情報処理学会論文誌　数理モデル化と応用　Vol.5　No.2　29–38 (June 2012)

ユーザフィードバックを用いた重み付き自己組織化マップ

久田 大地[1]　吉川 毅[1]　野中 秀俊[1,a)]

受付日 2011年11月4日，再受付日 2011年12月21日，
採録日 2012年1月19日

概要：ユーザにとって望ましいクラスタリング結果の定義は，ユーザそれぞれによって異なるものである．そのため，入力データの特徴のみを用いてクラスタリングを行っても，望ましいクラスタリング結果が得られない場合が多くある．そこで，本研究では教師なしクラスタリング手法である自己組織化マップに重みベクトルとユーザフィードバックを導入し，クラスタリングにユーザの意図を反映させる手法を提案する．

キーワード：自己組織化マップ，ユーザフィードバック，クラスタリング，重みベクトル

Weighted Self-organizing Map with User Feedback

Daichi Hisada[1]　Takeshi Yoshikawa[1]　Hidetoshi Nonaka[1,a)]

Received: November 4, 2011, Revised: December 21, 2011,
Accepted: January 19, 2012

Abstract: A clustering result which a user wants to obtain differs from user to user. A desirable clustering result will not be obtained only with the data features in clustering. We propose a new clustering algorithm which can reflect user's intention by introducing weight vectors and user feedback into self-organizing maps. By using this algorithm the user can refine and adjust the clustering system interactively.

Keywords: self-organizing maps, user feedback, clustering, weight vector

1. はじめに

1.1 背景

互いに特徴が似ているデータの集まりをクラスタとよび，データを適切なクラスタに分けることをクラスタリングという．クラスタリングを行うシステムは，あらかじめ与えられた何らかの基準に基づいてクラスタリングを行い，この基準の下で最善のクラスタリング結果を出力する [1]．そのため，どのような基準に基づいてクラスタリングするかによって，様々なクラスタリングを考案できることが指摘されている [2]．

たとえば，ある図形を入力データとしてクラスタリングすることを考える（**図 1**）．図形の辺の数と頂点数を基準としてクラスタリングを行うとクラスタリング A のようになり，図形の色を基準としてクラスタリングを行うとクラスタリング B のようになる．さらに，特徴としては説明ができないが実験者の直感に基づいたクラスタリングとして，クラスタリング C のようなクラスタリングも考えられる．

上に述べた例は，基準の選び方によって，クラスタリング結果に差異が生じることを示している．この他にも入力データ間の距離定義を変更したり，クラスタリングの手順を変えたりすることにより，様々なクラスタリング結果が得られる可能性がある．

入力データに対して，正しいクラスタリングが一意に定まらない原因として，クラスタの厳密な定義が存在しないことが指摘されている [1], [2]．そのためクラスタリング手法の評価実験などでは，正しいクラスタリング結果を実験者があらかじめ作成したり，あるいはすでにクラスに分類されているデータを使用し，実験で得られたクラスタリング結果と比較することによって，手法の性能を評価することが一般的となっている．

[1] 北海道大学大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Hokkaido 060–0814, Japan
[a)] nonaka@main.ist.hokudai.ac.jp

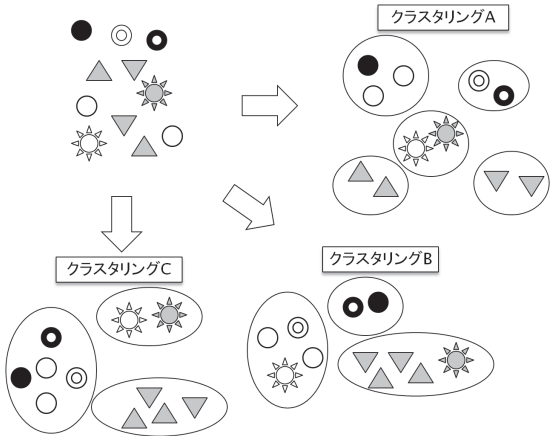

図 1　クラスタリング結果の例
Fig. 1　Examples of clustering result.

このように，正しいクラスタリング結果やその定義がユーザの主観に依存するという問題点に対し，筆者らは，システムがユーザからクラスタリングに対する何かしらの情報を受け取り，ユーザの意図をクラスタリングに反映させることにより，望ましいクラスタリング結果が得られる可能性があることに着目した．

本研究では，自己組織化マップに Subspace Clustering の手法を導入し，それぞれのクラスタに対する特徴の関連度を重みベクトルでとらえた Weighted SOM を提案する．さらに，Weighted SOM が出力したクラスタリング結果に対してユーザフィードバックを与えることにより，特徴の関連度を変化させ，実験者の意図を考慮したクラスタリングを行う手法を提案する．

1.2　提案概要

本研究ではユーザフィードバックを用いた重み付き自己組織化マップである Weighted SOM を提案する．Weighted SOM では，各クラスタに重みベクトルを付加することにより，それぞれのクラスタに対して支配的な特徴や各特徴の関連度を定量的に得ることが可能となる．また，Weighted SOM が出力したクラスタリング結果に対して，ユーザフィードバックを与え，重みベクトルに反映させることにより，ユーザの意図を考慮したクラスタリングを行うことが可能となる．

本研究では Weighted SOM によりクラスタリングされた結果に対し，ある入力データ$\boldsymbol{x_f}$が配置される望ましいクラスタ k'_f を，他の入力データのクラスタリングを考慮してユーザが指定することをユーザフィードバックとする．

このユーザフィードバックによりシステムは，クラスタ k'_f の周囲にクラスタリングされている入力データのうち，$\boldsymbol{x_f}$と似ている入力データを k'_f に集めることによって，入力データ$\boldsymbol{x_f}$がクラスタ k'_f に配置されることが望ましいというユーザの意図をクラスタリングに反映させる．

