫻井　啓介[†]　　神谷　祐樹[††a)]　　長谷川　修[†††b)]

Online Semi-Supervised Active Learning Method Using Competitive Neural Network

Keisuke SAKURAI[†], Youki KAMIYA[††a)], and Osamu HASEGAWA[†††b)]

あらまし　一般に，教師付学習は入力データと出力ラベルの組を用いて行われる．しかし通常，入力データに対する出力ラベルの付与は人手によって行われるため，コストがかかる．そこで，より少ない出力ラベルの付与でより良い学習結果を得るための学習法として，準教師付学習と能動学習が提案されている．本論文では，オンライン教師なし学習手法の一つである自己増殖型ニューラルネットワークに，準教師付学習と能動学習の機能を加えることにより，オンライン準教師付能動学習手法を提案する．また，計算機実験によって，提案手法が有効に働くことを示す．

キーワード　準教師付学習，能動学習，オンライン学習，自己増殖型ニューラルネットワーク (SOINN)

1. まえがき

一般に機械による学習は，教師付学習と教師なし学習に大別される．教師付学習は入力データとそれに対応する出力ラベルの組，すなわちラベル付データを用いて，それらの間に存在する写像関係を推定する学習である．推定した写像関係が正しければ，未知の入力データに対する出力ラベルを正しく予測することができる．一方，教師なし学習は入力のみ，すなわちラベルなしデータを用いて，それらに包含される構造を推定したり，いくつかの異なる概念に分類する学習である．一般に，教師付学習によってより良い学習結果を得るためには，多数のラベル付データが必要である．しかし通常，出力ラベルの付与は人手で行う必要があり，コストがかかるため，多数用意することは困難である．

準教師付学習は，一部の入力データのみに出力ラベルを付与し，それらのみから，入力データ全般の一般的な写像関係を推定する学習法である．準教師付学習は，多数の入力データにラベルを付与するコストを削減できることから，近年注目されている．

また，より少ないラベルの付与で，より良い学習結果を得るための学習法の一つに，能動学習がある．能動学習は質問学習とも呼ばれる．能動学習とは，これまでの学習結果に基づいて，最も必要な出力ラベルを推定し，それを得ることによって，より効率的に学習を行う方法である．

準教師付学習と能動学習は，ともに少ないラベル付データを用いてより良い学習結果を得る方法である．したがって，これらを組み合わせることで，更に少ない出力ラベルの付与で良好な学習結果を得ることができると期待される．従来の準教師付能動学習法として，Generative Model [9]，Co-Training [2]，Transductive SVM (TSVM) [1] や GRF [12] 等が提案されている．また，これらの手法を拡張した準教師付能動学習法も提案されている [7], [13]．これらの手法は，全入力データを用いて学習を行うバッチ学習である．

ここで実環境においては，システムは非常に膨大な数のデータを扱うことになる．それらのデータに対してバッチ学習を行うためには，そのすべてを蓄えてお

[†] アップルジャパン株式会社，東京都
Apple Japan, Inc., Tokyo, 163–1480 Japan
[††] 東京工業大学大学院総合理工学研究科知能システム科学専攻，横浜市
Department of Computational Intelligence and System Science, Tokyo Institute of Technology, Yokohama-shi, 226–8503 Japan
[†††] 東京工業大学像情報工学研究施設，横浜市
Imaging Science and Engineering Laboratory, Tokyo Institute of Technology, Yokohama-shi, 226–8503 Japan
a) E-mail: kamiya.y.ab@m.titech.ac.jp
b) E-mail: hasegawa.o.aa@m.titech.ac.jp

く必要があるため，膨大なメモリが必要である．そこで，学習データを入手するごとに学習機械が更新され，後の更新にはこの学習データを用いない学習法，すなわちオンライン学習が重要になる．

そのようなオンライン学習の手法の一つに，Shen と Hasegawa によって提案された，Self-Organizing Incremental Neural Network (SOINN) [11] がある．SOINN は，それぞれのデータ間の距離，すなわち類似性に基づいて，データの分布をいくつかのクラスタに分ける，教師なしクラスタリングの手法の一つである．SOINN は，非定常的な入力を学習でき，複雑な形状の分布をもつクラスに対しても，クラス数などの事前知識なしに，適切なクラスタ数及び適切な位相構造を出力できる．

本論文では，こうした SOINN の機能を改良し，入力データの構造を自己組織的に近似・抽出しながら，その構造中の適当な箇所に能動的にラベルを付与する，オンライン準教師付能動学習の手法を提案する．

本論文の構成を以下に示す．**2.** では，提案手法のもととなる SOINN の概要について説明する．**3.** では，SOINN の新たなクラスタ判別方法を提案した後，教師なし学習である SOINN を準教師付学習に拡張する方法を与える．**4.** では，**3.** で提案した手法を用いて計算機実験を行い，その有効性を示す．**5.** では，本研究の成果をまとめるとともに，今後解決すべき課題を述べる．

2. SOINN の概要

本章では，提案手法のもととなる SOINN の概要について説明する．

SOINN は，自己組織化マップ (SOM) [4] と競合ヘッブ学習則 [6] を組み合わせて入力データの位相構造を学習する手法である．入力データの代表点としてノードを生成し，それらを結合する辺を用いて位相構造を表現する．同様の手法に，Neural Gas (NG) [6] や Growing Neural Gas (GNG) [3] がある．しかし，NG や GNG ではノード数が恒久的に増加してしまうため，永続的な追加学習には適さない．これに対し SOINN では，入力データの位相構造を適切に表現するために必要なノード数を自律的に獲得することにより，永続的な学習に適している．

SOINN は 2 層ネットワーク構造により学習を行う．1 層目の学習によって，入力データの確率密度分布を近似するようにノードを生成し，2 層目の学習によって，データの分布が低密度の領域を切り分けることでクラスタリングを行う．1 層目と 2 層目は同じ学習アルゴリズムで動作する．図 1 に SOINN の学習アルゴリズムのフローチャートを示す．SOINN は，入力データ集合から入力ベクトルが与えられると，入力ベクトルに最も近いノード（第 1 勝者ノード）及び 2 番目に近いノード（第 2 勝者ノード）を探索する．そして，類似度しきい値という規準を用いて，入力ベクトルがそれらのノードと同一クラスタに属するかを判定する．SOINN 1 層目では，入力データの分布は未知であるので，類似度しきい値はノードごとに適応的に変化させる．ノード i が辺によって直接的に接続しているノード（隣接ノード）をもつ場合，その類似度しきい値 T_i は，隣接ノードとの最大距離

