論　文

自己増殖型ニューラルネットワークを用いたプロトタイプ生成による高速最近傍識別器の構成手法

神谷　祐樹[†a)]　　申　富饒[††b)]　　長谷川　修[†††c)]

A Fast Prototype-Based Nearest-Neighbor Classifier with Self-Organizing Incremental Neural Network

Youki KAIMYA[†a)], Furao SHEN[††b)], and Osamu HASEGAWA[†††c)]

あらまし　本論文では，プロトタイプに基づく高速識別器の構成手法，Adjusted SOINN Classifier（ASC）を提案する．提案手法は自己増殖型ニューラルネットワーク，adjusted SOINN に基づいている．提案手法は，識別境界の決定に必要なプロトタイプ数を自律的に学習し，追加的な学習データに容易に対処することができる．また，ノイズの影響を受けたプロトタイプを削除可能であり，ノイズの多い学習データに対しても頑健に学習可能である．更に，識別に不要なプロトタイプを削除し，高速識別器を構成することが可能である．独自に作成した人工データセットを用いた実験により，提案手法の学習の性質を示した．また，UCI データセットを用いた実験を行い，従来の様々なプロトタイプに基づく識別手法と比較した結果，ASC が最も高い識別性能を達成し，プロトタイプ削減効率も高い能力を示した．

キーワード　最近傍識別器，自己増殖型ニューラルネットワーク，高速化，パターン認識

1. まえがき

k-最近傍法（以下，k-NN）[1] は，機械学習，データマイニング，自然言語処理，情報検索など様々な分野で広く用いられている [2]．k-NN では，入力パターンと距離の近い順に k 個のプロトタイプを選択し，それらプロトタイプのクラスラベルから多数決により入力を識別する．このような識別器を最近傍識別器（NNC(k)）と呼ぶ [1]．$k = 1$ とした場合，1-最近傍法（1-NN）となる．

† 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 Systems Science, Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, Yokohama-shi, 226–8503 Japan
†† 中国南京大学計算機ソフトウェア新技術国家重点実験室，中国
The State Key Laboratory for Novel Software Technology, Nanjing University, China
††† 東京工業大学大学院理工学研究科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: frshen@nju.edu.cn
c) E-mail: hasegawa.o.aa@m.titech.ac.jp

k-NN は（1）少量のデータから学習可能，（2）学習データを容易に追加可能，（3）最適化や複雑な識別関数のモデル化などを必要としない，などの利点がある．しかし一方で，全数記憶方式を採用する k-NN では，大規模データを用いた際の識別計算量が問題とされる．そのため，プロトタイプ数を削減することで，メモリ使用量の節約が図られる．したがって，プロトタイプに基づく識別器の構成において，学習データセットから識別に必要なプロトタイプ集合をいかにして求めるかが問題となる．

プロトタイプに基づく識別器の構成手法は従来から多く提案されている．NNC が一般的に学習データすべてをプロトタイプとする全数記憶方式で識別を行うのに対し，学習データの一部をプロトタイプとして選択する Editing [3] 及び Condensing [4] の手法がある．これらは学習データの一部を選択するため，NNC の識別性能を低下させる可能性がある．和田らは，学習データから識別境界の決定に関係するプロトタイプのみを選択する direct condensing [5] を提案した．この手法は，NNC の識別性能を低下させることなくプロトタイプ数を削減可能であるが，学習データが 20

次元程度を超える高次元の場合は，選択に要する計算量の観点から利用することは難しい [6].

一方，学習データの分布などの情報からプロトタイプを求める手法がある．Nearest Mean Classifier（NMC）[7] は，各クラスの平均，すなわち各クラスに対して一つのプロトタイプを用いた識別を行う．NMC では一般的に学習データ，テストデータともに多くの誤認識が生じる．ほかに，k-means classifier（KMC）[7], Learning Vector Quantization（LVQ）[8]，その他の手法 [9], [10] などがある．これらの手法は，過学習を回避する戦略に基づいて，各クラスについて同一定数のプロトタイプを選択する．しかし，各クラスのデータ分布や密度，及びサンプル数などが異なる場合，最適なプロトタイプ数が各クラスで異なる可能性は十分に考えられる．また，Nearest Subclass Classifier（NSC）[11] では，分散抑制パラメータを導入し，各クラスのプロトタイプ数の制限が試みられている．しかし，彼らはすべてのパターンの特徴は同程度のノイズを含み，教師ラベルのノイズはないと仮定している．また，学習データのアンダサンプリングの程度は特徴空間内で一様であると仮定しているが，実用的な学習タスクでは，これらの仮定は必ずしも成り立たない．

更には，いくつかのプロトタイプに基づく識別器構成手法では，実時間の追加学習やノイズの除去が困難である．ここで，ノイズとは学習データのパターン情報及びクラスラベルに影響を及ぼすものとし，その内容・程度は把握できないものと定義する．

そこで本論文では，高速最近傍識別器の構成手法を提案する．提案手法では，自己増殖型ニューラルネットワーク（SOINN）[12], [13] に基づいてプロトタイプ抽出を行う．提案手法は以下の特徴，（1）各クラスの分布構造に基づいて，各々の分布に必要な数のプロトタイプを自律的に配置可能，（2）追加学習可能，（3）ノイズの多い学習データに対処可能，（4）識別時に不要なプロトタイプを削除可能，をもつ．

次章では，提案手法 Adjusted SOINN Classifier（ASC）について詳述する．**3.** では，人工データを用いた実験によって提案手法の性質を示し，その後 UCI データセットを用いて他手法との比較実験を行い，識別率及びメモリ使用率に関する提案手法の性能を評価する．

なお本論文では，すべての実験において特に断りのない限り，識別手法に 1-NN 法を採用している．

2. Adjusted SOINN Classifier（ASC）

提案手法である，Adjusted SOINN Classifier（ASC）は，自己増殖型ニューラルネットワーク（SOINN）[12], [13] の性質を継承している．そのため，追加学習，ノイズへの耐性，プロトタイプ数の自律的決定が可能である．しかし，SOINN は教師なし学習による入力の分布の近似を目的としており，教師あり学習や，高速な識別のためのプロトタイプ集合の抽出を考慮したものではない．また，学習に必要なパラメータ数が多いことや，学習データの入力順に学習結果が影響され安定しないことなどが問題とされる．そこで，SOINN を以下の点で改良する．

（1） より少ないパラメータで入力の分布を近似する（adjusted SOINN）

（2） k-means 法 [7] を導入し学習結果の安定性を高める

（3） ノイズの影響を受けたプロトタイプを削除する（noise-reduction）

（4） 識別時に不要なプロトタイプを削除する（center-cleaning）

ASC の学習フローを図 1 に示す．図 1 に示したように，ASC は各クラスの入力に対して独立に adjusted SOINN と k-means 法とを用いる．その後，全クラスの学習結果を用いてノイズ除去（noise-reduction）を行

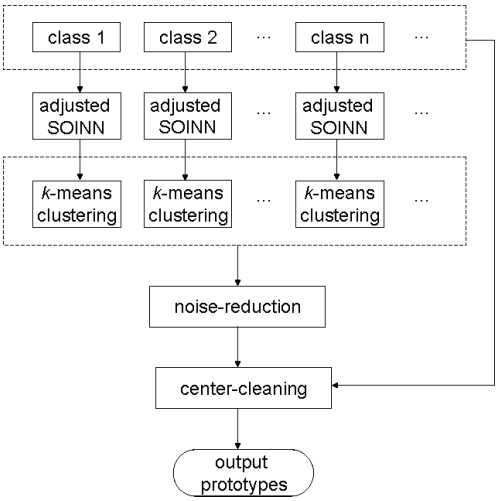


図 1 Adjusted SOINN Classifier（ASC）の学習フロー
Fig. 1 Learning process of the Adjusted SOINN Classifier (ASC).