自己組織化マップ（SOM: Self-Organizing Maps）は，Kohonen が提案したニューラルネットワークを用いた教師なしクラスタリング手法である [3]．SOM をクラスタリングに用いると，高次元の入力データを 2 次元のマップに射影することができる．本研究で SOM を用いる理由は，クラスタリング結果を 2 次元のマップ上で表現でき，それぞれのクラスタがどのような関係を示しているかを視覚的に観察できるからである．また SOM では，2 次元のマップ上で入力データ間の距離関係が保存されているため，ユーザは他のクラスタを参考に望ましい入力データのクラスタを予想することができる．さらに，SOM は教師なしクラスタリング手法であるため，教師データの一種であるユーザフィードバックがない場合でも，クラスタリングを行うことができる．

2.　関連研究

2.1　ユーザフィードバックを用いるクラスタリング手法

Cohn ら [4] は，ユーザフィードバックを用いたクラスタリング手法を提案した．この研究では，大量のデータを人間の直感に基づいてグループ分けする作業を Yahoo! Problem とよび，Yahoo! Problem の解決方法として，クラスタリング手法にユーザの意図を反映させる手法を提案している．彼らは，ある特定の 2 つの入力データが同じクラスタに属さないというユーザフィードバックをクラスタリングに導入している．しかし，彼らが指摘しているように，特定の入力データ間に制約情報を与える方法以外にも様々なユーザフィードバックが考えられる．

Nurnberger ら [5] は，教師なしクラスタリング手法である SOM に，ユーザフィードバックを学習する重みベクトルを適用した手法を提案した．この研究では，ある参照ノードに属すると判断された入力データを，ユーザが別の参照ノードに移動させるというユーザフィードバックを SOM に導入している．このユーザフィードバックを用いて重みベクトルを更新し，入力データをクラスタリングしなおすことによりユーザの意図に考慮したクラスタリングを行っている．この手法は本研究の提案手法とは異なり，ユーザフィードバックを受けて初めて重みベクトルが更新される．

山田ら [6] は，人間と知的システムが協調しながら問題解決を行う知的インタラクティブシステム実現のために，最小ユーザフィードバック（MUF: Minimal User Feedback）を提唱している．この研究では，知的システム全体のパフォーマンスを維持しつつ，ユーザが与えるフィードバックの計算論的コストと認知的コストを最小にすることを目的としている．計算論的コストを最小化するために制約つきクラスタリングを拡張した手法 [7] を，認知的コストを最小化するために人の能動的学習を促進する GUI [8] を提案している．この両者では，主に制約付き k-means 法を改

良することを考えている．

2.2　重みベクトルを用いるクラスタリング手法

Doeniconi ら [9] は，クラスタリングに重みベクトルを適用した Locally Adaptive Clustering（LAC）を提案している．クラスタリングに重みベクトルを適用することにより，各クラスタにおける各次元の関連度をとらえることができ，クラスタリング精度が向上している．しかし，LAC を用いるには重みの影響をどの程度重視するのかを表すパラメータ h を適切に決定する必要があり，適切なパラメータの調整が問題となる．また，この研究では何らかの制約条件やユーザフィードバックを用いて，重みを更新する方法については述べられていない．

Cheng ら [10] は，k-means 法に重みベクトルを適用した Locally Weighted Clustering（LWC）と，LWC に must-link と cannot-link といった制約情報を付加した，Constrained Locally Weighted Clustering（CLWC）を提案している．LWC と CLWC は LAC とは異なり，重みの影響度を表すパラメータが存在しないためパラメータの調整が必要ない．LWC と CLWC はいくつかのデータに対して LAC などの既存クラスタリング手法よりも高いクラスタリング精度を示している．しかし，LWC と CLWC はどちらも k-means 法を拡張した手法であるため，正しいクラスタ数が分からない場合は，正しい分類ができない．

3.　背景知識

3.1　Subspace Clustering

一般的なクラスタリング手法は入力データのすべての次元を用いてクラスタリングを行っている．しかし，高次元の入力データにはクラスタリングに不必要な次元が多く，そのすべてを用いてクラスタリングを行うとクラスタリング精度が落ちる場合がある．また，入力データが高次元になると入力データ間の距離が等距離になってしまい，データ間の類似性を正しく求めることができなくなってしまうという指摘がある [11]．そこで Subspace Clustering では，各クラスタに重みベクトルを付加して，クラスタごとに次元の重みを変化させ，クラスタに関連性のある次元を用いてクラスタリングを行うことでクラスタリング精度の向上を図っている．

3.1.1　Locally Weighted Clustering

提案手法では，Cheng ら [10] が提案している k-means 法に重みベクトルを加えた手法を SOM に拡張することを考える．そこで，ここでは Cheng らが提案している Locally Weighted Clustering（LWC）について説明する．

3.1.1.1　概要

LWC（**図 2**）ではまず，各クラスタの中心点 $\boldsymbol{c_k} = (c_{k1}, c_{k2}, \ldots, c_{kM})$ と各クラスタに付加されている重みベクトル $\boldsymbol{w_k} = (w_{k1}, w_{k2}, \ldots, w_{kM})$ を初期化する．次に，

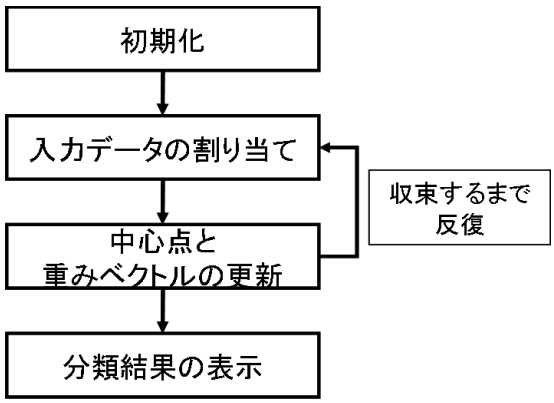

図 2　LWC の流れ図
Fig. 2　Flow diagram of locally weighted clustering.