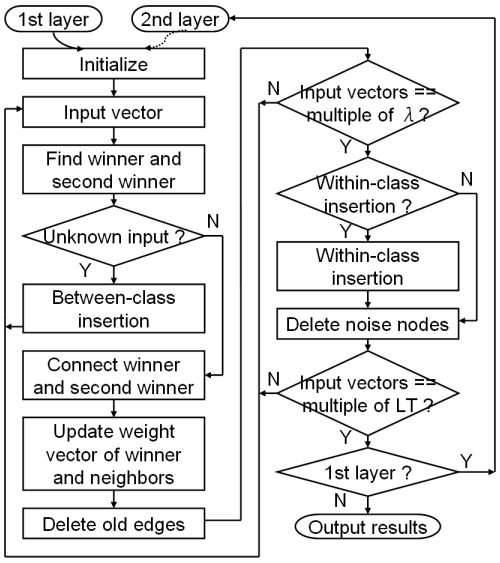

図 1　SOINN のフローチャート
Fig. 1　Flowchart of SOINN.

$$T_i = \max_{j \in N_i} \|\boldsymbol{W}_i - \boldsymbol{W}_j\| \tag{1}$$

によって与えられる．ここで，N_i はノード i の隣接ノードの集合，$\boldsymbol{W}_i$ はノード i の重みベクトルを表す．また，ノード i が隣接ノードをもたない場合，その類似度しきい値 T_i は，ノード i からそれ以外の各ノードとの最小距離

$$T_i = \min_{j \in N \setminus \{i\}} \|\boldsymbol{W}_i - \boldsymbol{W}_j\| \tag{2}$$

によって与えられる．ここで，N はノード全体の集合を表す．

入力データと第 1 勝者ノード及び第 2 勝者ノードとの距離が類似度しきい値よりも大きい場合，それらは異なるクラスタに属すると判定する．このとき，入力ベクトルと同じ位置にノードを挿入し，次の入力を処理する．このときの挿入はクラス間挿入と呼ばれる．また，同一クラスタに属すると判定された場合，第 1 勝者ノード及び第 2 勝者ノードの隣接ノードの重みベクトルを更新する．このとき第 1 勝者ノードになったノードを i と表し，i がこれまでに第 1 勝者ノードになった回数を M_i と表せば，第 1 勝者ノード i の重みベクトルの更新量 $\Delta \boldsymbol{W}_i$ と，i の隣接ノード $j(\in N_i)$ の重みベクトルの更新量 $\Delta \boldsymbol{W}_j$ は，

$$\Delta \boldsymbol{W}_i = \frac{1}{M_i}(\boldsymbol{W}_s - \boldsymbol{W}_i) \tag{3}$$

$$\Delta \boldsymbol{W}_j = \frac{1}{100 M_i}(\boldsymbol{W}_s - \boldsymbol{W}_j) \tag{4}$$

で与えられる．ここで，$\boldsymbol{W}_s$ は入力の重みベクトルを表す．また，第 1 勝者ノードと第 2 勝者ノードの間に辺が存在しなければ，それらの間に辺を生成し，その辺の「年齢」を 0 にする．そして，第 1 勝者ノードに接続するすべての辺の年齢を +1 する．このとき，あらかじめ決められたパラメータである $\mathrm{age}_{\max}$ を超えた年齢をもつ辺は削除する．更に，入力が λ 回与えられるごとに，隣接ノード数が一つ以下のノードを低密度の領域に位置するとみなし削除する．その際，入力データとノードの位置との誤差が大きい領域にノードを挿入する．ただし，挿入によって誤差が減少しない場合は，挿入は取り消される．このときの挿入はクラス内挿入と呼ばれる．

ところで，前述のとおり SOINN 1 層目では類似度しきい値が適応的に変化する．そのため，入力データとノードの位置との誤差は大きくならず，クラス内挿入はほとんど行われない．そこで，SOINN 1 層目ではクラス内挿入を行う必要はない．ここで SOINN 1 層目は学習結果として，入力データの代表点であるノードの重みベクトルとその位相構造を出力する．1 層目の学習で，入力回数があらかじめ設定された回数 LT に達したときに出力結果として得られた重みベクトルは，2 層目の入力として使用される．SOINN 2 層目は 1 層目と同様のアルゴリズムで動作する．しかし，SOINN 2 層目では，SOINN 1 層目の学習結果が更新された場合，これまでの学習結果を消去し，学習し直す必要があり，オンライン追加学習に対応していない．一方，SOINN 1 層目の学習だけでは，重なりがあるクラスタを適切に分離することができない．

そこで本論文では，オンライン学習である SOINN 1 層目の学習のみで，従来の 2 層目のクラスタ識別性能と同等の結果を得られる，新たなクラスタ判別方法を提案する．その詳細は次章で述べる．ここで，本論文で用いる SOINN 1 層目の学習アルゴリズムをまとめて述べる．ただし前述の理由から，クラス内挿入は行わないこととする．

アルゴリズム 2.1：SOINN 1 層目の学習アルゴリズム

Step 1. 入力データ集合 I_s から二つの入力 $s1$, $s2(\in I_s)$ をランダムに選択し，それぞれの重みベクトル $\boldsymbol{W}_{s1}$，$\boldsymbol{W}_{s2}$ をもつノード n_1 と n_2 を生成する．また，ノード全体の集合を $N = \{n_1, n_2\}$ に初期化する．

Step 2. 入力 $s \in I_s$ をランダムに選択する．その重みベクトルを $\boldsymbol{W}_s$ と表す．

Step 3. 第 1 勝者ノード

$$w1 = \underset{i \in N \setminus \{i\}}{\operatorname{argmin}} \|\boldsymbol{W}_i - \boldsymbol{W}_s\| \tag{5}$$

及び第 2 勝者ノード

$$w2 = \underset{i \in N \setminus \{i, w1\}}{\operatorname{argmin}} \|\boldsymbol{W}_i - \boldsymbol{W}_s\| \tag{6}$$

を探索する．

Step 4. 入力データの重みベクトル $\boldsymbol{W}_s$ と第 1 勝者ノードの重みベクトル $\boldsymbol{W}_{w1}$ 及び第 2 勝者ノードの重みベクトル $\boldsymbol{W}_{w2}$ との距離と，類似度しきい値 T_{w1}, T_{w2} との間で

$$\|\boldsymbol{W}_s - \boldsymbol{W}_{w1}\| > T_{w1} \tag{7}$$

$$\|\boldsymbol{W}_s - \boldsymbol{W}_{w2}\| > T_{w2} \tag{8}$$

のいずれかの条件を満たす場合，重みベクトル $\boldsymbol{W}_s$ をもつノード n_s を生成し，$M_{n_s} = 0$ に初期化する．また，ノード集合を $N = N \cup \{n_s\}$ に更新し，Step 8. へ．上のいずれの条件も満たさない場合，$M_{w1} = M_{w1} + 1$ に更新する．

Step 5. 第 1 勝者ノードと第 2 勝者ノードの間に辺の接続がない場合，辺 $(w1, w2)$ を生成し，$\mathrm{age}_{(w1,w2)} = 0$ に初期化する．ここで，$\mathrm{age}_{(i,j)}$ は辺 (i, j) の年齢を表す．

Step 6. 第 1 勝者ノードの重みベクトルを

$$\boldsymbol{W}_{w1} = \boldsymbol{W}_{w1} + \Delta \boldsymbol{W}_{w1} \tag{9}$$

