情報処理学会論文誌 Vol.62 No.3 936–945 (Mar. 2021)

多観点類似度を用いた凝集型階層クラスタリング

藤原 勇二[1,a)] 古賀 久志[1,b)]

受付日 2020年6月4日, 採録日 2020年12月1日

概要：cosine 類似度は文書クラスタリングで頻繁に使われる代表的な類似度であり，2 点間の類似度を原点を基準点（始点）とする 2 ベクトルのなす角度から決定する．近年，ベクトルの基準点を原点ではなくデータセット内の様々な点に移動して類似度を計算する多観点類似度 MVS（MultiViewpoint-based Similarity）が提案され，非階層クラスタリングにおいて cosine 類似度よりも高い分類性能を達成した．本研究では MVS を，階層クラスタリングとくに群平均法と組み合わせたクラスタリングアルゴリズム MVS-AVE を提案する．MVS は cosine 類似度よりも複雑であるにもかかわらず，MVS-AVE はクラスタ間類似度を高速計算する工夫により，cosine 類似度を利用した群平均法 CS-AVE と同等の時間計算量で動作する．実文書データを用いた評価実験により，MVS-AVE が，CS-AVE と比べて計算時間をほとんど増やさずに分類精度を向上できることを示す．さらに，MVS-AVE はクラスタサイズ（各クラスタのメンバ数）の均衡性をコントロールする仕組みを容易に導入できるという面白い特徴を持つ．

キーワード：階層クラスタリング，多観点，類似度，時間計算量，群平均法

Application of MultiViewpoint-based Similarity to Hierarchical Clustering

Yuji Fujiwara[1,a)] Hisashi Koga[1,b)]

Received: June 4, 2020, Accepted: December 1, 2020

Abstract: The cosine similarity is a similarity measure useful for document clustering. The cosine similarity between two points is determined by the angle between their corresponding vectors which start from the origin. Recently, a new similarity measure called MVS (MultiViewpoint-based Similarity) which observes the vectors from multiple viewpoints is proposed. MVS is shown to outperform the cosine similarity, when they are combined with some non-hierarchical clustering algorithm. This paper proposes an agglomerative hierarchical clustering named as MVS-AVE which couples the average-link method with MVS. Despite MVS is more complex than the cosine similarity, MVS-AVE achieves the same time complexity as the average-link method with the cosine similarity because of an effort to compute the inter-cluster similarity efficiently. Experimentally in document clustering, MVS-AVE outputs more accurate clustering results than the average-link method with the cosine similarity almost without lengthening the running time. Interestingly, our algorithm can be expanded to control the size fairness among clusters.

Keywords: hierarchical clustering, multiview, similarity measure, time complexity, average-link

1. はじめに

データ集合を教師データを用いた事前学習なしにクラスタと呼ばれる部分集合に分割する手法をクラスタリングと呼ぶ．クラスタリングはビッグデータが隆盛する今日，大規模データに対するデータマイニングやアノテーションをサポートする重要技術である．クラスタリングの基本は，類似したデータどうしを同じクラスタに所属させることである．このため，データ間類似度はクラスタリングにおいて重要である．

cosine 類似度は，2 点間の類似度を，原点を基準点とする 2 ベクトルのなす角により評価するもので，文書データのような高次元で疎なデータに対する類似度指標としてよく用いられる．近年，Nguyen ら [1] は cosine 類似度における基準

[1] 電気通信大学
The University of Electro-Communications, Chofu, Tokyo 182–8585, Japan
[a)] fujiwara@sd.is.uec.ac.jp
[b)] koga@sd.is.uec.ac.jp

点を原点ではなく，分析対象のデータセットから複数選択する多観点類似度 MVS（MultiViewpoint-based Similarity）を提案した．そして，MVS を利用した k-means タイプの非階層型クラスタリングアルゴリズムが，cosine 類似度に基づく k-means アルゴリズムである spherical k-means [2] より文書を適切にクラスタリングできることを示した．

クラスタリング手法は，非階層クラスタリングと階層クラスタリングの 2 種類に大別される．階層クラスタリングは非階層クラスタリングと比べると，クラスタ数を指定せずにクラスタの階層構造を抽出できるのが長所である．