各入力ベクトル $\boldsymbol{x_i} = (x_{i1}, x_{i2}, \ldots, x_{iM})$ を重み付き距離関数 $L_{2,\boldsymbol{w_k}}$ を用いて各クラスタに割り当てる．その後，各クラスタの中心点$\boldsymbol{c_k}$と重みベクトル$\boldsymbol{w_k}$を更新する．入力データの各クラスタへの割当てと，中心点$\boldsymbol{c_k}$と重みベクトル$\boldsymbol{w_k}$の更新をクラスタリング結果が収束するまで繰り返し行う．ここで M は，入力データの次元数である．

3.1.1.2　初期化

k-means 法において中心点$\boldsymbol{c_k}$の初期値はクラスタリング精度に影響を及ぼすため，中心点$\boldsymbol{c_k}$を入力データを用いて初期化を行う Forgy initialization などの手法を用いて初期化する．また，重みベクトル$\boldsymbol{w_k}$は式 (1) の制約条件を設けるため，重みベクトル$\boldsymbol{w_k}$のすべての要素を 1 で初期化する．ここで M は入力ベクトルと重みベクトルの次元数である．

$$\prod_{j=1}^{M} w_{kj} = 1 \tag{1}$$

3.1.1.3　入力データの割り当て

重み付き距離関数 $L_{2,\boldsymbol{w_k}}$ を用いて，入力データを各クラスタに割り当てていく（式 (2)）．ここで K は割当てを行う全クラスタの総数である．

$$\phi_c(\boldsymbol{x_i}) = \arg \min_{1<k<K} L_{2,\boldsymbol{w_k}}(\boldsymbol{x_i}, \boldsymbol{c_k}) \tag{2}$$

$$L_{2,\boldsymbol{w_k}}(\boldsymbol{x_i}, \boldsymbol{c_k}) = \sqrt{\sum_{j=1}^{M} w_{kj} |c_{kj} - x_{ij}|^2} \tag{3}$$

3.1.1.4　中心点と重みベクトルの更新

中心点$\boldsymbol{c_k}$は各クラスタ C_k に割り当てられた入力ベクトルの平均値として更新する（式 (4)）．中心点の更新後，式 (5) を用いて重みベクトル$\boldsymbol{w_k}$を更新する．ここで，式 (6) 中の $\sum_{\boldsymbol{x_i} \in C_k} |x_{ij} - c_{kj}|^2$ の値が，非常に小さくなると重みベクトルの更新時に悪影響を及ぼす可能性がある．そこで $\sum_{\boldsymbol{x_i} \in C_k} |x_{ij} - c_{kj}|^2$ の値が閾値 $O_{threshold}$ より小さい場合は，$\sum_{\boldsymbol{x_i} \in C_k} |x_{ij} - c_{kj}|^2 = O_{threshold}$ とする．逆に $\sum_{\boldsymbol{x_i} \in C_k} |x_{ij} - c_{kj}|^2$ の値が非常に大きくなる場合もあり，この場合は計算途中で桁あふれが発生する可能性がある．

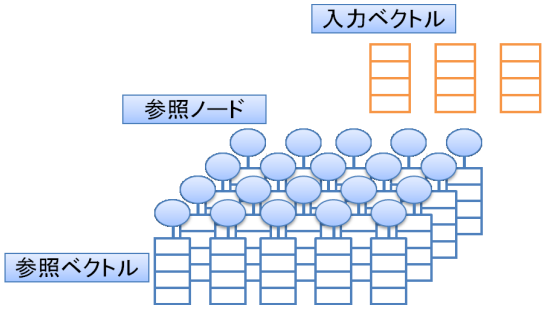

図 3　SOM の概念図

Fig. 3　Conceptual diagram of self-organizing map.

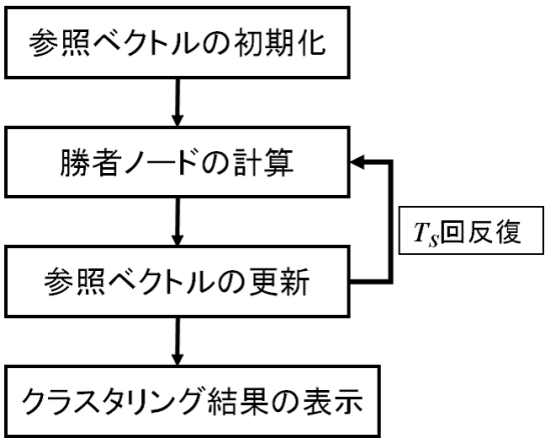

図 4　SOM の流れ図

Fig. 4　Flow diagram of self-organizing map.

そこで実際には式 (5) の自然対数をとった値を計算し，得られた値を用いて重みベクトルを計算している．

$$c_{kj} = \frac{1}{|C_k|} \sum_{\boldsymbol{x_i} \in C_k} x_{ij} \tag{4}$$

$$w_{kj} = \frac{\lambda_k}{\sum_{\boldsymbol{x_i} \in C_k} |x_{ij} - c_{kj}|^2} \tag{5}$$

$$\lambda_k = \left\{ \prod_{j=1}^{M} \left(\sum_{\boldsymbol{x_i} \in C_k} |x_{ij} - c_{kj}|^2 \right) \right\}^{\frac{1}{M}} \tag{6}$$

3.2　自己組織化マップ

SOM は Kohonen により提案された教師なしクラスタリング手法である．SOM は多次元の入力データを 2 次元平面に配置された参照ノードに分類する（**図 3**）．このとき，ある入力データに対して特徴が似ている他の入力データは 2 次元平面上で近くの参照ノードに分類され，特徴が似ていない入力データは遠くの参照ノードに分類される．

SOM には Online 型 SOM と Batch 型 SOM がある．提案手法では，学習効率が良いとされている Batch 型 SOM を用いる．Batch 型 SOM は 4 段階のステップで構成されている（**図 4**）．

3.2.1　参照ベクトルの初期化

2 次元平面に配置された参照ノード k はそれぞれ，参照ベクトル $\boldsymbol{m_k}$ とよばれる M 次元のベクトルを持っている．M は入力データの次元数と同じである．ここではまず参照ノードの学習を行う前に，それぞれの参照ベクトルの初期化を行う．一般的に参照ベクトルは乱数を用いて初期化される．