に更新する．また，$w1$ のすべての隣接ノード $i \in N_{w1}$

に対して，重みベクトルを

$$\boldsymbol{W}_i = \boldsymbol{W}_i + \Delta \boldsymbol{W}_i \tag{10}$$

に更新し，第 1 勝者ノードとの間の辺の年齢を

$$\mathrm{age}_{(w1,i)} = \mathrm{age}_{(w1,i)} + 1 \tag{11}$$

に更新する．

Step 7. 辺の年齢が $\mathrm{age}_{\max}$ を超える辺を削除する．

Step 8. 入力回数が λ の倍数ならば，隣接ノード数が一つ以下のノードをノイズノードとして削除する．

Step 9. 学習を終了する場合は停止する．終了しない場合は Step 2. へ．

次に，学習結果として得られたノードの属するクラスタを決定する方法について述べる．SOINN では競合ヘッブ学習則に従って，ノード間にパス，すなわち連続的な辺の接続が存在するとき，それらのノードが同一クラスタに属すると判断する．以下にノードの属するクラスタを決定するアルゴリズムを示す．

アルゴリズム 2.2：ノードの属するクラスタ決定

Step 1. クラスタ決定されていないノードの集合 J_C を $J_C = N$ に初期化する．

Step 2. ノード $i \in J_C$ をランダムに選択し，新しいクラスタラベル C_i を付与する．また，$J_C = J_C \backslash \{i\}$ に更新し，クラスタ C_i のノード集合 N_{C_i} を $N_{C_i} = \{i\}$ に初期化する．

Step 3. ノード i との間にパスが存在するノードの集合を N_p と表す．$j \in N_p$ であるすべてのノードに対して，クラスタラベル C を付与する．また，$J_C = J_C \backslash N_{C_i}$，$N_{C_i} = N_{C_i} \cup N_p$ に更新する．

Step 4. $J_C = \Phi$ ならば終了．そうでないなら Step 2. へ．

次章で，本章で紹介した SOINN を拡張したオンライン準教師付能動学習手法を提案する．

3. 提案手法

本章ではまず，SOINN 1 層目の学習のみで，従来の 2 層構造の SOINN のクラスタ識別性能と同等の性能を得られる，新たなクラスタ判別方法を提案する．その後，教師なし学習である SOINN を準教師付能動学習に拡張する方法を与える．

3.1 提案手法によるクラスタリング方法

従来の SOINN では，1 層目の学習で入力の確率密度分布を近似するようにノードとその位相関係を生成し，2 層目の学習で低密度の領域を切り分けることでクラスタリングを実現していた．しかし，SOINN 2 層目はオンラインの追加学習ができない．SOINN 1 層目の学習結果のみから低密度の領域を切り分けることができれば，オンライン教師なし分類が実現できる．

SOINN 1 層目の学習結果であるノードとその位相関係は，入力データの確率密度分布を近似するように生成されたものである．一般に，分布が高密度の領域には多くのノードが生成される．そのため，そのような領域では，隣接ノード間の距離，すなわちノードを接続している辺の長さが小さくなる．そこで，ノード i の密度 $D(i)$ を，ノード i の隣接ノードとの平均距離

$$d_i = \frac{\sum_{j \in N_i} \|\boldsymbol{W}_i - \boldsymbol{W}_j\|}{|N_i|} \tag{12}$$

を用いて，次のように定義する．

$$D(i) = \frac{1}{(1+d_i)^2} \tag{13}$$

式 (12) で，$|N_i|$ は N_i の要素数を表す．このように定義されたノードの密度が大きな領域ほど，入力データの確率密度が大きいと考えられる．ここで，各クラスの中心では入力データの確率密度が高く，クラスの重なる領域では入力データの確率密度が低い（図 2）と仮定すれば，ノードの密度が大きい領域がクラスの中心で，ノードの密度が小さい領域がクラスの重なり領域であると判断することができる．そこで，局所的に最大となるノード，すなわちすべての隣接ノード $j \in N_i$ について $D(i) > D(j)$ を満たすようなノード i を中心ノードと呼ぶことにする．

この中心ノードに基づき，SOINN 1 層目によって得られたそれぞれのクラスタを，より小さなサブクラスタに分割する．以下に，ノードの属するサブクラスタを決定するアルゴリズムを示す．

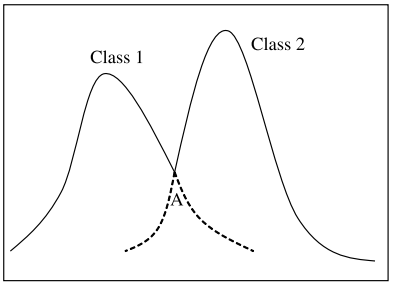

図 2 重なりのあるクラスの確率密度分布
Fig. 2 Censity distribution of overlapped classes.

アルゴリズム 3.1：ノードの属するサブクラスタ決定
Step 1. 局所的に最大となるノード（中心ノード）を $\{c_i\}_{i=1}^{n}$ と表し，それらすべてに異なるサブクラスタラベルを付与する．また，サブクラスタラベル付与済ノードの集合 J_S を $J_S = \{c_i\}_{i=1}^{n}$ に初期化する．
Step 2. サブクラスタラベルが付与されていないノードの中で，密度が最大のノード

$$i = \underset{j \in N \setminus S}{\operatorname{argmax}} D(j) \tag{14}$$

を探索する．
Step 3. ノード i に，隣接ノードの中で密度が最大のノード

$$\underset{j \in N_i}{\operatorname{argmax}} D(j) \tag{15}$$

のサブクラスタラベルと同一のラベルを付与する．また，サブクラスタラベル付与済ノード集合 J_S を

$$J_S = J_S \cup \{i\} \tag{16}$$

に更新する．
Step 4. $J_S = N$ ならば終了．そうでないなら Step 2. へ．

ノードの密度はノイズなどの影響によって，細かい凹凸のある分布になってしまう場合がある（図 3）．そのような場合，必要以上に小さなサブクラスタに分割してしまうことになる．そこで，細かい凹凸によって分割されたサブクラスタであるかを判定し，複数のサブクラスタをグループ化することで平滑化を実現する．

提案手法ではグループ化する条件に，サブクラスタの中心ノードの密度と，サブクラスタ境界の密度との差を用いる．ここで，サブクラスタ A に属するノード

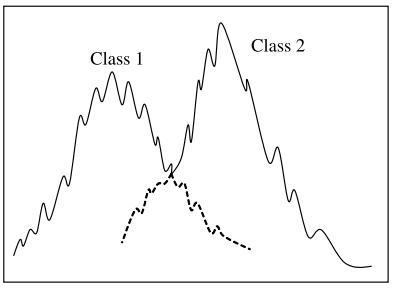

図 3 重なりのあるクラスの確率密度分布（細かい凹凸がある場合）
Fig. 3 Clusters with overlapped area (with fluctuation).