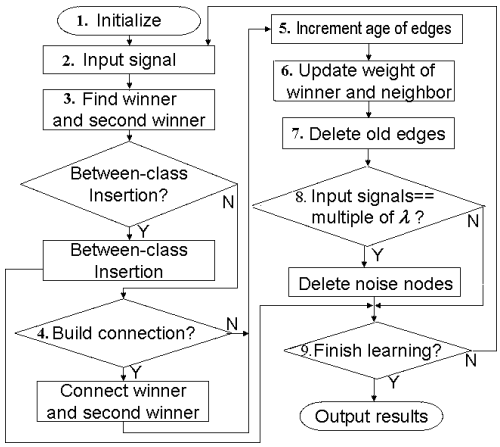


図 2 Adjusted SOINN のフローチャート
Fig. 2 Flowchart of adjusted SOINN.

う．そして最後に，ノイズ除去の結果と全入力データとを用いて不要なプロトタイプの削除（center-cleaning）を行う．

2.1 Adjusted SOINN

adjusted SOINN の主な働きは，オンライン学習によってノードを徐々に増殖させ，各ノード間の隣接関係をエッジ（辺）を用いて構成し，入力の分布を近似することである．また，ある程度ノイズの影響を除去することができる．図 2 に adjusted SOINN のフローチャートを示す．adjusted SOINN では，各入力に対して 4 種類の処理,（1）ノードの挿入,（2）エッジの挿入・削除,（3）ノードの位置ベクトル更新,（4）ノードの削除，を必要に応じて行う．

ノードの挿入は，近似されていない領域への入力に対して実行される．挿入の必要性の判断は，既存のネットワークの各ノードがもつ類似しきい値 T に基づいて行う．ノード挿入の例を図 3 に示す．入力パターンと勝者ノード及び第二勝者ノードとの距離がそれらのノードの類似しきい値 T を超える場合，入力パターンを新たなクラスタに属すると判断する．その際，入力パターンを新ノードとしてネットワークに挿入する．

類似しきい値 T_i は，アルゴリズム *2.1* のように定義する．入力の分布を近似するために，入力に対してノードの位置ベクトルの更新処理（後述）が行われる場合がある．そのため，ノード i の位置は随時変化するので，類似しきい値 T_i の値もそれに伴い変化する．

アルゴリズム *2.1*：類似しきい値 T の計算方法

（1） ノード i の類似しきい値を，ノードの生成時に $+\infty$ に初期化する．

（2） ノード i が勝者ノードまたは第二勝者ノードである場合，T_i を更新する．

- ノード i に隣接ノードが存在する場合，T_i をノード i から最も遠い隣接ノードとの距離値に更新する．

$T_i = \max_{c \in N_i} \|W_i - W_c\|$，　ただし N_i は，ノード i の隣接ノード集合を表す．

- ノード i に隣接ノードが存在しない場合，T_i をノード i から最も近い他のノードとの距離値に更新する．$T_i = \min_{c \in A \setminus \{i\}} \|W_i - W_c\|$，ただし A はノード集合を表す．

本論文では，勝者ノードとは入力パターンの最近傍ノードを意味し，第二勝者ノードとは，入力パターンに 2 番目に近いノードを意味する．W_i はノード i の位置ベクトルを表す．

エッジの挿入・削除及びノードの位置ベクトルの更新は，既存のネットワークを入力の分布の形状に適応させるためにそれぞれ実行される．エッジの挿入・削除は，ノード間の隣接関係の構成を目的としている．その基準には，competitive Hebbian rule（CHL）[14] を採用している．CHL は，各入力に対してユークリッド距離基準で最も近い二つのノード（勝者ノードと第二勝者ノード）をエッジで連結する．その結果，入力パターンの生起確率が 0 より大きい領域にのみエッジが形成される（図 4 (b)）．構成されるネットワークはドローネーグラフ（図 4 (a)）の部分グラフを表している．CHL によって得られたグラフは分布の形状を有効に近似する [14]．一方，ノードの位置ベクトルの更新は，ネットワークの入力パターンへの局所的な適応を目的としている．各入力パターンに対して，勝者

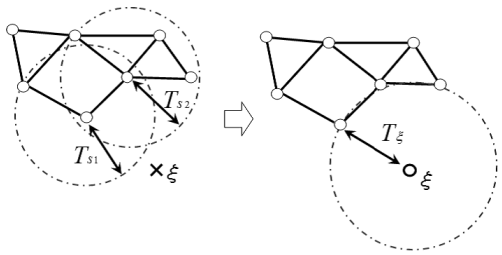
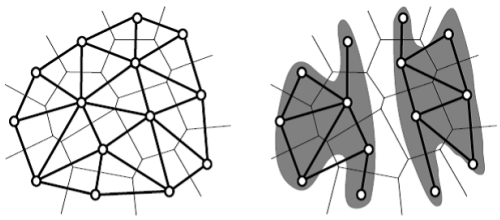


図 3 ノードの挿入処理：入力パターン ξ と勝者ノード s_1 及び第二勝者ノード s_2 との距離が類似しきい値 T_{s_1}, T_{s_2} より大きい場合（左），入力 ξ を新たなノードとして挿入する（右）．新ノードの類似しきい値 T_ξ は勝者ノードとの距離で表される．
Fig. 3 Between-class insertion process.



(a) Delaunay triangulation　(b) induced Delaunay triangulation

図 4　ノードの隣接関係の定義：CHL を用いた場合，入力データの生起確率が 0 より大きい領域（灰色領域）に隣接関係を形成する (b)．得られたグラフ (b) は，ドローネーグラフ (a) の部分グラフとなる [15]．

Fig. 4　Two ways of defining closeness among a set of points. (a) The *Delaunay triangulation* (thick lines) connects points having neighboring *Volonoi polygons* (thin lines). (b) The *induced Delaunay triangulation* (thick lines) is obtained by masking the original Delaunay triangulation with a data distribution P(ξ) (shaded). [15]

ノードとそれに連結する隣接ノードの位置ベクトルが更新される．adjusted SOINN では，入力パターンへの適応を収束させるために，ノードの位置ベクトルの更新の程度は，該当ノードが勝者ノードに選択された回数に反比例するように設定している．

以上の処理を併せてオンライン学習を行った場合，ノードの位置は入力への適応のために変化するため，初期の学習で構成した隣接関係が以後の学習によって不適切になる可能性がある．そのため，エッジの加齢処理（*edge aging* scheme [15]）を用いて，参照されなくなったエッジを削除して対応する．具体的には各入力に対して，勝者ノードに連結するすべてのエッジを加齢し，その一方で勝者ノードと第二勝者ノードの間のエッジの年齢を 0 に更新する．そして，事前定義するしきい値 a_d を超える年齢になったエッジを削除する．ノードの移動によって不適切となったエッジは，隣接関係が成り立たないため，エッジの年齢が更新されずに削除される．

ノードの削除は，ノイズの影響を受けたノードを削除するため実行される．ノードの削除基準は，隣接ノード数が 1 以下のノードがノイズの影響を受けたノードである可能性が高いという仮定に基づいている．したがって，事前定義パラメータ λ を定義し，入力パターン数が λ の倍数となるごとに全ノードを評価し，削除基準を満たすノードを削除している．

以上の処理を各入力に対して行うことで，adjusted SOINN は入力の分布からある程度のノイズを除去して近似する．アルゴリズム *2.2* に adjusted SOINN の詳細を示す．アルゴリズムの各ステップは図 2 の各要素の番号に対応している．adjusted SOINN による学習結果は次章の実験で示す．

アルゴリズム *2.2*: Adjusted SOINN

（1） ノード集合 A を，学習データからランダムに選択した位置ベクトルをもつ二つのノード（$A = \{c_1, c_2\}$ ）に初期設定する．また，エッジ集合 C（$C \subset A \times A$）は空集合とする．