3.2.2　勝者ノードの決定

次に，各入力データに対して最も距離が小さい参照ノードを計算する．ある入力データ $\boldsymbol{x_i}$ に対して最も距離が小さい参照ノードを入力データ $\boldsymbol{x_i}$ の勝者ノード k_i^* とよぶ．勝者ノードは式 (7) を用いて計算される．

$$k_i^* = \arg\min_k D(\boldsymbol{x_i}, \boldsymbol{m_k}) \tag{7}$$

$$D(\boldsymbol{x_i}, \boldsymbol{m_k}) = \sqrt{\sum_{j=1}^{M} (m_{kj} - x_{ij})^2} \tag{8}$$

3.2.3　参照ベクトルの更新

前項で求めた勝者ノードと入力ベクトルを用いて参照ベクトルの更新を行う（式 (9)）．ここで，$N_{k,r(t)}$ は参照ノード k の近傍半径 $r(t)$ 内に存在する参照ノードを勝者ノードとした入力データの集合である（式 (10)）．$Dist(k_1, k_2)$ は，参照ノード k_1 と参照ノード k_2 の 2 次元マップ上でのユークリッド距離であり，$n(N_{k,r(t)})$ は $N_{k,r(t)}$ の要素数である．

$$\boldsymbol{m_k} = \frac{1}{n(N_{k,r(t)})} \sum_{\boldsymbol{x_i} \in N_{k,r(t)}} \boldsymbol{x_i} \tag{9}$$

$$N_{k,r(t)} = \{\boldsymbol{x_i} \mid Dist(k_i^*, k) < r(t)\} \tag{10}$$

3.2.4　更新反復

勝者ノードの決定と参照ベクトルの更新を T_s 回反復する．反復する際に，近傍半径 $r(t)$ を反復回数 t にあわせて減衰させることにより，参照ノードの学習効率が上昇する．ここでは式 (11) を用いて近傍半径 $r(t)$ を減衰させる．T_s は参照ベクトル更新回数であり，t は現在の更新回数である．

$$r(t) = \left(1 - \frac{t}{T_s}\right) r(0) \tag{11}$$

4.　提案手法

4.1　概要

提案手法である Weighted SOM では，Kohonen が提案した Batch 型 SOM [3] の各参照ノードに重みベクトル $\boldsymbol{w_k}$ を適用し，参照ベクトル $\boldsymbol{m_k}$ と入力データ $\boldsymbol{x_i}$ を用いて重みベクトル $\boldsymbol{w_k}$ を更新する．SOM に重みベクトルを導入することにより，参照ベクトルごとに各次元の重みを変化させクラスタリング能力の向上を図る（**図 5**）．本研究では，重みベクトル $\boldsymbol{w_k}$ の更新には LWC で用いられている重みベクトルの更新式を拡張した式を採用した．さらに，Weighted SOM から得られたクラスタリング結果にユーザフィードバックを与えることにより，重みベクトル $\boldsymbol{w_k}$ の値を変化させ，ユーザの意図に合ったクラスタリングを行う（**図 6**）．

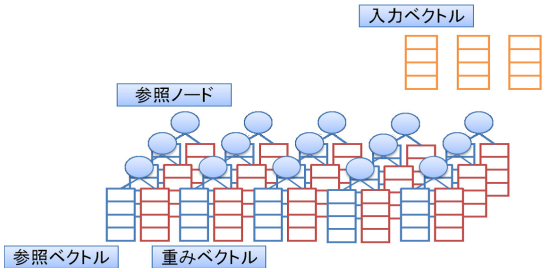

図 5　Weighted SOM の概念図
Fig. 5　Conceptual diagram of weighted self-organizing map.

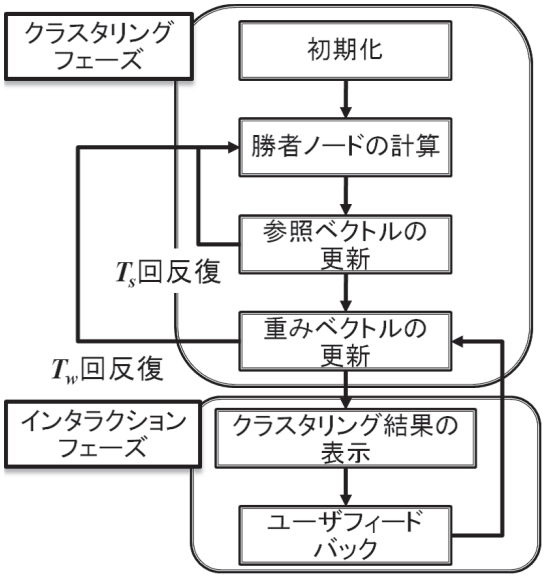

図 6　提案手法の流れ図
Fig. 6　Flow diagram of proposed clustering algorithm.

4.2　勝者ノードの計算

提案手法では，LWC [10] で用いられている重み付きの距離関数 $L_{2,\boldsymbol{w_k}}$ を拡張した重み付き距離 $D_{\boldsymbol{w_k}}$ を使用し，入力データ $\boldsymbol{x_i}$ に対して最も距離が小さい参照ノードを勝者ノード k_i^* とする（式 (12)）．勝者ノード k_i^* は入力データ $\boldsymbol{x_i}$ と最も距離が小さい参照ノードのことであり，入力ベクトル $\boldsymbol{x_i}$ は参照ノード k_i^* に属すると判断される．

$$k_i^* = \arg\min_k D_w(\boldsymbol{x_i}, \boldsymbol{m_k}) \tag{12}$$

$$D_w(\boldsymbol{x_i}, \boldsymbol{m_k}) = \sqrt{\sum_{j=1}^{M} w_{kj} |m_{kj} - x_{ij}|^2} \tag{13}$$