a とサブクラスタ B に属するノード b を接続する辺 (a, b) を，A と B のサブクラスタ境界辺と呼ぶことにする．サブクラスタ境界辺は図 2 の A で示した，密度の「谷間」に位置する辺である．そこで，その谷間の密度をサブクラスタ境界辺の密度と呼ぶことにし，

$$D(a, b) = \min(D(a), D(b)) \tag{17}$$

と定義する．一般にサブクラスタ境界辺は，それぞれのサブクラスタ間に複数存在するため，その中で密度が最大の境界辺を最大境界辺と呼ぶことにし，最大境界辺の密度

$$D_{AB} = \max_{(a,b) \in E_{AB}} D(a, b) \tag{18}$$

を用いてグループ化判定を行うことにする．ここで，E_{AB} は A と B のサブクラスタ境界辺の集合を表す．このとき，

$$D(c_A) - D_{AB} < G_C \tag{19}$$

または，

$$D(c_B) - D_{AB} < G_C \tag{20}$$

のいずれかを満たす場合，サブクラスタ A とサブクラスタ B は同一グループであると判定する．ここで，c_A，c_B はサブクラスタ A とサブクラスタ B の中心ノード，G_C はサブクラスタ A とサブクラスタ B が属するクラスタ C のグループ化しきい値を表す．また，クラスタ C のグループ化しきい値 G_C を

$$G_C = \frac{\alpha}{|E_C|} \sum_{(i,j) \in E_C} |D(i) - D(j)| \tag{21}$$

と定義する．式 (21) で，E_C はクラスタ C の辺の集合，$|E_C|$ は E_C の要素数，α は平滑化パラメータを表す．それぞれのサブクラスタの最大境界辺について，式 (19) と式 (20) の判定を行ってグループ化を実現する．以下に，サブクラスタをグループ化するアルゴリズムをまとめる．
アルゴリズム 3.2：サブクラスタのグループ化
Step 1. グループ化判定していない最大境界辺の集合を $J_G = \{(a_i, b_i)\}_{i=1}^{n}$ と表す．また，すべてのノードに対し，グループラベルが付与されていない状態にする．
Step 2. J_G の中から，

$$(a_{\max}, b_{\max}) = \underset{(a_i, b_i) \in J_G}{\operatorname{argmax}} D(a_i, b_i) \tag{22}$$

を探索する．このとき，辺 $(a_{\max}, b_{\max})$ の両端のノード $a_{\max}$ と $b_{\max}$ の両方に対してグループラベルが付与されているならば，Step 3. へ．そうでない場合，$a_{\max}$ と $b_{\max}$ が属するサブクラスタをそれぞれ A，B と表す．このとき，式 (19) または式 (20) を満たすならば，サブクラスタ A と B が属するグループのすべてのノードに対して，同一のグループラベルを付与する．

Step 3. $(a_{\max}, b_{\max})$ をグループ化判定済みにする．すなわち，

$$J_G = J_G \backslash \{(a_{\max}, b_{\max})\} \tag{23}$$

に更新する．このとき $J_G = \Phi$ ならば，グループラベルが付けられていないノードに対して，それぞれのサブクラスタごとに異なるグループラベルを付与して終了．そうでなければ Step 2. へ．

同様な方法で，SOINN 1 層目の学習のみで低密度領域の切り分けを実現した手法に，小倉らが提案した Enhanced-SOINN (ESOINN) がある [10]．しかし ESOINN では，密度を用いて第 1 勝者ノードと第 2 勝者ノードとの間に辺を生成するかを判定し，クラスタリング方法は従来と同じくパスの有無によって行う．そのため ESOINN では，サブクラスタ間の重なり領域を学習後に検出することができない．

一方提案手法では，SOINN によって得られた位相関係は全く変えずに，クラスタリング方法だけを変えているため，学習後にもサブクラスタ間の重なり領域を検出することが可能である．この重なり領域の情報は，準教師付能動学習に拡張する際に用いる．

3.2 SOINN の準教師付能動学習への拡張

SOINN は教師なし学習の手法である．そのため，SOINN は入力データにクラスラベルが付与されていても扱うことができない．そこでまず，SOINN でラベル付データを扱う方法を与える．SOINN に対して，重みベクトルが $\boldsymbol{W}_s$ であるラベル付データが入力されたとき，その最近傍ノード

$$\underset{i \in N}{\operatorname{argmin}} \|\boldsymbol{W}_i - \boldsymbol{W}_s\| \tag{24}$$

に，入力データと同じクラスラベルを付与する．このように，ラベル付データによって直接的にクラスラベルを付与されたノードのことを教師付ノードと呼ぶことにする．これ以降は入力されたラベル付データを用いずに，SOINN によって生成されたノードである教師付ノードのみを用いることで，SOINN のもつオンライン性を保持したままラベル付データを扱うことができる．

ところで SOINN では，入力データの確率密度分布に基づいてノードと位相関係を生成し，同一クラスのデータが同一クラスタになるようにクラスタリングしている．更にアルゴリズム 3.1 で，クラスタ内のノードの分布を用いてクラスの重なり領域を検出し，クラスタをより詳細なサブクラスタに分割した．したがって，同一クラスタに属するノードは同一クラスラベルをもつ可能性が高く，同一サブクラスタに属するノードは同一クラスラベルをもつ可能性がより高いと考えられる．また，密度が小さいノードはクラスの重なり領域のノードである可能性が高く，密度が大きいノードほどクラスの中心付近のノードであると考えることができる．したがって，教師付ノードでないノードのクラスラベルは，そのノードが属するサブクラスタ内で最も密度が大きい教師付ノードのクラスラベルと同一であると判断することにする．また，同一サブクラスタ内に教師付ノードがない場合，同一サブクラスタグループ内，同一クラスタ内の順で教師付ノードを探索し，同一のクラスラベルを付与する．これをまとめた，ノードへのクラスラベル付与のアルゴリズムを示す．

アルゴリズム 3.3：教師付ノードを用いたクラスラベル付与

Step 1. 教師付ノードの集合を N_L と表し，クラスラベル付与済ノードの集合 J_L を $J_L = N_L$ に初期化する．

Step 2. ノード $i \in N \backslash J_L$ を一つ選ぶ．

Step 3. ノード i と同一サブクラスタラベルをもつノードの集合を N_S と表す．このとき，$N_S \cap J_L \neq \Phi$ ならば，

$$l = \underset{j \in N_S \cap J_L}{\operatorname{argmax}} D(j) \tag{25}$$

であるノード l を探索し，Step 7. へ．

Step 4. ノード i と同一グループラベルをもつノードの集合を N_G と表す．このとき，$N_G \cap J_L \neq \Phi$ ならば，

$$l = \underset{j \in N_G \cap J_L}{\operatorname{argmax}} D(j) \tag{26}$$

であるノード l を探索し，Step 7. へ．

Step 5. ノード i と同一クラスタラベルをもつノードの集合を N_C と表す．このとき，$N_C \cap J_L \neq \Phi$ な

らば，

$$l = \underset{j \in N_C \cap J_L}{\operatorname{argmax}} D(j) \tag{27}$$

であるノード l を探索し，Step 7. へ．

Step 6.　N_S のすべてのノードに対して，未知クラスラベルを付与する．また，クラスラベル付与済ノード集合を $J_L = J_L \cup N_S$ に更新し，Step 8. へ．