（2） $\xi \in R^n$ を入力パターンとする．

（3） 入力パターンに対する勝者ノード（*winner*）s_1 と第二勝者ノード（*second winner*）s_2 を探索する．

$$s_1 = \arg\min_{c \in A} \|\xi - W_c\| \tag{1}$$

$$s_2 = \arg\min_{c \in A \setminus \{s_1\}} \|\xi - W_c\| \tag{2}$$

入力パターン ξ とノード（s_1 または s_2）との距離が類似しきい値（T_{s_1} または T_{s_2}）より大きい場合，入力パターンを新ノードとして A に追加する．その後，新しい入力パターンの学習のためにステップ（2）に戻る．類似しきい値 T はアルゴリズム *2.1* で算出される．

（4） s_1 と s_2 との間のエッジが存在しなければ，新たに作成して C に追加する．存在する場合は該当するエッジの年齢を 0 にリセットする．

（5） s_1 につながるすべてのエッジの年齢をインクリメントする．

（6） 勝者ノードと勝者ノードに隣接するノードの位置ベクトルを，以下の式を用いて更新する．ただし，係数 ϵ_1 及び ϵ_2 を，$\epsilon_1(t) = 1/t$，$\epsilon_2(t) = 1/100t$，また，t を該当ノードが勝者ノードに選択された回数，と定義する．

$$\Delta W_{s_1} = \epsilon_1(t)(\xi - W_{s_1}) \tag{3}$$

$$\Delta W_i = \epsilon_2(t)(\xi - W_i) \quad (\forall i \in N_{s_1}) \tag{4}$$

（7） 事前定義したしきい値 a_d を超える年齢のエッジを削除する．その結果，隣接関係をもたないノードが現れた場合は，該当するノードを削除する．

（8） 入力パターン数が λ の倍数となった場合，隣接ノード数が 1 以下のノードを削除する．

（9） 学習が十分に行われるまで，ステップ（2）に戻り学習を繰り返す．

アルゴリズム *2.2* では，二つのパラメータ（a_d，λ）

の設定が必要である．この二つのパラメータは，ノードの隣接関係を表すエッジ及び疎の領域に存在するノードを削除する頻度に影響する．学習した情報を長く残したい場合は，二つのパラメータを大きな値に設定する．反対に，メモリ使用量を節約したい場合は，ノード数を少なくするためにパラメータを小さな値に設定し，頻繁にノードやエッジを消す．つまり，二つのパラメータの設定によって，ASC の識別能力を調節することができる．

adjusted SOINN は，SOINN の第一層を手法の基盤としているが，クラス内のノード挿入の機能（within-class insertion [13]）を除くことで，手法を平易にし，SOINN のもつ五つのユーザ定義パラメータを削除した．この修正は，近似性能にはほとんど影響しない．前述のノード挿入の機能のみで，入力の分布を表現する十分な数のノードが保障される．類似しきい値 T の定義によって，近似されていない領域への入力に対してノード挿入が生じる．その後それらの挿入されたノードは competitive Hebbian rule によって他のノードと連結する．類似しきい値 T はノードの位置ベクトルの更新に伴って適応的に更新されるため，しきい値が大きなままでノードの分布が疎になることはない．またノードがクラスの識別境界付近まで分布するようになると，類似しきい値によるノード挿入は自動的に生起しなくなる．したがって，Growing Neural Gas（GNG）[17] に見られるようなノードの永続的な増加は避けられる．

2.2 k-means 法

k-means 法 [7] は，教師なしデータ集合からクラスタとそのクラスタの重心を探索する手法である．クラスタの数を m 個に設定した場合，k-means 法では以下の二つの手順，（1）学習データの各サンプルに対して最近傍のクラスタの重心を算出し，学習データを m 個のクラスタに分割する，（2）各クラスタに含まれる学習サンプルのベクトルの平均値を算出し，新たなクラスタ重心に採用する，を収束するまで繰り返し行う．

ASC では，k-means 法で事前に決定すべきクラスタ数とクラスタの重心の初期値として，adjusted SOINN の学習したノードの数をクラスタ数に設定し，ノードの位置をクラスタの重心の初期値に設定している．

ASC では，adjusted SOINN の結果が学習データの入力順序に影響されるため，同一環境であっても入力順序によってノードの数やその位置は異なる．そこで，k-means 法を用いてそれらのノード位置を移動させることで ASC の安定化を図っている．adjusted SOINN によって生成されたノードは入力データの分布を表しているため，各ノードはサブクラスタの重心付近に位置する．その後，k-means 法を用いて，それらのノードをサブクラスタの重心に移動する．

2.3 ノイズの影響のあるプロトタイプの削除（noise-reduction）

ノイズの多い学習データの場合，adjusted SOINN と k-means 法の過程では，いくつかのノードがノイズによって生成される可能性がある．そこで，ノイズの多い学習データへの適合を防止するために，従来手法である k-Edited Neighbors Classifier（ENC）[16] のアイデアを採用する．すなわち，あるノードのラベルが k 個の隣接ノードの多数決ラベルと異なる場合，そのノードは異常値とみなしプロトタイプ集合から削除する．このノイズ除去過程では，パラメータ k を調節するために cross-validation などを用いる．

ただし，このノイズ除去過程は学習データに含まれるノイズの影響の状況によって利用しない場合がある．具体的には，ノイズが少量であり，adjusted SOINN の学習の際に既にノイズが除去されていると予想される場合，このノイズ除去過程によって識別に重要なノードを削除する可能性があるため利用しない．この点については，**3.** の実験で詳述する．

2.4 不要なプロトタイプの削除（center-cleaning）

adjusted SOINN と k-means 法の過程を経て，入力データの分布を近似したプロトタイプ集合が得られる．しかし，識別過程を考えると，不要なプロトタイプが存在する可能性がある．すなわち識別の際に 1-NN 法を用いるので，識別境界付近のプロトタイプのみが必要になる．メモリ使用量の節約のために，不要なプロトタイプの削除は重要である．したがって，アルゴリズム *2.3* を定義して不要なプロトタイプの削除を行った．アルゴリズム *2.3* は，あるクラスのプロトタイプが他のクラスのどのプロトタイプからも最近傍でない場合，そのプロトタイプが識別に不要な部分にあるという考えに基づいている．

アルゴリズム *2.3*: Center-cleaning

（1） n クラスが存在する場合，各クラス i（$i = 1, \ldots, n$）について以下の手順を行う．

（2） クラス i 以外のクラスのすべてのプロトタイプについて，クラス i の最近傍プロトタイプを選択する．

（3） クラス i のプロトタイプについて，選択されなかったプロトタイプが存在する場合は，該当するプロトタイプを削除する．プロトタイプが削除されなくなるまでこの手順を繰り返す．

2.5 ASC アルゴリズム

2.1〜**2.4** の分析に基づいて，adjusted SOINN classifier（ASC）全体のアルゴリズムをアルゴリズム *2.4* に示す．

アルゴリズム *2.4*: ASC の学習アルゴリズム

（1） 学習するクラスを n と仮定した場合，各クラスの学習データについて独立に adjusted SOINN（アルゴリズム *2.2*）を行う．アルゴリズム *2.2* の学習結果のノードをプロトタイプとすることで，各クラスのプロトタイプ数 $\{N_i, i = 1, \ldots, n\}$，及びそれらのプロトタイプの位置ベクトル $\{SOINN\_Prototype(c_j, i), j = 1, \ldots, N_i; i = 1, \ldots, n\}$ を得る．

（2） 各クラスについて独立に k-means 法を行う．各クラスのクラスタの重心の数は異なる可能性があるが，ステップ（1）で得られた N_i をクラス i のクラスタ重心の数に設定する．また，$\{SOINN\_Prototype(c_j, i), j = 1, \ldots, N_i\}$ をクラス i のクラスタ重心の初期位置に設定する．このステップで得られた結果を $\{k - means\_Prototype(c_j, i), j = 1, \ldots, N_i; i = 1, \ldots, n\}$ とする．