4.3　参照ベクトルの更新

提案手法では，式 (9) に従って T_s 回参照ベクトルを更新する．

4.4　重みベクトルの更新

参照ベクトルを T_s 回更新した後，各参照ノードに分類された入力データと参照ベクトル $\boldsymbol{m_k}$ を用いて重みベクトル $\boldsymbol{w_k}$ の更新を行う．参照ノード k を勝者ノードとする入力データの集合を $N_{k,0}$ とする．提案手法では LWC で用いられていた重みベクトルの更新式におけるクラスタの中心点 $\boldsymbol{c_k}$ を，参照ベクトル $\boldsymbol{m_k}$ に置き換えた式を重みベクトルの更新に使用する（式 (14)）．重みベクトルを更新し，再び勝者ノードの計算と参照ベクトルの更新を T_s 回行う．重みベクトルが T_w 回更新された後に，インタラクションフェーズに移行する．ここで M は入力データの次元数，N は入力データの数を表す．

$$w_{kj} = \frac{\lambda_k}{\sum_{\boldsymbol{x_i} \in N_{k,0}} |x_{ij} - m_{kj}|^2} \tag{14}$$

$$\lambda_k = \left\{ \prod_{j=1}^{M} \sum_{\boldsymbol{x_i} \in N_{k,0}} |x_{ij} - m_{kj}|^2 \right\}^{\frac{1}{M}} \tag{15}$$

4.5　クラスタリングと結果の表示

各入力データはそれぞれの勝者ノードに割り当てられることにより，クラスタリングされる．提案手法では分類されたデータ間の関係が分かりやすいように，2 次元平面で碁盤の目状に各参照ノードと分類された入力データをディスプレイに描画する．

4.6　ユーザフィードバック

クラスタリングフェーズによりクラスタリングされた結果を見たユーザは，自分の意図と違った参照ノードに配置されている入力データ $\boldsymbol{x_f}$ に対して，適切だと思われる参照ノード k_f' を指定する．いくつかの入力データに対して参照ノードの指定を受けた後，重みベクトルを更新する．以降のクラスタリングにおいて，指定を受けた入力データ $\boldsymbol{x_f}$ の勝者ノード k_f^* を k_f' に変更する．入力データ $\boldsymbol{x_f}$ は k_f' に分類されているものとして重みベクトルの更新と参照ベクトルの更新を行う．ある入力データに対して，直接最適な配置を指定することにより，cannot-link や must-link といった制約条件よりも，ユーザにとって直感的なフィードバックを与えることができる．

5.　実験

5.1　実験 1

重みベクトルを付加しない Batch 型 SOM と提案手法のクラスタリング結果を比べ，重みベクトルを SOM に付加することにより，どのようなクラスタリングがなされるかを調べる．また，ユーザフィードバックを受けたことにより重みベクトルと参照ベクトルに変化が起こり，クラスタリング結果にどのような影響が起きるのか調べる．

5.1.1　実験方法

提案手法では入力データの各次元の関連度を重みベクトルで表現する．そのため，次元の大きい入力データにおいて重みの影響が鮮明になると考えられる．そこで，提案手法のクラスタリング結果を調べるために，筆者が先行研究 [12] で行ったソフトウェアのクラスタリングを行う．ソ

フトウェアのクラスタリングでは，単語の出現頻度を元に入力データを作成するため，次元が大きい入力データを扱う．また，ソフトウェアのクラスタリングにおける正しいクラスタリング結果はユーザの好みに大きく影響されるため，ユーザフィードバックがより重要になる．

クラスタリングに用いる入力データは，以下の方法で作成した．

(1) それぞれのソフトウェアに対して，ソフトウェア名を検索クエリとして Yahoo! Search を用いて web 検索を行う．
(2) 上位 50 件分の検索結果スニペットを形態素解析し，名詞のみ集める．
(3) それぞれのソフトウェアに対して名詞の出現回数をベクトルで表現し，ソフトウェアベクトルとする．
(4) すべてのソフトウェアに対して求めたソフトウェアベクトルを tf-idf を用いて変換する．

単語の出現頻度を tf-idf [13] を用いて変換することにより，すべての文書に共通して出現する単語や，特定の文書にのみ出現する単語の影響を抑えることができる．クラスタリングに用いたソフトウェアは**表 1** のとおりである．今回の実験では上記の手順で作成した 7,666 次元のソフトウェアベクトル 29 個に対してクラスタリングを行う．SOM の参照ノードは 6×6 の碁盤の目状に配置し，近傍半径 $r(0)$ を 2 としている．また，参照ベクトルの更新回数 T_s を 100，重みベクトルの更新回数 T_w を 10 としている．

5.1.2　実験結果

図 7 は，Batch 型 SOM を用いてソフトウェアのクラスタリングを行った結果である．この結果から，領域 A に音楽・動画プレイヤ，領域 B に Microsoft Visual Studio，領域 C に Microsoft Office，領域 D にネットワーク関係のソフトがそれぞれ分類されていることが分かる．

次に**図 8** は，提案手法である Weighted SOM を用いてソフトウェアのクラスタリングを行った結果である．この結果からは，領域 A に統合開発環境，領域 B に Microsoft Office，領域 C にチャットソフト，領域 D に音楽・動画プレイヤ，領域 E にインターネットブラウザが分類されていることが分かる．また，クラスタリング後の Weighted SOM の参照ベクトルと重みベクトルの値を調べると，参照ベクトルの値が 0 の次元の重みが大きくなっていることが分かった．

さらに，図 8 のクラスタリング結果を出力した Weighted SOM にユーザフィードバックを与えた結果が**図 9** である．ここでは，ユーザフィードバックとして EmEditor の分類を変更している．その結果，フィードバックを与えられた参照ノード付近に存在していた Microsoft Office 関係のソフトウェアが EmEditor と同じ参照ノードに移動し，移動した EmEditor の周辺参照ノードに存在した WinShell がプログラミングのクラスタに移動している．また，Weighted SOM にユーザフィードバックを与える前後の参照ベクトルと重みベクトルの値を比較すると，両者の値が変化していることが確認できた．

表 1　分類対象のソフトウェア一覧
Table 1　List of software to be clustered.