Step 7.　N_S のすべてのノードに対して，教師付ノード l のクラスラベルを付与する．また，クラスラベル付与済ノード集合を $J_L = J_L \cup N_S$ に更新する．

Step 8.　$J_L = N$ ならば終了．そうでなければ Step 2. へ．

このアルゴリズムによって，SOINN で得られたノードの一部にクラスラベルを付与するだけで，データ全体を表すすべてのノードに対してクラスラベルを付与することができる．すなわち，SOINN を準教師付学習に拡張したことになる．

ここで，アルゴリズム 3.3 を用いてクラスラベル付与を行うとき，サブクラスタの中心ノードが教師付ノードならば，同一サブクラスタ内の他の教師付ノードは，その他のノードのクラスラベル付与に影響を与えない．したがって，少ないクラスラベルの付与で効率良く学習するためには，まず中心ノードのクラスラベルだけを得ればよいことが分かる．また，サブクラスタよりサブクラスタグループ，サブクラスタグループよりクラスタの方がより広範囲の分布を表現している．更に，その中のノード数が多いほど広範囲の分布を表していると考えられる．したがって，より効率の良い学習のためには，ノード数の多いクラスタ，サブクラスタグループ，サブクラスタの順で中心ノードのクラスラベルを能動的に要求すればよいことが分かる．

クラスの確率密度分布に重なりがある場合，アルゴリズム 3.3 に基づいてクラスラベルを付与すると，辺で直接的に接続した隣接ノード同士で異なる推定クラスラベルをもつノードが生じる可能性がある．これらのノードが存在する領域は，クラスの境界であると考えられる．そこで，クラス間の境界を精度良く学習するために，異なるクラスラベルの隣接ノードをもつノード（クラス境界ノードと呼ぶことにする）についてクラスラベルを要求する．これらをまとめた，クラスラベル質問アルゴリズムを以下に示す．

アルゴリズム 3.4：クラスラベルの質問

Step 1.　教師付ノードをもたないクラスタの集合を $A_C = \{C_i\}_{i=1}^{n}$ と表す．このとき，$A_C \neq \Phi$ ならば，

$$C = \underset{a \in A_C}{\operatorname{argmax}} |N_a| \tag{28}$$

を探索する．そして，クラスタ C に属する最大密度のノード

$$q = \underset{i \in N_C}{\operatorname{argmax}} D(i) \tag{29}$$

を探索し，Step 5. へ．

Step2.　教師付ノードをもたないグループの集合を $A_G = \{G_i\}_{i=1}^{m}$ と表す．このとき，$A_G \neq \Phi$ ならば，

$$G = \underset{a \in A_G}{\operatorname{argmax}} |N_a| \tag{30}$$

を探索する．そして，グループ G に属する最大密度のノード

$$q = \underset{i \in N_G}{\operatorname{argmax}} D(i) \tag{31}$$

を探索し，Step 5. へ．

Step 3.　教師付ノードをもたないサブクラスタの集合を $A_S = \{S_i\}_{i=1}^{l}$ と表す．このとき，$A_S \neq \Phi$ ならば，

$$S = \underset{a \in A_S}{\operatorname{argmax}} |N_a| \tag{32}$$

を探索する．そして，サブクラスタ S に属する最大密度のノード

$$q = \underset{i \in N_S}{\operatorname{argmax}} D(i) \tag{33}$$

を探索し，Step 5. へ．

Step 4.　教師付ノードでないクラス境界ノードの集合を N_B と表す．このとき，$N_B = \Phi$ ならば，質問せずに終了．$N_B \neq \Phi$ ならば，$q \in N_B$ をランダムに一つ選択．

Step 5.　ノード q の重みベクトル $\boldsymbol{W}_q$ に対応する出力ラベルを質問し終了．

このようにして，準教師付能動学習を実現する．ただし，クラス境界ノードについてクラスラベルを質問する場合，ノードに付与されているクラスラベルと，質問して得たラベル付データのクラスラベルとが矛盾する場合がある．アルゴリズム 3.3 に基づいて付与されるクラスラベルは，他の教師付ノードのクラスラベルとクラスタリング情報から推定したものである．したがってそのような場合には，ラベル付データのクラスラベルを優先してクラス境界ノードのクラスラベルを上書きする．

また，クラスの確率密度分布に重なりがある分布が追加的に生じる場合，既存の教師付ノードが新たに生じたクラスの分布と重なる領域に含まれる可能性がある．しかし，前述のとおり，サブクラスタの中心ノードが教師付ノードならば，同一サブクラスタ内の他の教師付ノードは，その他のノードのクラスラベル付与に影響を与えない．したがって，あるクラスの教師付ノードが追加的に生じた別クラスに関するサブクラスタの分布の頂点にない限り，追加的なクラスラベルの付与（アルゴリズム 3.3）に悪影響を及ぼすことはない．本章で述べた提案手法を用いた計算機実験を次章に示す．

4. 計算機実験

本章では，計算機実験によって，**3.** で述べた提案手法が有効に働くことを示す．

4.1 人工データを用いた実験

最初に，入力データとして人工データを用いた実験を行った．使用した入力データ集合を図 4 に示す．データは分布に重なりのある二つのガウス分布，二つの同心円，及びサインカーブによって構成されている．二つのガウス分布については，10%のベイズ誤り確率を含むように配置した．同心円及びサインカーブの形状の分布は一様分布である．同心円の内側とガウス分布の下のものは同一クラス，他の分布はすべて異なるクラスの計 4 クラスとした．また，実世界の環境を想定して，入力データには 10%の一様ノイズを加えた．

まず，入力をデータ集合全体からランダムに選択する，定常的な環境での実験を行った．クラスラベルの要求は，SOINN の入力 200,000 回後に行われるノイズノードの削除の後に，アルゴリズム 3.4 に従って要求されるノードに対して行い，要求が生じなくなるまで行った．提案手法のパラメータは，$\lambda = 750$, $\mathrm{age}_{\max} = 30$, $\alpha = 2.2$ で実験を行った．

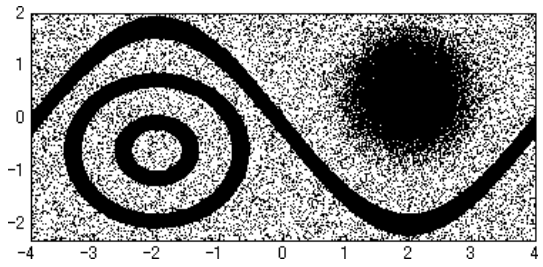

図 4 実験に用いた人工データ集合
Fig. 4 Artificial data used for experiment.