本論文は，MVS を取り入れた階層的クラスタリングアルゴリズムの実現を目的とする．とくに MVS を階層的クラスタリング手法の代表例である群平均法 [3] と組み合わせたアルゴリズム MVS-AVE を提案する．MVS は基準点を複数使用するため cosine 類似度よりも複雑であり，MVS-AVE は cosine 類似度を利用する通常の群平均法よりも実行速度が遅くなることが懸念される．しかし，本研究ではクラスタ間類似度を高速計算する手段を開発し，MVS-AVE の時間計算量を（多観点ではない）cosine 類似度を使った通常の群平均法と同等にすることに成功した．さらに，通常の群平均法とは違って，MVS-AVE はクラスタサイズ（各クラスタのメンバ数）の均衡性をコントロールする仕組みを容易に導入できるという面白い特徴を持つ．実文書データを用いた評価実験により，MVS-AVE が，cosine 類似度を用いた通常の群平均法より，計算時間をほとんど増やさず，分類精度を向上させることができることを示す．

本論文の構成は次のようになる．2 章では，提案手法の要素技術である (1) 多観点類似度 MVS と (2) 群平均法を紹介する．3 章では提案手法を記述する．4 章では文書クラスタリングに提案手法を適用し，提案手法を実験的に評価する．5 章で関連研究を記述し，6 章で結論を述べる．なお，本論文は我々の国際会議論文 [4] を発展させたものである．国際会議論文との差分は主に以下の 4 点である．

- クラスタ間類似度の更新式が正当であることを厳密に証明した．
- クラスタ間類似度を高速に更新する工夫が，実行速度の短縮に寄与することを実験的に確認した．
- 提案手法の比較対象にユークリッド距離ベースの群平均法を追加した．
- クラスタサイズの均衡性をコントロールする仕組みを導入しても，実行時間が増加しないことを実験的に示した．

2. 準備

2.1 多観点類似度

多観点類似度 MVS [1] は単位球面上に存在する単位ベクトルを対象とする類似度である．

単位球面上のデータ d_i，d_j 間の cosine 類似度 $\mathrm{CS}(d_i, d_j)$ は，ベクトルの内積 $d_i^{\mathrm{T}} d_j$ として定義される．cosine 類似度は，$\mathrm{CS}(d_i, d_j) = (d_i - 0)^{\mathrm{T}}(d_j - 0)$ のように，各データと原点 0 の差の内積としても表せる．これは cosine 類似度が原点を唯一の基準点として類似性を評価していることを表している．

MVS では，この基準点を与えられたデータセット S 中の様々な点に移動させて，cosine 類似度よりもデータ分布に適応した類似性評価を実現する．具体的には，MVS では式 (1) のように同一クラスタ r に所属する 2 データ d_i，d_j の類似度を，r に属さないデータ $d_h \in S \setminus S_r$ を基準点とし，各 d_h に対する差ベクトルの内積の平均値によって定義する．ここで，$n = |S|$ であり，クラスタ r のメンバを S_r，データ数 $n_r = |S_r|$ と表記する．

$$\begin{aligned}&\mathrm{MVS}(d_i, d_j \mid d_i, d_j \in S_r)\\&= \frac{1}{n - n_r} \sum_{d_h \in S \setminus S_r} (d_i - d_h)^{\mathrm{T}}(d_i - d_h)\end{aligned} \tag{1}$$

MVS では，クラスタ r に属さない点 d_h を基準として d_i，d_j 間の類似度が高いときに，d_i，d_j が同じクラスタ r に所属する妥当性が高いと判断する．つまり，データペアが同一クラスタに含まれることの妥当性を，データセットと無関係な原点ではなく，データセット内部の点を基準点として評価することで，クラスタリングにおける分類能力を向上させる．

一方，MVS に関して懸念される点は計算量である．式 (1) から分かるように，MVS は最大 $n - 2$ 回の内積の算術平均である．そのため，MVS の計算量は cosine 類似度の約 n 倍となる．

Nguyen ら [1] では，MVS を用いた非階層クラスタリング手法 MVSC を提案した．文書クラスタリングにおいて，MVSC は cosine 類似度をベースとする k-means 法である spherical k-means 等の既存手法より優れた分類性能を達成した．