	ソフトウェア名
1	BEAT!MusicPlayer
2	Google Chrome
3	Internet Explorer
4	Outlook Express
5	Windows Media Player
6	Rip!AudiCO FREE Ver 4.03
7	Adobe Reader 9
8	EmEditor
9	GOM Player
10	Lunascape6
11	Microsoft Office Excel 2007
12	Microsoft Office Outlook 2007
13	Microsoft Office Picture Manager
14	Microsoft Office PowerPoint 2003
15	Microsoft Office Word 2007
16	Microsoft Visual Basic 2008 Express Edition
17	Microsoft Visual C++ 2008 Express Edition
18	Visual Studio 2008 コマンドプロンプト
19	Mozilla Firefox
20	QuickTime Player
21	Skype
22	VLC media player
23	Windows Live Call
24	Windows Live Messenger
25	Windows Live メール
26	Windows Messenger
27	WinShell
28	ペイント
29	ワードパッド

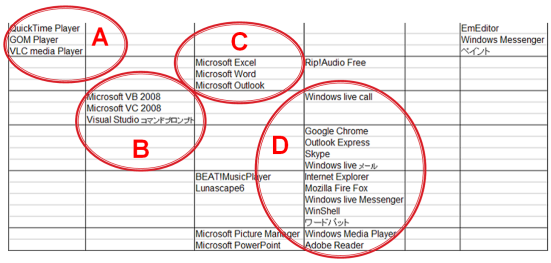

図 7　Batch 型 SOM でのクラスタリング結果
Fig. 7　Clustering result of batch version of SOM.

5.2　実験 2

重みベクトルを付加しない Batch 型 SOM と提案手法のクラスタリング結果を比較し，各手法によるクラスタリング結果が，どの程度ユーザの意図に近いか被験者実験を行い調査する．

また，提案手法と既存手法のクラスタリング結果にユー

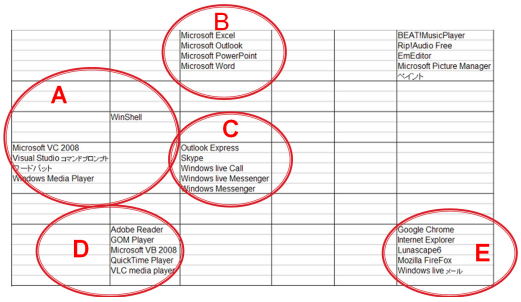

図 8　Weighted SOM でのクラスタリング結果
Fig. 8　Clustering result of Weighted SOM.

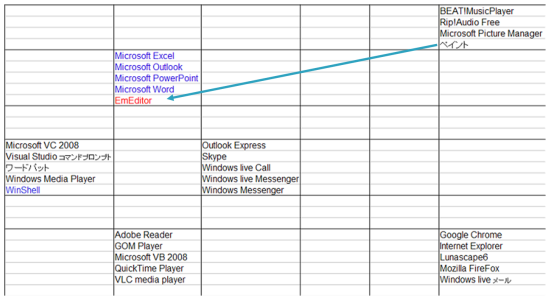

図 9　Weighted SOM にフィードバックを与えた場合のクラスタリング結果
Fig. 9　Clustering result of Weighted SOM with user feedback.

ザフィードバックを与えることにより，ユーザにとって妥当なクラスタリング結果の変更がなされるか調査する．

5.2.1　実験方法

被験者に入力データを複数の手法でクラスタリングした結果をランダムな順で提示し，それぞれのクラスタリング結果がどの程度ユーザの意図に近いか 5 段階で評価してもらう．その後，それぞれのクラスタリング結果に対し，最大 5 個のユーザフィードバックを与えてもらい，再度クラスタリングした結果を被験者に提示する．各手法にユーザフィードバックを与えた後のクラスタリング結果が，どの程度ユーザの意図に近いか，また，クラスタの移動がユーザフィードバックに対して妥当であるかを 5 段階で評価してもらう．

提案手法と比較する対象のクラスタリング手法は以下の 2 つである．

SOM1　式 (16) を用いて参照ベクトルを更新する従来の Batch 型 SOM に Nurnberger ら [5] の手法を導入したもの

SOM2　式 (9) を用いて参照ベクトルを更新する従来の Batch 型 SOM にユーザフィードバックを導入したもの

SOM2 の手法では，提案手法と同様に，ユーザフィードバックとして特定の入力データに対して適切な参照ノードを指定している．

質問1：3つの手法によるクラスタリング結果に対して，
自分が意図したクラスタリング結果にどの程度近いか．（1〜5：近い〜遠い）

		被験者					
		No1	No2	No3	No4	No5	評価値平均
評価値	SOM1	5	5	4	5	4	4.6
	SOM2	2	2	5	2	1	2.4
	提案手法	1	2	5	1	1	2

質問2：それぞれの手法にユーザフィードバックを与えた結果が，
自分が意図したクラスタリングにどの程度近いか．（1〜5：近い〜遠い）

		被験者					
		No1	No2	No3	No4	No5	評価値平均
評価値	SOM1	4	4	4	2	3	3.4
	SOM2	1	2	3	2	1	1.8
	提案手法	1	2	4	3	1	2.2

質問3：それぞれの手法にユーザフィードバックを与え，
妥当なクラスタの移動がなされたか．（1〜5：妥当〜妥当でない）

		被験者					
		No1	No2	No3	No4	No5	評価値平均
評価値	SOM1	4	4	2	2	4	3.2
	SOM2	1	2	2	2	1	1.6
	提案手法	2	2	2	4	1	2.2

図 10　被験者によるクラスタリング結果の評価
Fig. 10　User's evaluation of clustering result.

$$\boldsymbol{m_k} = \frac{\sum_{i=1}^{N} h_{k_i^*,k}\boldsymbol{x_i}}{\sum_{i=1}^{N} h_{k_i^*,k}} \tag{16}$$

$$h_{k_i^*,k} = \alpha(t)\exp\left(-\frac{Dist^2(k_i^*,k)}{2r^2(t)}\right) \tag{17}$$

実験 2 において使用する入力データは，実験 1 と同様のソフトウェアベクトルである．本実験では実験条件を統一するために，表 1 から使用用途が分かりにくいソフトウェアを除いた 24 個のソフトウェアに対してソフトウェアベクトルを作成し，クラスタリングを行う．SOM の設定は実験 1 と同じである．また，式 (17) で用いる学習率 $\alpha(0)$ は 0.1 とし，参照ベクトルの更新を重ねるにつれ減衰させていく．

被験者は 22〜27 歳の男性 5 名であり，実験前にクラスタリングを行うソフトウェアの一覧を提示し，使用用途が分からないソフトウェアに対しては，簡単な説明を行った．