（3） プロトタイプ c（$c \in k - means\_Prototype(c_j, i)$, $j = 1, \ldots, N_i; i = 1, \ldots, n$）に関して，$k$ 個の最近傍プロトタイプを探索する．探索した k 個のプロトタイプの過半数がプロトタイプ c と異なるクラスラベルをもつ場合，プロトタイプ c を削除する．すべてのプロトタイプについて同様の処理を行い，$\{Noise\_Reduction\_Prototype(c_j, i), j = 1, \ldots, \hat{N}_i; i = 1, \ldots, n\}$ を得る．ただし，$\hat{N}_i$ はクラス i の新たなプロトタイプ数（$\hat{N}_i \leq N_i$）である．

（4） $\{Noise\_Reduction\_Prototype(c_j, i), j = 1, \ldots, \hat{N}_i; i = 1, \ldots, n\}$ の全クラスのプロトタイプに関して，アルゴリズム *2.3* を用いて識別に不要なプロトタイプを削除する．その結果，最終的なプロトタイプ集合 $\{Prototype(c_j, i), j = 1, \ldots, M_i; i = 1, \ldots, n\}$ を得る．ただし，M_i は各クラスのプロトタイプ数（$M_i \leq \hat{N}_i$）である．

上記のアルゴリズムでは，adjusted SOINN 内で二つのパラメータ (a_d, λ) を必要とする．また，ステップ（3）のノイズ除去の過程でパラメータ k を用いる．これらのパラメータについては **2.1** 及び **2.3** で詳述したとおりである．

一般的にプロトタイプに基づく識別手法の性能を評価するためには，識別率及びメモリ使用量が用いられる．本論文では，学習後のメモリ使用量の節約度合として，プロトタイプ効率 $r_c = \frac{\text{プロトタイプ数}}{\text{学習サンプル数}}$ を定義し，識別率と r_c を識別器の性能の評価基準とする．

3. 評価実験

本章では，まず，独自に作成した二次元人工データセットを用いて ASC の性能を評価する．その後，UCI repository [18] に含まれるデータセットを用いた実験を通して ASC の有効性を示す．最後に ASC の性能を従来のプロトタイプに基づく識別器構成手法と比較する．

3.1 人工データセットを用いた実験

すべての人工データセットの実験において，パラメータの各値を $a_d = 20$，$\lambda = 20$，$k = 5$ に設定した．a_d と λ は生成されるプロトタイプ数，すなわちプロトタイプ効率に影響するが，十分なプロトタイプ数が得られた場合は識別率には影響しない．パラメータ k は影響力は弱く，他の値でも同程度の識別性能を得られる．また，本節のすべての実験において，学習及び識別を 10 回試行し，識別率とプロトタイプ効率の平均値を算出して提案手法の性能を評価した．

- 実験 1：オーバラップのない二つの正規分布

この実験では，学習データの分布をオーバラップのない 2 クラスの正規分布（図 5）に設定した．各クラスの学習データ数は 2,000，テストデータ数は 200 サンプルに設定し，分布からランダムに選択した．したがって，合計の学習データ数及びテストデータ数はそれぞれ 4,000，400 である．この条件において全数記憶方式の最近傍識別器を用いた場合の識別率は 100%である．

学習データを ASC のアルゴリズム *2.4* に従って学習させた．図 6 に，adjusted SOINN の学習結果を示す．この図から，adjusted SOINN が 2 クラスの入力データの分布を十分に表現していることが分かる．図 7 は ASC の学習結果を示している．この図から，ASC の学習によって識別時に 6 個（各クラス 3 個）のプロトタイプのみが生成されることが分かる．この実験の場合，ASC は平均 6.1 ± 0.3 個のプロトタイプを生成した．また，生成したプロトタイプによってテストデータの識別を行った場合，$99.9 \pm 0.1\%$ の識別率

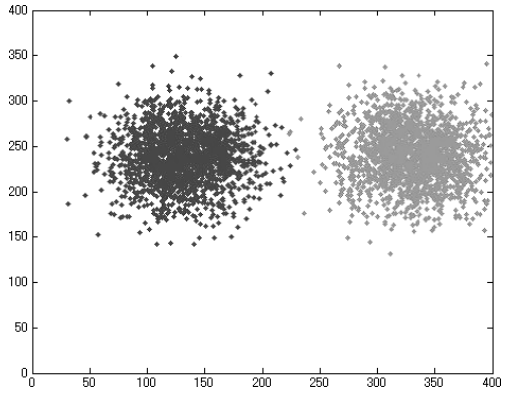


図 5　学習データ：オーバラップのない二つの正規分布（2 クラス）
Fig. 5　Two Gaussian distribution data sets, without overlap (2 classes).

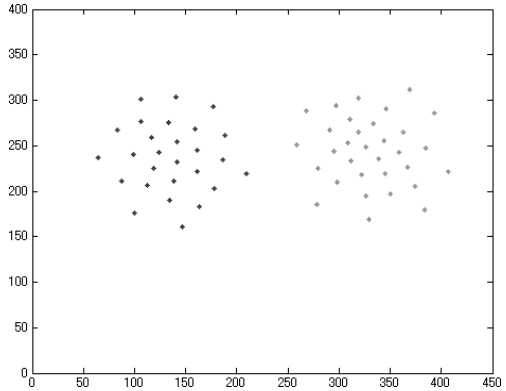


図 6　図 5 のデータに対する Adjusted SOINN の学習結果
Fig. 6　Adjusted SOINN results of Fig. 5.

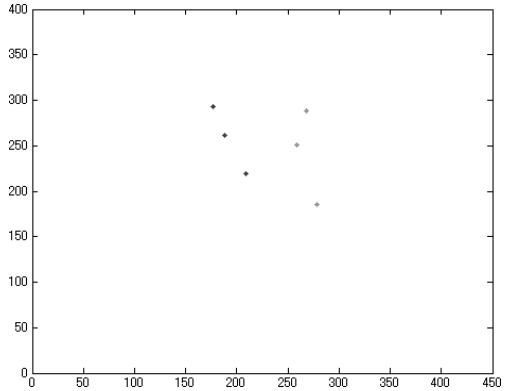


図 7　図 5 のデータに対する ASC の学習結果
Fig. 7　ASC results of Fig. 5.

が得られた．全数記憶方式の最近傍識別器と比べて，ASC は 0.15 ± 0.01% のプロトタイプ数（プロトタイプ効率 $r_c = 0.15 \pm 0.01\%$）で同等の性能を得ることができる．

- 実験 2：オーバラップのない 5 クラスの分布

この実験には，学習データの分布に，実験 1 から更に 3 クラス加えた分布（図 8）を用いた．実験データは，各クラスを様々な分布形状（正規分布，リング及びサインカーブ）に設定した．この分布から各クラスの学習データを 2,000 サンプル，テストデータを 200 サンプル，ランダムに選択した．したがって，合計の学習データ数及びテストデータ数はそれぞれ 10,000, 1,000 である．この条件において全数記憶方式の最近傍識別器を用いた場合の識別率は 100% であった．

図 9 は adjusted SOINN の学習結果を示している．この学習結果は入力データの分布を十分に表現している．図 10 に ASC の学習結果を示す．正規分布のクラスに関するプロトタイプ数が実験 1 より増えているが，これは別のクラスの分布が追加されたので，識別境界を決定するためにより多くのプロトタイプが必要となったためと考えられる．リング及びサインカーブの分布のクラスに関しても同様に，プロトタイプが識別境界の決定に必要な位置に自動的に生成されている．更に，ASC が分布の大きさや形状に合わせて各クラスに異なる数のプロトタイプを生成していることが，図 10 に示されている．この実験の場合，ASC は平均 69.6 ± 2.6 個のプロトタイプを生成し，100 ± 0.0% の識別率を示した．したがって，ASC は全数記憶方式の最近傍識別器に比べ，0.70 ± 0.03% のプロトタイプ数（$r_c = 0.70 \pm 0.03\%$）で同等の性能を示した．