結果を図 5 に示す．図 5 は，ノードに付与されたクラスラベルごとにそれぞれ異なる記号を用いて表されている．また，教師付ノードの記号は大きなもので表されている．この図から提案手法によって，ノイズの影響を受けずに入力分布が表現されていることが分かる．また，クラスラベルの要求について，一様分布である同心円及びサインカーブの分布では，クラスラベルの要求がほぼ一様に分布したサブクラスタに対して生じている．一方，二つのガウス分布では，クラスの分布が重なる領域にクラスラベルの要求が集中している．最終的に，70 個のクラスラベルで全体のノードに適切なクラスラベルが付与された．したがって本実験において，提案手法は生成された 331 個のノード全体に対して 21.1%のクラスラベル，入力データ全体に対して 70/200,000 = 0.035%のクラスラベルで，効率良く学習を行うことができた．

次に，オンライン学習を想定し，入力データ集合が変化する非定常的な環境での実験を行った．入力データ集合を入力 50,000 回ごとに，下側のガウス分布と同心円の内側，上側のガウス分布，同心円の外側，サインカーブの順で変化させ実験を行った．提案手法のパラメータは，$\lambda = 250$, $\mathrm{age}_{\max} = 30$, $\alpha = 2.5$ とした．クラスラベルの要求は入力 50,000 回ごとのノイズノード削除の後に，アルゴリズム 3.4 に従って要求されるノードに対して行い，要求が生じなくなるまで行った．

図 6 は，入力 50,000 回ごとのクラスラベル付与後の結果を表したものである．新規クラスのデータが入

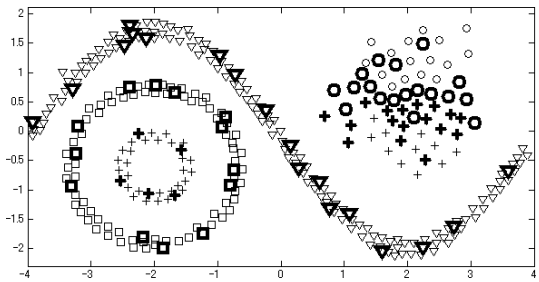

図 5 人工データを用いた定常的な環境での実験結果．生成されたノードを，それらに付与されたクラスラベルごとに異なる記号を用いて表した．また，教師付ノードは大きな記号で表した．
Fig. 5 Experiment results of artificial data under stationary environment. Unlabeled nodes are labeled with different marks, and teacher nodes are labeled with large marks.

力されるごとに，新たな分布を表すノードが生成され，それらに対してクラスラベルが要求されていることが分かる．また，定常的な環境での実験と同様に，クラスの分布が重なる領域にクラスラベルの要求が集中していることが分かる．更にその要求は，クラスの分布の重なりが生じた時点で新たに生じていることが分かる．最終的に，94 個のクラスラベルで全体のノードに適切なクラスラベルが付与された．したがって本実験において，提案手法は生成された 377 個のノード全体に対して 24.9%のクラスラベル，入力データ全体に対して 94/200,000 = 0.047%のクラスラベルで，効率良く学習を行うことができた．

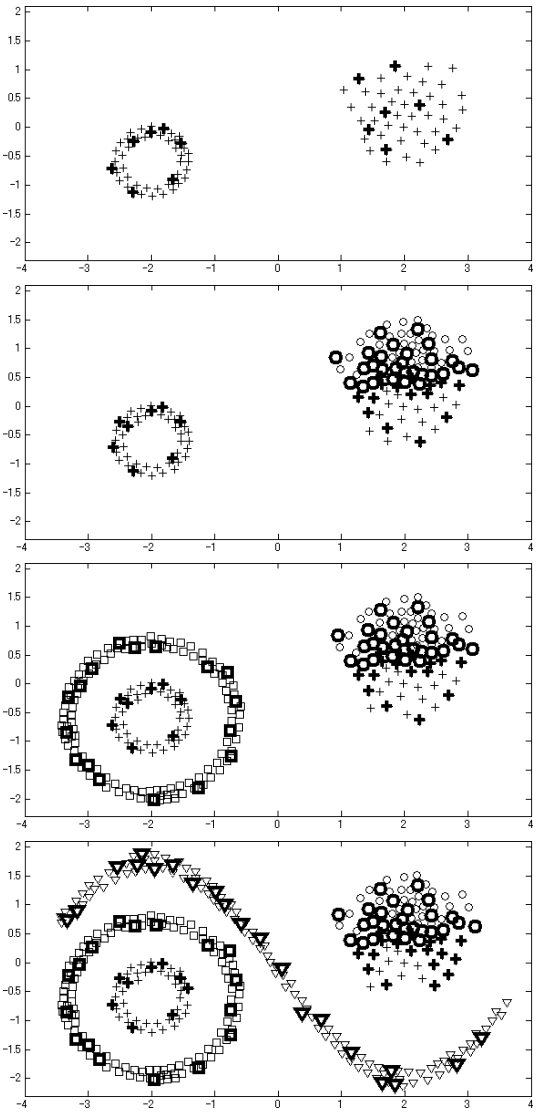

図 6　人工データを用いた追加学習実験結果．生成されたノードを，クラスラベルごとに異なる記号を用いて表した．また，教師付ノードは大きな記号で表した．上から SOINN への入力が 50,000 回，100,000 回，150,000 回，200,000 回後に行われたクラスラベル入力後の結果である．

Fig. 6　Incremental learning result of artificial data. From top to down, show the results after 50,000, 100,000, 150,000, and 200,000 times training.

4.2　実データを用いた実験

次に，入力データとして実データを用いて実験を行った．

4.2.1　最近傍法による全数記憶方式との比較

まず，全数記憶した場合と提案手法によって得られたノードをプロトタイプとした場合の，最近傍法による識別率と教師ラベル数を比較した．データセットは UCI Repository データベースの Pima，WDBC，Iris，Optdigits を用いた [8]．Pima，WDBC，Iris では Leave-one-out 実験の結果を用い，Optdigits では 100 回の試行の平均を用いた．また，それぞれのデータセットに対する，提案手法のパラメータを表 1 に示す．クラスラベルの要求は，Pima，WDBC，Iris では入力 100,000 回，Optdigits では入力 200,000 回で行われるノイズノード削除の後に，アルゴリズム 3.4 で要求されたノードに対してのみ行った．表 2 と表 3 に結果を示す．表 2 より，提案手法によって全数記憶の場合とほぼ同等の識別率が得られていることが分か

表 1　各データに対して提案手法で用いたパラメータ
Table 1　Parameters for different data sets.

	Pima	WDBC	Iris	Optdigits
λ	1,000	400	300	10,000
$\mathrm{age}_{\max}$	30	150	150	100

表 2　実データに対して，全数記憶の最近傍法と，提案手法によって得られたプロトタイプを用いた最近傍法との識別率の比較
Table 2　Recognition ratio comparison.

	Pima	WDBC	Iris	Optdigits
全数記憶	68.0	91.6	96.0	98.0
提案手法	69.4	92.1	96.7	96.5

表 3　実データを用いた実験での，全数記憶と提案手法で用いたラベル付データ数の比較．提案手法は，それぞれの試行で入力したクラスラベル数の平均値である．
Table 3　Needed teacher vectors comparison.