5.2.2　実験結果

図 10 は，被験者によるクラスタリング結果の評価である．質問 1，2 は，ユーザフィードバックをシステムに与える前後のクラスタリング結果がユーザの意図にどの程度近いか 5 段階で評価してもらった結果である．また，質問 3 はユーザフィードバックをシステムに与えた後のクラスタの移動がユーザにとって妥当であるか 5 段階で評価してもらった結果である．

提案手法に対して被験者がユーザフィードバックを与えたクラスタリング結果の例が**図 11** である．太い枠がそれぞれの参照ノードを表し，ユーザフィードバックを受けたソフトウェアは斜体で表している．図 11 のクラスタリング結果には，Visual Studio 系のソフトウェアや，音楽・動画プレイヤ，ブラウザといったソフトウェアを集めるというユーザの意図の下にユーザフィードバックが与えられて

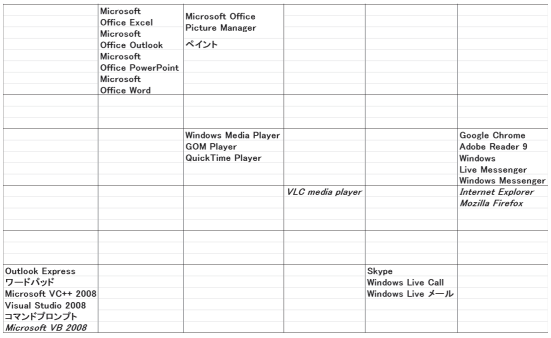

図 11　提案手法に対して被験者がユーザフィードバックを与えたクラスタリング結果 2

Fig. 11　Clustering result of proposed algorithm with user feedback.

いる．

6. 考察

6.1 実験 1

まず，Batch 型 SOM によるクラスタリング結果（図 7）と Weighted SOM によるクラスタリング結果（図 8）を比べる．両者のクラスタリング結果には音楽・動画プレイヤのクラスタや Microsoft Office のクラスタなど，共通して出現しているクラスタが存在している．これは，Weighted SOM における重みベクトルが Batch 型 SOM により得られた参照ベクトルを用いて更新されているため，Weighted SOM のクラスタリング結果が Batch 型 SOM の分類の影響を受けているからである．両者に同じクラスタが存在する一方で，Batch 型 SOM ではネットワークソフトとしてクラスタを構成していたソフトウェアが，Weighted SOM を用いたクラスタリング結果ではチャットソフトとインターネットブラウザのクラスタに分割されていることが分かる．これは，重みベクトルを用いることで各次元の関連度が変化したことが原因と考えられる．以上の結果から，Batch 型 SOM のクラスタリング結果を踏襲しつつ，分類能力を向上することができたと考えられる．

今回実験で使用したソフトウェアベクトルはその作成手順から，単語の出現頻度ベクトルであり，要素の大部分は 0 である．また，式 (14) から参照ノード k を勝者ノードとする入力ベクトル集合 $N_{k,0}$ 間で値が近い次元の重みが大きくなるように重みベクトルが更新されることが分かる．このことから，参照ベクトルの値が 0 の次元の重みが大きくなっていると考えられる．つまり，Weighted SOM によるソフトウェアクラスタリングでは，あるクラスタ内に出現しない単語を重視してクラスタリングを行っていることが分かる．

次に，Weighted SOM でのクラスタリング結果にユーザフィードバックを与える前後のクラスタリング結果（図 9）を比べる．今回の実験ではユーザフィードバックとして EmEditor の位置を変更している．この結果では，EmEditor という文書エディタに関係のある Microsoft Office のソフトウェアが EmEditor の近くに集まっていることが分かる．また，TEX の統合開発環境である WinShell は，文書エディタより統合開発環境と関係性が重みベクトルにより強調されて EmEditor と離れる方向に移動したと考えられる．このことから，ユーザフィードバックを与えることにより，クラスタリング結果に変化を与えることができたといえる．

ここで，ユーザフィードバックとして移動入力データ $\boldsymbol{x_i}$ を移動元参照ノード k_s から移動先参照ノード k_d へ移動させるときに生じるクラスタリング結果への影響について考える．重みベクトルは式 (14) を用いて更新されるため，移動元参照ノード k_s の重みベクトル $\boldsymbol{w_{k_s}}$ は元々移動元参照ノード k_s にクラスタリングされていた入力データから $\boldsymbol{x_i}$ を除いた入力データを用いて更新される．このとき，入力データの各次元のうち値が似通っている次元の重みを大きく，それ以外の次元の重みを小さくするように重みベクトルは更新される．そのため，移動元参照ノード k_s の重みベクトルは，元々移動元参照ノード k_s にクラスタリングされていた入力データから $\boldsymbol{x_i}$ を除いた入力データの間で共通して値が似ている重みの次元が大きくなり，それ以外の次元の重みが小さくなる．同様に移動先参照ノード k_d の重みベクトルは，移動先参照ノードにクラスタリングされていた入力データに $\boldsymbol{x_i}$ を加えた入力データを用いて更新される．このように更新された重みベクトルと参照ベクトルを用いて再度クラスタリングを行うため，参照ノード k_s と k_d の周囲にクラスタリングされている入力データのクラスタリングが変化する．以上の考察により，提案手法ではユーザフィードバックを与えた周囲のクラスタリングに影響を与えることができたことが分かる．

6.2 実験 2

まず，質問 1 の被験者評価結果から，それぞれの手法に対する評価値の平均を比べると，提案手法はユーザフィードバックを受ける前の初期クラスタリングにおいて，ユーザの意図に近いと評価されていることが分かる．しかし，被験者 No3 が提案手法によるクラスタリング結果が最も意図と遠いと評価していることから，ユーザによって良いクラスタリング結果の定義が大きく異なっていることが分かる．次に質問 2 の評価結果からユーザフィードバックを与えた後のクラスタリング結果に対する評価値の平均を比べると，SOM2 によるクラスタリング結果が最も良く，次いで提案手法，SOM1 との結果が得られた．ユーザフィードバックを与える前のクラスタリング結果に対する評価値の平均も考えると，SOM1，SOM2 はユーザフィードバックによりユーザの意図に近づいていることが分かる．その一方で，被験者 No4 は提案手法にユーザフィードバックを与えた後のクラスタリング結果が，ユーザの意図から少し