2.2 群平均法

群平均法は凝集型階層クラスタリングアルゴリズムの 1 つであり，外れ値に強いことから多様なアプリケーションに適用されている．凝集型階層クラスタリングでは，分析対象のデータセット S に含まれる n 個の点が，1 点のみで構成される n 個のクラスタとなる初期状態からスタートする．その後，クラスタ数が 1 になるまで，最も類似度が高い 2 つのクラスタのマージを繰り返すことで階層的なクラスタ分割構造を得る．クラスタリング結果は樹形図（デンドログラム）で表され，樹形図から任意のクラスタ数での分割結果を得られる．

群平均法では，クラスタ a，b 間のクラスタ間類似度を式 (2) のようにクラスタメンバ間の類似度の平均値と定める．

1: $n \times n$ の類似度行列 A の初期化
2: **while** クラスタ数 > 1 **do**
3: 最も類似度の高いクラスタ a,b を探す
4: a,b をマージしてクラスタ c を作る
5: **for** $k \leftarrow c$ 以外のクラスタ **do**
6: クラスタ c と k の間の類似度を計算

図 1 群平均法
Fig. 1 Average-link method.

$$Sim_{ab} = \frac{1}{n_a n_b} \sum_{d_i \in S_a} \sum_{d_j \in S_b} \mathrm{sim}(d_i, d_j) \tag{2}$$

群平均法のアルゴリズムを**図 1** に示す．初期状態では 1 つの点が 1 クラスタなので，全点間で類似度を計算し類似度行列 A を作成する．その後は，最も近い 2 クラスタのマージを繰り返す．既存のクラスタ a，b をマージして新しいクラスタ c を作成したとき，そのクラスタ c とその他の既存クラスタ k 間で類似度 Sim_{kc} を計算し，類似度行列 A を更新する必要がある．Sim_{kc} は式 (3) に示す Sim_{ka} と Sim_{kb} の重み付き平均で表せることが知られている．Sim_{ka} と Sim_{kb} はクラスタ a，b をマージする直前の類似度行列に記憶されているので，式 (3) を用いてクラスタ間類似度 Sim_{kc} は $O(1)$ で計算できる．

$$Sim_{kc} = \frac{n_a}{n_c} Sim_{ka} + \frac{n_b}{n_c} Sim_{kb} \tag{3}$$

2.2.1 群平均法の計算量

ここでは図 1 に沿って，データ間類似度が cosine 類似度である場合の，群平均法の計算量を説明する．データの次元数を m とする．

1 行目の類似度行列の初期化ではすべてのデータペア間で cosine 類似度が計算される．cosine 類似度は m 次元ベクトルの内積であり $O(m)$ で計算可能なので，類似度行列を初期化する計算量は $O(mn^2)$ になる．

次に，クラスタ凝集の計算量を考える．3 行目の類似度が最大となるクラスタペアの探索は，ヒープを用いて $O(n \log n)$ で計算できる．6 行目では，クラスタ間類似度は式 (3) を利用して $O(1)$ で計算できる．クラスタ間類似度は，新クラスタと他の全クラスタとの間で計算されるため，5 行目の類似度行列更新に要する計算量は $O(n)$ である．

そしてクラスタリングが完了するまでに，クラスタペアが $n-1$ 回マージされるため，群平均法の計算量は，類似度行列の初期化と合わせて全体で $O(mn^2 + (n \log n + n) \times (n-1)) = O(mn^2 + n^2 \log n)$ になる．

3. 多観点類似度に基づく階層クラスタリング

Nguyen らは非階層クラスタリングにおいて，MVS が cosine 類似度より分類性能が優れていることを示した．この cosine 類似度に対する MVS の優位性は，階層クラスタリングでも期待できる．そこで本研究では，群平均法に MVS を導入した階層クラスタリングアルゴリズムを構築する．以降では，我々の提案手法のことを MVS-AVE と呼ぶ．MVS は cosine 類似度より計算量が高いにもかかわらず，MVS を導入した群平均法 MVS-AVE は cosine 類似度に基づく通常の群平均法（以降では CS-AVE と呼ぶ）と同等の計算量を達成する．また，後述するように MVS-AVE は，クラスタサイズ（各クラスタのメンバ数）の均衡性をコントロールする仕組みを容易に導入できるという特徴を持つ．

3.1 クラスタ間類似度

MVS を群平均法に適用するには，式 (2) に示したクラスタ間類似度の定義において，2 データ間の類似度指標 sim(,) を式 (4) のように多観点類似度 MVS に変更すればよい．

$$Sim_{ab} = \frac{1}{n_a n_b} \sum_{d_i \in S_a} \sum_{d_j \in S_b} \mathrm{MVS}(d_i, d_j) \tag{4}$$