	Pima	WDBC	Iris	Optdigits
全数記憶	767	568	149	3,823
提案手法	78.9	34.2	14.2	147.4

る．また，表 3 より，提案手法に必要な教師数は，全数記憶に比べ大幅に少なくてよいことが分かる．これらのことから，提案手法は全数記憶の最近傍法に比べ，非常に少ない教師数で同等の識別率が得られることが分かった．

文献 [5] では，TSVM と GRF と Logistic GRF の比較実験が行われ，それらの手法で最良の場合，Pima で教師数 70 のとき識別率約 72%，WDBC で教師数 30 のとき識別率約 95.5%という結果が得られている．提案手法はこれらの識別率に数%劣るが，新規クラスの追加学習やオンライン学習が可能であるという優位性を有している．

4. 2. 2 能動学習の有効性の実験

次に Optdigits を用いて，アルゴリズム 3.4 によって能動的にクラスラベルを付与した場合と，ランダムにクラスラベルを付与した場合の比較を行った．Optdigits は，3,823 個の訓練データと 1,797 個のテストデータからなる 64 次元の手書き数字データである．入力データ集合としてすべての訓練データを用いて実験を行った．クラスラベルの入力を，入力 200,000 回で行われるノイズノード削除の後にアルゴリズム 3.4 に従って行った場合と，入力データ集合の中からランダムに選択して行った場合とで比較を行った．パラメータはいずれの場合も $\lambda = 10{,}000$，$\mathrm{age}_{\max} = 100$，$\alpha = 1.0$ とした．

図 7 に結果を示す．結果は 100 回の試行の平均値である．また，縦軸が識別率，横軸が入力したラベル付データ数，実線が能動的にクラスラベルを付与した結果，点線がランダムにクラスラベルを付与した結果を表す．図 7 より，能動学習を行った方は少ないラベル付データで，識別率が大幅に上がっていることが分かる．更にその後も，識別率が徐々に上がっていくことが確認できる．一方，ランダムにクラスラベルを付与した方は，識別率の上がり方が緩やかで，能動学習での識別率を下回っていることが確認できる．この結果より，アルゴリズム 3.4 を用いれば，少ないラベル付データでより効率良く学習できることが分かる．

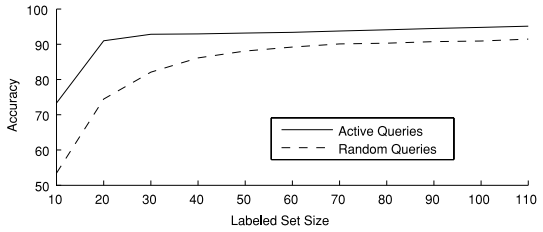

図 7 Optdigits を用いた，ラベル付データの入力を能動的に選択した場合と，ランダムに選択した場合の比較
Fig. 7 Comparison: actively queries or random queries.

4. 2. 3 クラス追加学習の実験

次に，Optdigits を用いて新規クラスの追加学習の実験を行った．入力データ集合を，入力 50,000 回ごとに奇数，偶数，全データの順で変化させ実験を行った．提案手法のパラメータは $\lambda = 10{,}000$，$\mathrm{age}_{\max} = 100$，$\alpha = 1.0$ とした．クラスラベルの要求は 10,000 ステップごとに行われるノイズノード削除の後に，アルゴリズム 3.4 の Step 2. で質問されるノード，すなわちサブクラスタグループの中心ノードに対してのみ行った．

図 8 と図 9 に結果を示す．ここで，結果は 100 回の試行の平均値であり，入力 50,000 回までの識別率は奇数のテストデータのみに対するものである．図 8 より，入力データが偶数に変わった入力 50,000 回の時点で，教師の数が大幅に増えていることが分かる．また，図 9 より，ある程度の識別率を保っていることが分かる．これは，システムが入力は未知のクラスであるか判定し，必要な教師ラベル数を自律的に決定できることを示すものである．また，入力 50,000 回ま

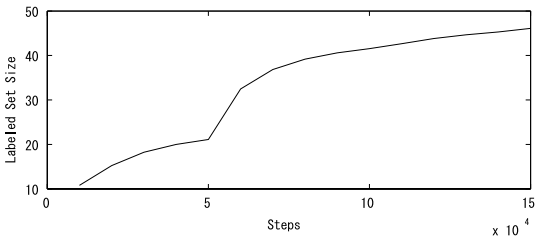

図 8 Optdigits を用いた，新規クラスの追加学習実験による入力回数とラベル付データ数の関係
Fig. 8 Incremental learning: needed number of teacher vectors.

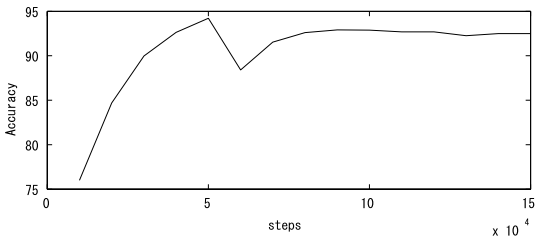

図 9 Optdigits を用いた，新規クラスの追加学習実験による入力回数と識別率の関係
Fig. 9 Incremental learning: recognition ratio.

でに識別率が上がっていくのは，SOINN での学習が進み，入力データの分布をより良く近似できるようになったためであると考えられる．

4.2.4 顔画像を用いたオンライン学習実験

最後に，実環境のデータに対するオンライン学習の実験を行った．実環境のデータとして，固定されたビデオカメラで撮影した 10 人分の顔画像を用いた（図 10）．用いた画像は，1 秒間 30 フレームで撮影された，20×26 ピクセルのグレースケール 256 階調の画像である．撮影中，被撮影者は顔の方向を少しずつ変化させている（図 11）．1 人当り 3,000 フレーム分の連続的な画像，計 30,000 枚の画像を訓練データとして用いた．また，訓練データとは別に撮影した，1 人当り 700 フレームの画像をテストデータとして用いた．提案手法のパラメータは $\lambda = 2{,}000$，$\mathrm{age}_{\max} = 200$，$\alpha = 1.0$ で実験を行った．入力データ集合は，λ のサイズの First-in-first-out のバッファに画像を保存したものを用いた．つまり，入力データ集合はカメラからの新たな入力があるごとに，バッファ内で最も前に入力されたデータを破棄し，新たに入力されたデータを保持するという方法で更新される．また，クラスラベルの要求は入力 2,000 回ごとに行われるノイズノード削除の後に，アルゴリズム 3.4 の Step 3. で要求されるノード，すなわちサブクラスタの中心ノードに対してのみ行った．