遠いと評価している．これは，提案手法にユーザフィードバックを与えた際，2 つの大きなクラスタが 1 つに統合するなどの，クラスタの大きな移動が生じていたためであると考えられる．

また，提案手法において初期クラスタリング結果とユーザフィードバックを与えた後のクラスタリング結果の評価にあまり変化が生じていないことが分かる．このことに対して，実験後に被験者から，提案手法の初期クラスタリング結果が，意図していたクラスタリング結果と非常に近いものであったため，ユーザフィードバックによる効果が表れにくいのではないかとの意見をいただいた．

質問 3 の評価結果から，提案手法は Nurnberger ら [5] の手法と比べ，ユーザフィードバックによるクラスタの移動がユーザにとって妥当であることが分かった．Nurnberger ら [5] の手法では，ユーザフィードバックとして入力データの適切な参照ノードを指定しても，ユーザフィードバックどおりにクラスタが移動しない場合が多かったため，被験者はクラスタの移動が妥当でないと評価していたと考えられる．

7. 終わりに

本研究では，SOM の各参照ノードに重みベクトルを付加することにより，各次元の重要度を調節し，クラスタリング性能を向上させる Weighted SOM を提案した．また，Weighted SOM にユーザフィードバックを与えることにより，ユーザの意図をクラスタリング結果に反映させる手法を提案した．実験により，初期クラスタリングにおいて提案手法の Weighted SOM は既存の SOM よりユーザの意図に近いクラスタリングがなされることが分かった．また，提案手法の Weighted SOM においてインタラクションフェーズでユーザフィードバックを与えることにより，ユーザにとって妥当なクラスタの移動が行えること分かった．今後の課題として，ソフトウェアのクラスタリングだけでなく，他のクラスタリングに提案手法を適用し，手法の有効性を調べる必要がある．さらに，ユーザフィードバックには様々なユーザの意図が考えられるため，それらに対応することができる重みベクトルの更新手法を考える必要がある．また，提案手法では入力データが何も分類されていない参照ノードの重みは更新されていない．しかし，SOM ではある参照ノードの近傍に存在する参照ベクトルの値は互いに似ているため，近傍参照ノードにおける次元の重要度も似ている可能性がある．そのため，周囲の重みベクトルを考慮して重みベクトルを更新する手法も考えられる．

参考文献

[1] 大橋靖雄，分類手法概論，計測と制御，Vol.24, No.11 (1985).

[2] Estivill, V.: Why so many clustering algorithms: a position paper, *ACM SIGKDD Explorations Newsletter*, Vol.4, Issue 1 (2002).

[3] Kohonen, T.: The self-organizing map, *Neurocomputing*, Vol.21, pp.1–6 (1998).

[4] Cohn, D., Caruana, R. and McCallum, A.: Semi-supervised Clustering with User Feedback, Technical Report TR2003-1892, Cornell University (2003).

[5] Nurnberger, A. and Detyniecki, M.: Weighted Self-Organizing Maps: Incorporating User Feedback, *Proc. Joined 13th International Conference on Artificial Neural Networks and Neural Information Processing*, pp.883–890 (2003).

[6] 山田誠二，岡部正幸，高間康史，小野田崇：最小ユーザフィードバックの枠組みとその要素技術，知能と情報（日本知能情報ファジィ学会誌），Vol.23, No.1, pp.80–85 (2011).

[7] Okabe, M. and Yamada, S.: Learning Similarity Matrix from Constraints of Relational Neighbors, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol.14, No.4, pp.402–407 (2010).

[8] Okabe, M. and Yamada, S.: An Interactive Tool for Human Active Learning in Constrained Clustering, *Journal of Emerging Technologies in Web Intelligence*, Vol.3, No.1 (2011).

[9] Domeniconi, C., Papadopoulos, D., Ma, S., Gunopulos, D. and Yan, B.: Locally Adaptive Metrics for Clustering High Dimensional Data, *Data Mining and Knowledge Discovery Journal*, Vol.14 (2007).

[10] Cheng, H., Hua, K.A. and Vu, K.: Constrained Locally Weighted Clustering, *Proc. VLDB Endowment* (2008).

[11] Parsons, L., Haque, E. and Liu, H.: Subspace Clustering for High Dimensional Data: A Review, *ACM SIGKDD Explorations Newsletter*, Vol.6, pp.90–105 (2004).

[12] 久田大地，吉川　毅，野中秀俊：自己組織化マップを用いたソフトウェア分類表示手法，第 26 回ファジィシステムシンポジウム (2010).

[13] Harman, D.: An Exerimental Study of Factors Important in Document Ranking, *SIGIR '86*, *Proc. 9th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.186–193 (1986).

久田 大地　（学生会員）

昭和 62 年生．平成 22 年北海道大学工学部卒業．平成 24 年北海道大学大学院情報科学研究科修士課程修了．自己組織化マップ，ユーザインタフェースに関する研究に従事．

吉川 毅　（正会員）

昭和 45 年生．平成 7 年北海道大学大学院工学研究科情報工学専攻修士課程修了．平成 13 年北海道大学大学院システム情報工学専攻博士後期課程修了．平成 14 年より北海道大学大学院工学研究科助手．平成 19 年より北海道大学大学院情報科学研究科助教．知能情報学，数理情報科学の研究に従事．博士（工学）．人工知能学会，日本応用数理学会各会員．

野中 秀俊　（正会員）

昭和 35 年生．昭和 60 年東京大学大学院工学系研究科計数工学専門課程修士課程修了．同年北海道大学大学院工学研究科情報工学専攻博士後期課程入学．同年北海道大学助手．現在北海道大学大学院情報科学研究科准教授．博士（工学）．知能情報学，ヒューマンインタフェースの研究に従事．計測自動制御学会，電子情報通信学会，ヒューマンインタフェース学会，日本知能情報ファジィ学会，日本人間工学会各会員．