しかし，式 (4) では異なるクラスタ a，b に属する 2 点 d_i，d_j 間で MVS を計算している．元々，MVS は式 (1) が示すように同一クラスタ内のデータペアに対する類似度であるため，式 (4) は well-defined でない．そこで，異なるクラスタに属するデータ d_i，d_j 間の MVS は，式 (5) のようにクラスタ a とクラスタ b をマージした仮想的なクラスタを考え，d_i と d_j が同じクラスタに所属する状況を作り出して定義する．式 (5) において a，b をマージしたクラスタのメンバは $S_a \cup S_b$ であり，メンバ数は $n_a + n_b$ になる．

$$\begin{aligned} &\mathrm{MVS}(d_i, d_j | d_i \in S_a, d_j \in S_b) \\ &= \frac{1}{n - n_a - n_b} \sum_{d_h \in S \setminus (S_a \cup S_b)} (d_i - d_h)^{\mathrm{T}} (d_j - d_h). \end{aligned} \tag{5}$$

3.2 群平均法で変更が必要な処理

3.1 節では MVS に基づくクラスタ間類似度を新たに導入した．したがって，群平均法に MVS を導入するにあたっては，クラスタ間類似度の計算を含む処理を修正する必要がある．該当する処理は (1) クラスタ間類似度行列 A の初期化と (2) 最も類似したクラスタペアをマージして新クラスタを作った後の類似度行列 A の更新の 2 カ所である．

まず，MVS を単純に群平均法に導入した場合の時間計算量を考えてみよう．類似度行列 A を初期化するには MVS を n^2 回計算する必要がある．1 回の MVS の計算は，m 次元ベクトルの内積計算を $n-2$ 回ともなうので計算量が $O(mn)$ となる．したがって，A の初期化には $O(mn^3)$ の時間がかかる．

一方，クラスタマージ後の類似度行列 A の更新に関しては，新クラスタ c と既存クラスタ k 間で類似度を計算する必要がある．通常の群平均法では式 (3) を用いて $O(1)$ の時間で Sim_{kc} を計算可能であった．しかし，式 (3) は，データ間類似度がクラスタメンバに依存しないことを前提条件とする．cosine 類似度はこの前提条件を

満足する．しかし，MVS ではクラスタメンバが変化すると基準点集合が変化しデータ間類似度も変わるため前提条件を満たさない．たとえば，クラスタ k に属するデータ d_i とクラスタ a に属する d_j 間の類似度 $\mathrm{MVS}(d_i, d_j)$ は $\frac{1}{n-n_k-n_a}\sum_{d_h\in S\setminus(S_k\cup S_a)}(d_i-d_h)^{\mathrm{T}}(d_j-d_h)$ である．ここでもし，クラスタ a が他のクラスタ b とマージされたなら，$\mathrm{MVS}(d_i, d_j)$ は $\frac{1}{n-n_k-n_a-n_b}\sum_{d_h\in S\setminus(S_k\cup S_a\cup S_b)}(d_i-d_h)^{\mathrm{T}}(d_j-d_h)$ となり値が変わる．つまり，$\mathrm{MVS}(d_i, d_j)$ は d_j の所属クラスタに依存する．このため，MVS-AVE では式 (3) を用いてクラスタ間類似度を更新できない．

更新式に頼らず，式 (4) の定義に従って，クラスタマージ後の Sim_{kc} を求める計算量は，MVS を $n_k n_c$ 回計算するため $\Theta(n_k n_c \times m(n-n_k-n_c))$ であり，$\Omega(mn)$ かかる．そして，階層クラスタリングではクラスタが $n-1$ 回マージされ，マージ 1 回あたりクラスタ間類似度は平均で $\frac{n}{2}$ 回計算される．よって，A の更新に要する計算量の下界は $\Omega(mn\times(n-1)\times\frac{n}{2})=\Omega(mn^3)$ となる．これは通常の群平均法の計算量 $O(mn^2+n^2\log n)$ より大きく，単純に MVS を導入すると群平均法の計算量が大幅に悪化する．