図 12 と図 13 に結果を示す．結果は 100 回の試行の平均で，識別率は入力されたことがあるクラスのテストデータのみに対するものである．また，図 12 の縦軸は入力されたラベル付データ数で横軸は入力回数，図 13 の縦軸は識別率で横軸は入力回数を表す．図 13 より，新規クラスのデータが入力されるごとに識別率が下がるが，学習が進むことで再び識別率が上がっていることが確認できる．また，入力 30,000 回後のノード数の平均は 353.19 であった．一方，この実験で用いたデータを全数記憶する場合，30,000 のデータを蓄えておく必要がある．したがって提案手法では，必要なメモリ量や最近傍法による識別の計算量を 1.2%程度に抑えることができるといえる．更に，入力された画像に対して，カメラから次の画像が入力されるまでにノードの生成は終了する．つまり提案手法では，カメラからの動画像に対して実時間での学習が可能である．このように，カメラで撮影したような，実環境の多数高次元データに対しても，識別率を大幅に下げることなくオンラインで学習できていることが分かる．

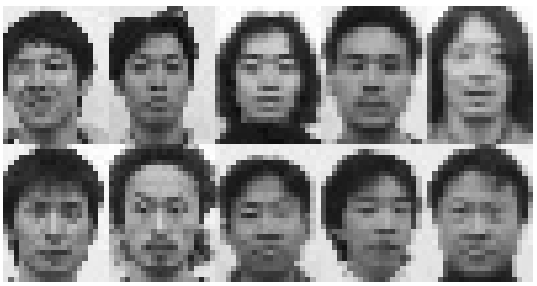

図 10　実験に用いた 10 人分の顔画像
Fig. 10　An example of facial images of each subject.

図 11　実験に用いた顔画像．顔の方向が少しずつ変化する
Fig. 11　An example of face sequences.

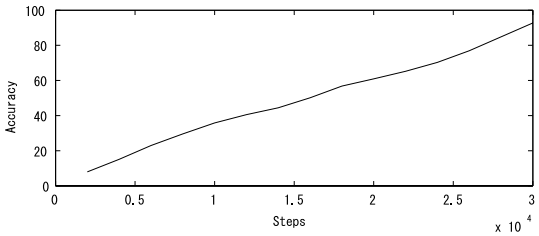

図 12　顔画像データを用いた実験による入力回数とラベル付データ数の関係
Fig. 12　Face recognition: needed number of teacher vectors.

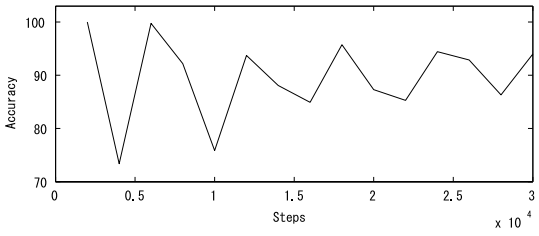

図 13　顔画像データを用いた実験による入力回数と識別率の関係
Fig. 13　Face recognition: recognition ratio.

5. むすび

本論文ではSOINNを拡張し，オンライン準教師付能動学習手法を提案した．提案手法は，ノード数やクラス数などの事前知識を必要としない．また，必要な教師数を自律的に獲得することで永続的な学習を行うことができる．更に，入力データの中に分布するノイズを除去することが可能であるという特長をもつ．

計算機実験によって，提案手法の有用性を示した．人工データを用いた実験では，データ中に存在する10%程度のノイズを除去し，入力データの分布とそのクラスラベルを適切に表現できることを示した．次に，実データを用いて，提案手法は全数記憶の最近傍法と同程度の識別率で，ラベル付データ数を大幅に削減できることを示した．また，必要なラベル付データやその数を自らで決定することができ，新たなデータに対しても事前知識なしで追加的に学習できることを示した．更に，実環境のデータに対してもオンラインで学習できることを示した．しかし，従来のバッチ学習の手法に比べ，提案手法は識別率で数%劣る．それらの手法の識別率に近づけることが今後の課題である．

謝辞 本研究の実施にあたりNEDO産業技術研究助成事業から支援を頂きました．記して感謝致します．

文　献

[1] K. Bennett and A. Demiriz, “Semi-supervised support vector machines,” Advances in Neural Information Processing Systems, vol.11, pp.368–374, 1999.

[2] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” Proc. Conference on Computational Learning Theory, pp.92–100, 1998.

[3] B. Fritzke, “A growing neural gas network learns topologies,” Advances in Neural Information Processing System, vol.7, pp.625–632, 1995.

[4] T. Kohonen, “Self-organized formation of topologically correct feature maps,” Biol. Cybern., vol.43, no.1, pp.59–69, 1982.

[5] B. Krishnapuram, D. Williams, Y. Xue, A. Hartemink, L. Carin, and M. Figueiredo, “On semi-supervised classification,” Advances in Neural Information Processing Systems, vol.17, pp.721–728, 2004.

[6] T.M. Martinetz, S.G. Berkovich, and K.J. Schulten, ““Neural-Gas” network for vector quantization and its application to time-series prediction,” IEEE Trans. Neural Netw., vol.4, no.4, pp.558–569, July 1993.

[7] I. Muslea, S. Minston, and C. Knoblock, “Active + semi-supervised learning = robust multi-view learning,” Proc. ICML-02, 19th International Conference on Machine Learning, pp.435–442, 2002.

[8] C. Merz and M. Murphy, UCI Repository of machine learning database, University of California Department of Information, Irvine, CA, 1996.

[9] K. Nigam, A.K. McCallum, S. Thrum, and T. Mitchell, “Text classification from labeled and unlabeled documents using EM,” Mach. Learn., vol.39, pp.103–134, 2000.

[10] 小倉和貴，申　富饒，長谷川修，“オンライン教師なし分類のための追加学習手法,” 信学論（D），vol.J90-D, no.6, pp.1610–1622, June 2007.

[11] F. Shen and O. Hasegawa, “An incremental network for on-line unsupervised classification and topology learning,” Neural Netw., vol.19, no.1, pp.90–106, 2006.

[12] X. Zhu, Z. Ghahramani, and J. Lafferty, “Semi-supervised learning using Gaussian fields and harmonic functions,” Proc. ICML-03, 20th International Conference on Machine Learning, pp.912–919, 2003.

[13] X. Zhu, J. Lafferty, and Z. Ghahramani, “Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions,” ICML 2003 Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, pp.58–65, 2003.

（平成19年2月20日受付，6月28日再受付）

櫻井　啓介

2005東工大・工・情報工学卒．2007同大大学院総合理工学研究科知能システム科学専攻修士課程了．現在，アップルジャパン（株）に勤務．

神谷　祐樹　（学生員）

2003東工大・工・電気電子卒．2005同大大学院総合理工学研究科知能システム科学専攻修士課程了．現在，同大学院総合理工学研究科知能システム科学専攻博士後期課程在籍中．

長谷川　修　（正員）

1993東京大学大学院電子工学専攻博士課程了．博士（工学）．同年電子技術総合研究所，1999から1年間米国カーネギーメロン大学客員研究員，2001産業技術総合研究所主任研究員，2002東京工業大学像情報工学研究施設助教授，2003〜2006 JST さきがけ研究21研究者（兼任），情報処理学会，日本認知科学会，人工知能学会，IEEE-CS等各会員．