- 実験 3：オーバラップを含む 5 クラスの分布

この実験では，実験 2 の学習データの分布から，更に正規分布の 2 クラスにオーバラップを含むように設定した（図 11）．この分布から，実験 2 と同様に各クラス学習データ 2,000，テストデータ 200 をランダムに選択した．この条件において全数記憶方式の最近傍識別器を用いた場合の識別率は 97.3%であった．

図 12 は adjusted SOINN の学習結果である．正規分布の部分では，プロトタイプの分布にオーバラップが生じている．しかし，ASC の学習結果（図 13）では，ノイズ除去過程の効果によりオーバラップの部分が解消されている．ASC によって，平均 78.0 ± 28.0 個のプロトタイプが生成され，その結果 97.9 ± 0.3% の識別率を示した．したがってオーバラップを含むこ

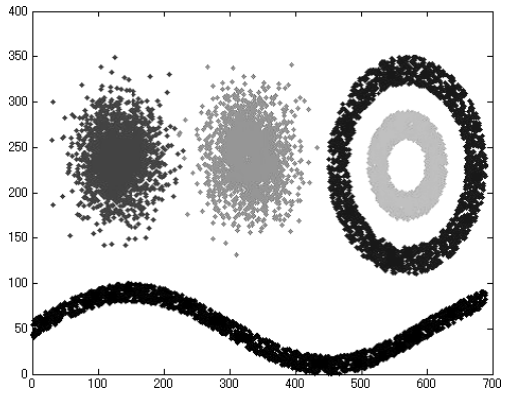


図 8　学習データ：オーバラップのない 5 クラスの分布（二つの正規分布，二つのリング，サインカーブ）
Fig. 8　Two Gaussian distributions, two concentric rings, and a sinusoid curve, without overlap (5 classes).

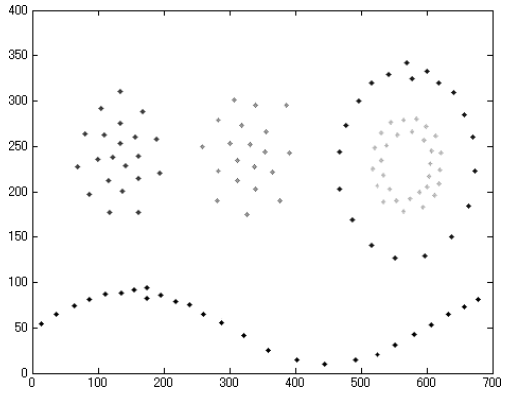


図 9　図 8 のデータに対する Adjusted SOINN の学習結果
Fig. 9　Adjusted SOINN results of Fig. 8.

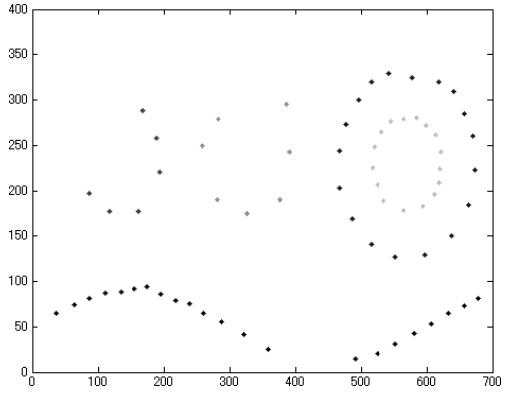


図 10　図 8 のデータに対する ASC の学習結果
Fig. 10　ASC results of Fig. 8.

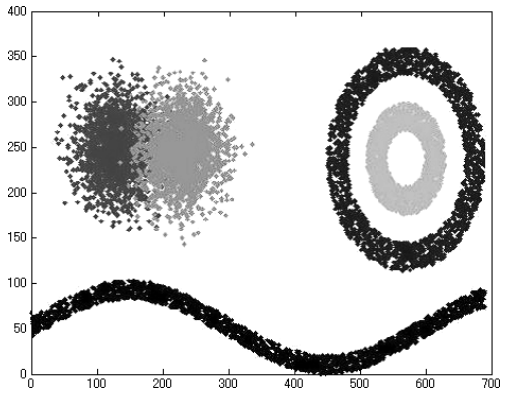


図 11　学習データ：オーバラップを含む 5 クラスの分布（二つの正規分布，二つの輪，サインカーブ）
Fig. 11　Two Gaussian distribution with overlap, two concentric rings, and sinusoid curve (5 classes).

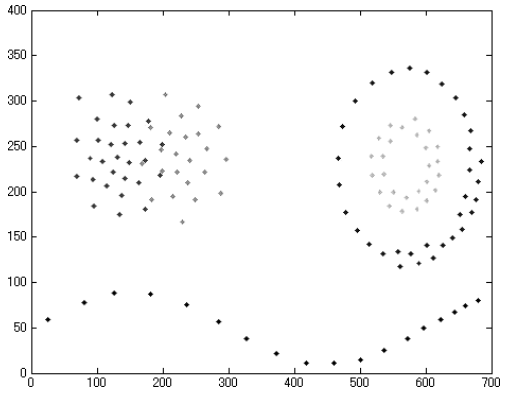


図 12　図 11 のデータに対する Adjusted SOINN の学習結果
Fig. 12　Adjusted SOINN results of Fig. 11.

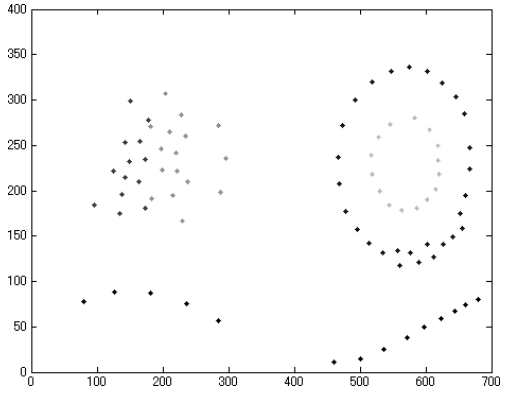


図 13　図 11 のデータに対する ASC の学習結果
Fig. 13　ASC results of Fig. 11.

の実験において，ASC は全数記憶方式の最近傍識別器を用いた場合に比べて，$0.78 \pm 0.28\%$ のプロトタイプ数 ($r_c = 0.76 \pm 0.28\%$) で，識別率で上回る結果を示した．

- 実験 4：10%のノイズデータ及びオーバラップを含む 5 クラスの分布

この実験では，実験 3 で用いた学習データの分布に更にノイズデータを追加した．本実験で用いたノイズデータは，特徴空間に一様分布し，各クラスの分布に対し 10% の確率で生起すると設定した．したがって，ノイズデータにもランダムにクラスラベルが付加されている．図 14 に本実験で用いた学習データの分布を示す．この図から，同一クラスタ内のサンプルであってもクラスラベルが異なる状況が多数存在することが分かる．この分布から，他の実験と同様に学習データ 2,000 サンプルを各クラスについてランダムに選択した．ただしテストデータについては，実験 3 の分布（図 11）から各クラス 200 サンプルをランダムに選択した．このようにデータを選択することで，学習データに含まれるノイズが識別率に与える影響を調べ，ASC のノイズ除去性能を評価することができる．このデータに対して全数記憶方式の最近傍識別器を用いた場合の識別率は 94.1% であった．

図 15 は adjusted SOINN の学習結果である．adjusted SOINN の学習においてもノイズを除去するプロセスは含まれているが，ノイズの影響のあるプロトタイプが完全に削除されていないことが分かる．一方，ASC の学習結果（図 16）では，ノイズ除去過程の効果からそれらのプロトタイプが削除されている．ASC で学習した結果，平均 82.9 ± 12.0 個のプロトタイプを生成し，$96.0 \pm 2.3\%$ の識別率を示した．したがってこの実験では，ASC は全数記憶方式の最近傍識別器と比べて，$0.83 \pm 0.12\%$ のプロトタイプ数 ($r_c = 0.83 \pm 0.12\%$) でより高い識別率を示した．また，10% のノイズを含む本実験の識別率が，ノイズを含まない実験 3 の識別率と同程度であることから，ASC がノイズに対して頑健に学習可能であると結論づけられる．