3.3 処理の高速化

本節では，類似度行列 A の初期化および更新を高速化する手法を提案する．これらの高速化手段により，MVS-AVE の計算量は $O(mn^2+n^2\log n)$ に削減され，cosine 類似度を用いた通常の群平均法 CS-AVE と同等に抑えられる．

3.3.1 類似度行列の初期化

ここでは，MVS に基づく類似度行列を計算量 $O(mn^2)$ で初期化する手法を説明する．とくに，この手法では m 次元ベクトルの内積を CS-AVE とまったく同じ回数だけ計算する．

階層クラスタリングでは，初期状態ではクラスタ数は n 個存在し，各クラスタはデータを 1 つだけ保有する．よって，式 (4)，(5) より群平均法におけるクラスタ間類似度は，式 (6) で表せる．$|d_h|=1$ であることに注意されたい．

$$
\begin{aligned}
Sim_{ij} &= \mathrm{MVS}(d_i, d_j)\\
&= \frac{1}{n-2}\sum_{d_h\in S\setminus(d_i\cup d_j)}(d_i-d_h)^{\mathrm{T}}(d_j-d_h)\\
&= \frac{1}{n-2}\sum_{d_h\in S\setminus(d_i\cup d_j)}(d_i^{\mathrm{T}}d_j-d_i^{\mathrm{T}}d_h-d_j^{\mathrm{T}}d_h+1)\\
&= \frac{1}{n-2}\big\{(n-2)d_i^{\mathrm{T}}d_j-d_i^{\mathrm{T}}(D-d_i-d_j)\\
&\qquad -d_j^{\mathrm{T}}(D-d_i-d_j)+(n-2)\big\}\\
&= \frac{1}{n-2}(nd_i^{\mathrm{T}}d_j-d_i^{\mathrm{T}}D-d_j^{\mathrm{T}}D+n). \qquad (6)
\end{aligned}
$$

ここで，D は全データの総和ベクトル $\sum_{d_i\in S}d_i$ である．式 (6) を眺めると D を事前計算しておけば式 (6) は m 次元ベクトルの内積を 3 回計算すれば求まることが分かる．

1: **for** $i\leftarrow 1,\ldots,n$ **do**
2: **for** $j\leftarrow 1,\ldots,n$ **do**
3: $\mathrm{CS}(d_i,d_j)=d_i^{\mathrm{T}}d_j$
4: **for** $i\leftarrow 1,\ldots,n$ **do**
5: $d_i^{\mathrm{T}}D=\sum_{j=1}^{n}\mathrm{CS}(d_i,d_j)$
6: **for** $i\leftarrow 1,\ldots,n$ **do**
7: **for** $j\leftarrow 1,\ldots,n$ **do**
8: $Sim_{ij}=\frac{1}{n-2}(n\mathrm{CS}(d_i,d_j)-d_i^{\mathrm{T}}D-d_j^{\mathrm{T}}D+n)$

図 2 MVS-AVE における類似度行列 A の初期化
Fig. 2 Initialization of similarity matrix A.

よって，式 (6) の計算量は $O(m)$ になり，大きさ $n\times n$ の類似度行列 A は $O(mn^2)$ で初期化できることになる．

この $O(mn^2)$ という計算量は，CS-AVE が A を初期化するときの計算量とオーダは等しい．しかし定数倍レベルの差異を考慮すると，類似度行列の 1 要素を求める際に，m 次元ベクトルの内積を CS-AVE では 1 回だけ計算するのに対し，MVS-AVE では 3 回計算している．したがって，類似度行列 A の初期化にかかる時間計算量は MVS-AVE が CS-AVE の約 3 倍になってしまう．文書データのように次元数 m が大きいとき，$O(mn^2)$ の項が群平均法の計算量を支配するため，類似度行列の初期化にかかる時間が 3 倍に増加するのは望ましくない．

そこで，1 回の類似度計算あたりの内積計算回数を 1 回に削減する方法を示す．基本アイデアは D を事前計算するのではなく，すべての $1\le i\le n$ に対して $d_i^{\mathrm{T}}D$ を事前計算しておくことである．$d_i^{\mathrm{T}}D$ はデータ点 d_i と関連する n 種類の類似度 $\mathrm{MVS}(d_i, d_j)$（$1\le j\le n$）の計算でのべ n 回使われる．よって，$d_i^{\mathrm{T}}D$ を事前計算するメリットがある．