- k-means 法を用いた場合の学習結果との比較

本節の各実験において，adjusted SOINN または ASC で求められた各クラスのプロトタイプ数を利用して，k-means 法で識別器を構成することができる．したがって，ASC を k-means 法を用いてそれら二つの状況で構成した識別器と比較する．k-means 法のク

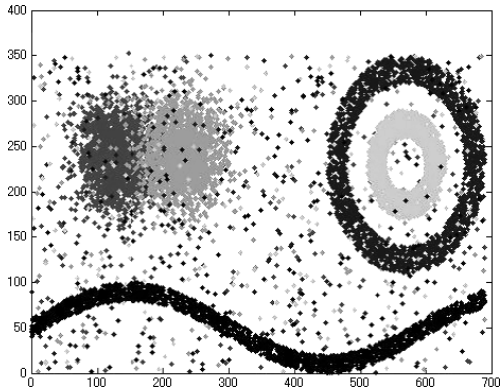


図 14　学習データ：10% のノイズデータ及びオーバラップを含む 5 クラスの分布（二つの正規分布，二つの輪，サインカーブ）
Fig. 14　Two Gaussian distribution with overlap, two concentric rings, and sinusoid curve; with 10% noise (5 classes).

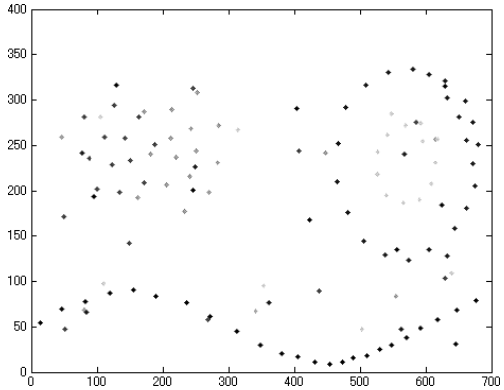


図 15　図 14 のデータに対する Adjusted SOINN の学習結果
Fig. 15　Adjusted SOINN results of Fig. 14.

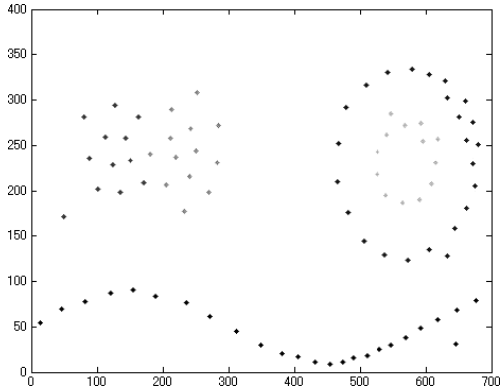


図 16　図 14 のデータに対する ASC の学習結果
Fig. 16　ASC results of Fig. 14.

表 1　各実験における ASC と k-means 法及び全数記憶方式の 1-最近傍法 (1-NN) の識別性能の比較：各実験結果の上段は 10 回の試行の識別率（%）の平均値と分散値，下段はプロトタイプ効率（%）を表す．
Table 1　Recognition performances of ASC classifier, k-means classifier, and one-nearest-neighbor classifier (1-NN).

	ASC	k-means (1)	k-means (2)	1-NN
実験 1	99.9 ± 0.1	99.8 ± 0.0	99.8 ± 0.1	100
	0.15 ± 0.00	1.33	0.15	100
実験 2	100.0 ± 0.0	98.9 ± 0.5	99.0 ± 0.4	100
	0.70 ± 0.00	1.24	0.70	100
実験 3	97.9 ± 0.3	95.5 ± 0.8	96.5 ± 0.5	97.3
	0.78 ± 0.28	1.28	0.78	100
実験 4	96.0 ± 2.3	87.0 ± 1.3	85.6 ± 2.0	94.1
	0.83 ± 0.12	1.38	0.83	100

ラスタ重心の初期値は，学習データからランダムに選択したサンプルの位置を用いた．10 回の試行により得られた識別率とプロトタイプ効率 r_c の平均値及び分散値を表 1 に示す．ただし，k-means (1) は Adjusted SOINN で求められたプロトタイプ数を用いた場合，k-means (2) は ASC で求められたプロトタイプ数を用いた場合を表す．

この結果から，2 クラス問題の実験 1 では，ASC と k-means 法の識別率に有意差は示されなかった．しかし，5 クラスの分布が存在する実験 2 では，ASC と k-means 法の識別率に 1% 程度の差が生じた．ASC では識別に不要なノードの削除過程（**2.4**）によって識別境界の決定に必要な位置にのみプロトタイプが生成される（図 10）のに対し，k-means 法では分布の全体にプロ トタイプが配置される．したがって，k-means 法では識別境界の決定に利用されないプロトタイプが存在することが，ASC と同数以上のプロトタイプ数であっても性能差が現れる要因と考えられる．

更に，オーバラップを含む分布の学習データを用いた実験 3 では，ASC と k-means 法の識別率に 1.4〜2.4%程度の差が生じた．実験 2 に比べて性能差が広がった要因は，図 13 に示したように，ASC ではノイズ除去過程によってオーバラップ領域にあるノードが削除され，オーバラップ領域が解消されることにあると考えられる．

ノイズを多く含んだ分布の学習データを用いた実験 4 では，ASC と k-means 法の識別率に 10% 程度の大きな差が生じた．この結果から，k-means 法はノイズの影響を受けやすいが，ASC は k-means 法の後に続くノイズ除去過程によってノイズに頑健に学習可能であるといえる．

表 2　各クラスの学習データ及びテストデータのサンプル数（Optdigits）
Table 2　Number of samples in the Optdigits database.

Class	No. of Training samples	No. of Test samples
0	376	178
1	389	182
2	380	177
3	389	183
4	387	181
5	376	182
6	377	181
7	387	179
8	380	174
9	382	180
Total	3823	1797

3.2　手書き文字データセットを用いた実験

本実験では，UCI repository のデータセットの一つである，Optical Recognition of Handwritten Digits database (optdigits) [18] を用いて ASC の学習効果を検証する．このデータセットは，‘0’ から ‘9’ の手書き文字を 42 人（学習データ 30 人，テストデータ 13 人）から収集したものである．各サンプルの次元数は 64 である．各クラスの学習データ及びテストデータのサンプル数を表 2 に示す．学習データの合計サンプル数は 3823，テストデータの合計サンプル数は 1797 である．全数記憶方式の最近傍識別器を用いた場合の識別率は 98% である．

ノイズ除去過程におけるパラメータ k を様々に変化させて予備実験を行った結果，$k = 0$ の場合，すなわちノイズ除去過程を除いた場合に最高の識別率を示した．**2.3** で指摘したように，データセットに含まれるノイズ量が少量であると予測される場合は，ノイズの影響を受けたノードは adjusted SOINN によって削除可能であると考える．したがって，本実験では，ASC からノイズ除去過程を除いて用いている．

ASC の学習に関して，パラメータ (a_d, λ) を 3 通りの異なるパラメータセット {（1）(50, 50)，（2）(25, 25)，（3）(10, 10)} に設定して，その学習結果の違いを検証した．各パラメータセットについて 10 回の学習を行い，各クラスのプロトタイプ数の平均値と分散値を算出した．その結果を表 3 に示す．更に，それらのプロトタイプを用いてテストデータを識別した結果を表 4 に示す．表 3 から，各クラスで別個に異なる

表 3 パラメータセット（a_d, λ）の違いによる各クラスのプロトタイプ数の変化（10 回の試行の平均値と分散値）
Table 3 The number of prototypes for different classes of Optdigits with different parameter sets (a_d, λ), displayed with the average obtained by 10-times training and standard deviation.