具体的にはまず，最初に全データ対 d_i，d_j 間で cosine 類似度 $d_i^{\mathrm{T}}d_j$ を計算する．この処理だけで m 次元の内積が n^2 回計算される．しかし，このように全データペア間で cosine 類似度をあらかじめ計算しておくと，$d_i^{\mathrm{T}}D=\sum_{j=1}^{n}d_i^{\mathrm{T}}d_j$ なので，$d_i^{\mathrm{T}}D$ は $O(n)$ で計算できる．さらに，式 (6) は，$d_i^{\mathrm{T}}d_j$，$d_i^{\mathrm{T}}D$，$d_j^{\mathrm{T}}D$ が事前計算されていれば，内積の計算なしに $O(1)$ で計算できる．

図 2 に MVS-AVE における類似度行列 A の初期化アルゴリズムをまとめる．このアルゴリズムは，(Step 1) 全データペア間での cosine 類似度の計算，(Step 2) すべての $1\le i\le n$ に対する $d_i^{\mathrm{T}}D$ の事前計算，(Step 3) 事前計算した cosine 類似度及び $d_i^{\mathrm{T}}D$（$i=1,\ldots,n$）を用いた類似度行列 A の生成という 3 ステップからなるので，時間計算量は $O(mn^2+n\times n+O(1)\times n^2)=O(mn^2+n^2)$ である．とくに類似度計算 1 回あたりの m 次元ベクトルの内積計算が 1 回になっており，上述した D を事前計算するやり方よりも内積計算回数を $\frac{1}{3}$ に減らせている．

3.3.2 マージ後の類似度の更新

3.2 節では，MVS に対しては通常の群平均法における類似度更新式（式 (3)）を適用できないことを述べた．本節では，MVS を対象とする新しい類似度更新式を導出し，そ

れが高速計算できることを示す．

MVS を適用した群平均法において，クラスタ a，b をマージしてできた新クラスタ c と既存クラスタ k とのクラスタ間類似度 Sim_{kc} は，Sim_{ka}，Sim_{kb} を用いて以下の定理のように表せる．定理の証明は付録に載せる．

定理 1. D_a，D_b をそれぞれクラスタ a，b に属するデータの総和ベクトル，n_a，n_b はクラスタ a，b を構成するデータ数とする．マージ後のクラスタ間類似度 Sim_{kc} は，マージ前のクラスタ間類似度 Sim_{ka}，Sim_{kb} を用いて以下の式 (7) により表せる．

$$\begin{aligned} Sim_{kc} = {} & \frac{1}{(n_a+n_b)(n-n_k-n_a-n_b)} \\ & \Big\{ n_a(n-n_k-n_a)Sim_{ka} \\ & + n_b(n-n_k-n_b)Sim_{kb} \\ & + 2\left(D_a^{\mathrm{T}}D_b - n_a n_b\right) \Big\} \end{aligned} \tag{7}$$

式 (7) は，各クラスタ（a とする）に対して総和ベクトル D_a を一緒に記憶させれば計算できる．しかし，これでは MVS-AVE は CS-AVE と同等の計算量を達成できない．その理由は CS-AVE は更新式 (3) を $O(1)$ で計算できるのに対し，MVS-AVE は更新式 (7) を D_a と D_b との内積を求め $O(m)$ で計算するからである．階層クラスタリングでは，クラスタは $n-1$ 回マージされ，1 回のマージあたりクラスタ間類似度が平均で $\frac{n}{2}$ 回計算されるため，類似度行列 A を更新する総計算量は CS-AVE で $O(1\times(n-1)\times\frac{n}{2}) = O(n^2)$ となり，MVS-AVE では $O(m\times(n-1)\times\frac{n}{2}) = O(mn^2)$ となる．その結果，MVS-AVE の計算量は CS-AVE の計算量より大きくなってしまう．

そこで本論文では，MVS を導入しても A の更新に要する総計算量が $O(n^2)$ に収まる手法を実現する．この手法ではクラスタごとに総和ベクトルを記憶するのではなく，全クラスタペアの総和ベクトルの内積を記憶する内積行列 B を保持する．3.3.1 項で述べたように，MVS-AVE では類似度行列 A を高速に初期化するために全データペア d_i，d_j 間で $\mathrm{CS}(d_i,d_j) = d_i^{\mathrm{T}}d_j$ を事前計算する．これはクラスタ数が n の初期状態において，全クラスタペア間で総和ベクトルの内積を計算し内積行列 B を初期化したとも見なせる．