Class	No. of nodes for different sets of (a_d, λ)		
	(50, 50)	(25, 25)	(10, 10)
0	32 ± 4	20 ± 3	11 ± 3
1	40 ± 4	31 ± 4	11 ± 1
2	38 ± 3	27 ± 2	12 ± 5
3	41 ± 5	24 ± 3	12 ± 3
4	35 ± 3	25 ± 2	9 ± 3
5	39 ± 5	28 ± 5	10 ± 3
6	34 ± 3	25 ± 2	11 ± 3
7	33 ± 3	24 ± 2	12 ± 2
8	39 ± 5	25 ± 4	12 ± 3
9	45 ± 5	30 ± 3	12 ± 2
Total	377 ± 12	258 ± 7	112 ± 7

表 4 パラメータセット (a_d, λ) の違いによる ASC の識別率の変化（10 回の試行の平均値と分散値）
Table 4 Recognition performance of ASC classifier for Optdigits with different parameter sets (a_d, λ), displayed with an average of 10-times training and standard deviation.

	Parameter set of $\{a_d, \lambda\}$		
	(50, 50)	(25, 25)	(10, 10)
recognition ratio (%)	97.7 ± 0.2	97.4 ± 0.2	97.0 ± 0.2
No. of prototype	377 ± 12	258 ± 7	112 ± 7
Compression ratio (%)	9.9 ± 0.3	6.8 ± 0.2	2.9 ± 0.2

プロトタイプ数に決定されており，$\{ a_d, \lambda\}$ の値が大きい場合にプロトタイプ数が増加することが分かる．また，表 4 から，ASC は 10% 以下のメモリ使用量（$r_c = 9.9\%$）で 97.7% の識別率を示している（表 4 第 2 列）．また，更にメモリ使用量の節約を必要とする場合は，識別率は低下するが（表 4 第 2〜4 列），パラメータセット（a_d, λ）を変化させることでプロトタイプ数と識別率を調節することができる．

他手法を用いた場合の Optdigits の識別率について，よく知られた LibSVM（http://www.csie.ntu.edu.tw/Ẽcjlin/libsvm/）を用いて実験したところ，カーネル関数に RBF（Radius Basis Function）を採用した C-SVM において識別率 96.9% のときにサポートベクトルの数が 1197 であった．これに対し ASC は，表 4 第 4 列に示すように，同程度の識別率で約 1/10 のメモリ使用率（認識率 97.0%，プロトタイプ数 112）を達成している．また，柴田ら [6] は，最近傍識別器の高速化手法 K-D Decision Tree（KDDT）について，全数記憶方式の最近傍識別器と比較して同等の識別性能と約 1.19 倍の高速識別を示している．これに比べて ASC は，全数記憶方式の最近傍識別器の識別性能からは若干低いが，10 倍以上の高速化を実現した．

3.3 他手法との比較実験

Veenman らは Nearest Subclass Classifier（NSC(σ)）を提案し，他の識別器構成手法との比較を行っている [11]．比較されている手法は，k-Means Classifier（KMC(M)）[7]，k-Nearest Neighbors Classifier（NNC(k)）[1]，k-Edited Neighbors Classifier（ENC(k)）[16]，Multiscale Data Condensation algorithm（MDC(k)），Bootstrap technique（BTS(M)：パラメータ値 $T = 100, k = 3$ [9]），Learning Vector Quantization（LVQ(M)：パラメータ値 $\alpha = 0.3, \eta = 0.8, T = 100$ [9]）である．これらの手法のパラメータは cross-validation によって最適化されている [11]．更に彼らは，NNC，Reduction Technique 3（RT3(k)）[16]，タブーサーチ（TAB）のパラメータ値を固定した場合（NNC(3)，RT3(3)，及び TAB(0.05)：パラメータ値 $T_t = 0.03N$, $T_i = 20$, $T = 100$ [9]）とも比較している．この実験によって，彼らの提案手法である NSC が，パラメータのない condensing 手法である Minimal Consistent Set（MCS）[4] の性能を上回ることを示している．また彼らは，NSC が上述のすべての従来手法と比べて識別率とメモリ使用量のバランスにおいて優れていると結論づけている．

そこで本節では，提案手法である ASC と従来のプロトタイプに基づく識別手法とを比較して ASC を評価する．比較する手法は，NSC，KMC 及び LVQ とし，比較手法の結果は Veenman らが行った実験の結果（[11]，表 2）を用いる．本実験で比較していない手法の結果は，Veenman らの論文を参照されたい [11]．

[11] の実験では，9 種類のデータセットについて NSC と他手法を比較している．しかし，そのうち 2 種類のデータセットについては以下の理由，（1）筆者によって独自に作成された人工データセット（2）一般的に 60 次元のベクトルを用いる Sonar データセットが 30 次元で用いられている，のために本実験では採用しない．したがって本実験では，残り 7 種類のデータセットを用いて ASC を NSC やその他の手法と比較した．使用したデータセットを表 5 に示す．これら

表 6　ASC と他手法の学習結果の比較
Table 6　Comparison of results from ASC and other classifiers.

(a) Recognition ratio in percentages, displayed with average of 10-times ten-fold cross-validation and standard deviation. The bold-typeface recognition ratio is the best or near-best classifier.

Dataset	ASC (a_d, λ, k)	NSC (σ^2_{max})	KMC (M)	NNC (k)	LVQ (M)
Iris	**99.3** ± 1.86	96.3 ± 0.4	96.2 ± 0.8	96.7 ± 0.6	96.1 ± 0.6
Breast cancer	**99.2** ± 0.98	97.2 ± 0.2	95.9 ± 0.3	97.0 ± 0.2	96.3 ± 0.4
Ionosphere	**90.4** ± 0.64	**91.9** ± 0.8	87.4 ± 0.6	86.1 ± 0.7	86.4 ± 0.8
Glass	**73.5** ± 1.6	70.2 ± 1.5	68.8 ± 1.1	72.3 ± 1.2	68.3 ± 2.0
Liver disorders	62.6 ± 0.83	62.9 ± 2.3	59.3 ± 2.3	**67.3** ± 1.6	66.3 ± 1.9
Pima Indians	72.0 ± 0.63	68.6 ± 1.6	68.7 ± 0.9	**74.7** ± 0.7	73.5 ± 0.9
Wine	**82.6** ± 1.55	75.3 ± 1.7	71.9 ± 1.9	73.9 ± 1.9	72.3 ± 1.5
Average	**82.8** ± 1.16	80.4 ± 1.2	78.3 ± 1.1	81.1 ± 0.99	79.9 ± 1.2

(b) Compression ratios in percentages, the bold-typeface compression ratio shows the best or near best classifier. The optimal parameter value tuned by cross-validation is shown in ( ).

Dataset	ASC (a_d^*, λ^*, k^*)	NSC ($\sigma^{2\,*}_{max}$)	KMC (M^*)	NNC (k^*)	LVQ (M^*)
Iris	**4.0** (6, 6, 3)	7.3 (0.25)	8.0 (4)	100 (14)	15 (22)
Breast cancer	1.3 (8, 8, 3)	1.8 (35.0)	**0.29** (1)	100 (5)	5.9 (40)
Ionosphere	**3.4** (15, 15, 0)	31 (1.25)	4.0 (7)	100 (2)	6.8 (24)
Glass	**13.7** (15, 15, 0)	97 (0.005)	17 (6)	100 (1)	45 (97)
Liver disorders	**4.6** (6, 6, 0)	**4.9** (600)	11 (19)	100 (14)	8.4 (29)
Pima Indians	**0.6** (6, 6, 0)	1.7 (2600)	1.0 (4)	100 (17)	3.4 (26)
Wine	**3.2** (6, 6, 0)	96 (4.0)	29 (17)	100 (1)	32 (57)
Average	**4.4**	34.2	10.0	100	16.6

表 5　比較実験に用いたデータセット
Table 5　Datasets used for comparison.

Dataset	objects	features	classes
Iris	150	2	2
Breast cancer	683	9	2
Ionosphere	351	34	2
Glass	214	9	6
Liver disorders	345	6	2
Pima Indians	768	8	2
Wine	178	13	3

のデータセットの詳細については関連する文献 [18] を参照されたい．

ASC では，学習データから直接決定することができない三つのパラメータが存在する．そのため 10-fold cross-validation を行い，それらのパラメータの値を決定した．その結果得られた値は表 6 (b) に示している．

表 6 に ASC と他手法の比較実験の結果を示す．表 6 (a) は各データセットに対する識別率の 10 回の試行の平均及び分散を表している．Iris，Breast cancer，Glass 及び Wine データセットに対しては，ASC が全手法中の最高識別率を示している．また，Ionosphere と Pima Indians データセットに対しては最高識別率に近い値を示している．表 6 (a) の最終行にすべてのデータセットに対する識別率の平均値を示し，全手法の識別性能を比較した．その結果，ASC は比較した他のプロトタイプに基づく識別器構成手法をすべて上回る識別率を達成した．

更に，プロトタイプ効率 r_c を表 6 (b) に示す．すべてのデータセットにおいて，ASC は最高値若しくはそれに近い値を示し，全データセットの平均では最高値を達成した．NNC 及び NSC などの手法は高い識別性能を示すが，プロトタイプ効率の値が高いことから，識別に時間が掛かることが予想される．KMC と LVQ は高速な識別が可能であるが，識別性能は低い．それに比べて提案手法の ASC は高い識別性能を最小のプロトタイプ効率で達成している．特に，最新の手法である NSC [11] と比較した場合，識別性能で上回る結果を示す中で，約 7.4 倍の高速化を達成した．すなわち，ASC は比較したすべての他手法より，識別に重要なプロトタイプを効率良く選択可能であると考えられ，特にメモリ使用量の観点で優れているといえる．以上の実験結果から，ASC が NSC やその他のプロトタイプに基づく識別器構成手法に比べて有効な手法であると結論づけられる．

ASC のノイズ除去過程における事前定義パラメータ k について，各データセットに対する最適値は，2 種類のデータセットでは $k > 0$，5 種類のデータセットでは $k = 0$ であった．すなわちこの結果は，データセットによってノイズ除去過程の必要性は異なるが，

実データにおいてもこの過程が有効に機能する場合があることを示している．またそのような場合，a_d 及び λ に加えて，ノイズ除去過程のパラメータ k を調節することで，より個々の問題へのフィッティングが可能であるといえる．

4. むすび

本論文では，adjusted SOINN を基とした，プロトタイプに基づく識別器構成手法，Adjusted SOINN Classifier（ASC）を提案した．提案手法は，学習データの分布を近似するプロトタイプを学習により自動決定する．更に，ASC はノイズの影響を受けたプロトタイプを削除することができ，ノイズに頑健に高い識別率を示すことができる．加えて，識別に不要なプロトタイプを削除し，従来の識別器に比べて高速な識別を実現した．パラメータは cross-validation などでの調節が必要であるが，他手法に比べて調節は容易である．従来の様々な識別器と比較した実験では，ASC が最も高い識別性能を非常に高いプロトタイプ効率で達成し，有効な手法であることを示した．

最後に，提案手法に含まれるいくつかの課題を挙げる．提案手法には，三つのパラメータ，a_d,λ 及び k があり，それらは何らかの方法で事前に決定する必要がある．また，ASC は追加データに対して再学習が必要になる．これは，k-means 過程及び，識別に不要なプロトタイプの除去過程において，学習した全データを用いることに起因する．今後はこれらの問題に対処し，提案手法をオンライン学習可能な高速識別器に拡張する．

謝辞 本研究の実施にあたり新エネルギー・産業技術総合開発機構（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] T.M. Cover and P.E. Hart, “Nearest neighbor pattern classification,” IEEE Trans. Inf. Theory, vol.IT-13, no.1, pp.21–27, 1967.

[2] B.V. Dasarathy, Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques, IEEE CS Press, Los Alamitos, Calif., 1991.

[3] G.W. Gates, “The reduced nearest neighbor rule,” IEEE Trans. Inf. Theory, vol.IT-18, no.3, pp.431–433, 1972.

[4] B.V. Dasarathy, “Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design,” IEEE Trans. Syst. Man Cybern., vol.24, no.3, pp.511–517, 1994.

[5] 和田俊和，加藤丈和，“近接性グラフに基づく効率的 Condensing の理論，” 信学技報，PRMU，vol.103，no.96，pp.19–24, May 2003.

[6] 柴田智行，加藤丈和，和田俊和，“K-D Decision Tree とその応用－最近傍識別器の高速化と省メモリ化，” 信学論（D-II），vol.J88-D-II, no.8, pp.1367–1377, Aug. 2005.

[7] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, 2001.

[8] T. Kohonen, “Improved versions of learning vector quantization,” Proc. Int’l Joint Conf. Neural Networks, vol.1, pp.545–550, 1990.

[9] J.C. Bezdek and L.I. Kuncheva, “Nearest prototype classifier design: An experimental study,” Int’l J. Intelligent Systems, vol.16, pp.1445–1473, 2001.

[10] D.R. Wilson and T.R. Martinez, “Reduction techniques for instance-based learning algorithms,” Mach. Learn., vol.38, pp.257–286, 2000.

[11] C.J. Veenman and M.J.T. Reinders, “The nearest subclass classifier: A compromise between the nearest mean and nearest neighbor classifier,” IEEE Trans. Pattern Anal. Mach. Intell., vol.27, no.9, pp.1417–1429, 2005.

[12] F. Shen and O. Hasegawa, “An on-line learning mechanism for unsupervised classification and topology representation,” IEEE Computer Society International Conference on Computer Vision and Pattern Recognition (CVPR 2005), San Diego, CA, USA, June 2005.

[13] F. Shen and O. Hasegawa, “An incremental network for on-line unsupervised classification and topology learning,” Neural Netw., vol.19, pp.90–106, 2006.

[14] T.M. Martinetz, “Competitive hebbian learning rule forms perfectly topology preserving maps,” Proc. Int’l Conf. on Artificial Neural Networks, pp.427–434, 1993.

[15] T.M. Martinetz and K.J. Schulten, “Topology representing networks,” Neural Netw., vol.7, no.3, pp.507–522, 1994.

[16] D.L. Wilson, “Asymptotic properties of nearest neighbor rules using edited data,” IEEE Trans. Syst. Man Cybern., vol.2, no.3, pp.408–421, 1972.

[17] B. Fritzke, “A growing neural gas network learns topologies,” Neural Information Processing Systems 7, pp.625–632, MIT Press, Denver, USA, 1995.

[18] C. Merz and M. Murphy, UCI Repository of Machine Learning Databases, Irvine, CA, University of California Department of Information, 1996. http://www.ics.uci.edu/m̃learn/MLRepository.html

（平成 19 年 1 月 30 日受付，6 月 18 日再受付）

神谷　祐樹　（学生員）

2003 東工大・工・電気電子卒．2005 同大大学院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に在籍中．主として，パターン認識に関する研究に従事．

申　　富饒

1995 中国南京大・数学卒．1998 同大大学院数学専攻卒．2006 東工大大学院総合理工学研究科知能システム科学専攻博士課程了．博士（工学）．現在，中国南京大学計算機ソフトウェア新技術国家重点実験室准教授．

長谷川　修　（正員）

1993 東京大学大学院博士課程了．博士（工学）．同年電子技術総合研究所入所．1999 年 6 月より 1 年間カーネギーメロン大学ロボティクス研究所滞在研究員．2001 産業技術総合研究所主任研究員．2002 年 5 月東京工業大学大学院理工学研究科付属像情報工学研究施設准教授．2002 から 3 年間科技団さきがけ研究 21 研究員．パターン認識，ニューラルネット，実世界知能システムなどの研究に従事．人工知能学会，日本認知科学会，日本顔学会，IEEE CS 等各会員．