次にクラスタマージ後の内積行列 B の更新手順を説明する．クラスタ a，b のマージによる新しいクラスタ c を生成したとき，c の総和ベクトルとその他のクラスタ k の総和ベクトルの内積は，$D_c^{\mathrm{T}}D_k = \sum_{d_i\in S_c}\sum_{d_j\in S_k} d_i^{\mathrm{T}}d_j = \sum_{d_i\in(S_a\cup S_b)}\sum_{d_j\in S_k} d_i^{\mathrm{T}}d_j = \sum_{d_i\in S_a}\sum_{d_j\in S_k} d_i^{\mathrm{T}}d_j + \sum_{d_i\in S_b}\sum_{d_j\in S_k} d_i^{\mathrm{T}}d_j = D_a^{\mathrm{T}}D_k + D_b^{\mathrm{T}}D_k$ であるため，すでに B に記憶されているスカラー値 $D_a^{\mathrm{T}}D_k$ と $D_b^{\mathrm{T}}D_k$ を足して $O(1)$ で更新できる．

このようにして内積行列 B を保持すれば，式 (7) を計算するときには $D_a^{\mathrm{T}}D_b$ は既知なので，式 (7) は $O(1)$ で計算可能になる．以上をまとめると，類似度 Sim_{kc} と $D_c^{\mathrm{T}}D_k$ の両者が $O(1)$ で更新できる．そして，類似度行列 A および内積行列 B を更新する総計算量は $O(1\times(n-1)\times\frac{n}{2}) = O(n^2)$ に抑えられる．

3.4 MVS-AVE の計算量

本節では MVS-AVE の計算量が，CS-AVE と同様に $O(mn^2+n^2\log n)$ となることを確認する．類似度行列の初期化は，3.3.1 項に示したとおり計算量 $O(mn^2+n^2)$ になる．クラスタのマージに関しては，

- 最も類似したクラスタペア a，b はヒープを用いて CS-AVE とまったく同様に $O(n\log n)$ で発見できる．クラスタは $n-1$ 回マージされるため，凝集ステップでの総計算量は $O(n^2\log n)$ となる．
- 類似度行列 A と内積行列 B の更新は，凝集ステップ全体で 3.3.2 項で述べたように $O(n^2)$ かかる．

以上を合計すると，MVS-AVE の計算量は，$O(mn^2+n^2+n^2\log n+n^2) = O(mn^2+n^2\log n)$ になる．

3.5 クラスタサイズの均衡化

一般にクラスタリングを利用したデータ分析では，クラスタ間でメンバ数が偏り，非常に大きなクラスタと非常に小さいクラスタが混在する状況は好まれない．データ分析者は似たサイズのクラスタが抽出されることを期待していることが多い．実際，Normalized Cuts [5] のように極小クラスタを防止してクラスタサイズを均衡化する仕組みを取り入れたクラスタリング手法も存在する．

CS-AVE のような凝集型階層クラスタリングは，クラスタサイズを均衡化する手段を持たないが，MVS-AVE では，クラスタサイズの均衡性を制御する機能を容易に導入できる．具体的には，類似度行列 A の初期化時に $\widehat{\mathrm{MVS}}(d_i,d_j\mid\lambda) = \mathrm{MVS}(d_i,d_j) - \lambda$ のように全要素から正の定数 λ を一律に減算すればよい．λ はパラメータで，λ が大きいほど均衡化へのバイアスを強くできる．

これでクラスタサイズを均衡化できる根拠は次のように説明できる．類似度の更新式 (7) において，Sim_{ka} の係数 $\frac{n_a(n-n_k-n_a)}{(n_a+n_b)(n-n_k-n_a-n_b)}$ と Sim_{kb} の係数 $\frac{n_b(n-n_k-n_b)}{(n_a+n_b)(n-n_k-n_a-n_b)}$ の合計は $\frac{(n_a+n_b)(n-n_k-n_a-n_b)+2n_an_b}{(n_a+n_b)(n-n_k-n_a-n_b)}$ となり 1 より真に大きい．したがって，クラスタがマージされ更新式 (7) が評価されるたびに $-\lambda$ の項が増幅される．その結果，類似度行列内で，より多数回のマージを経た大きなクラスタと関連する類似度は，より少数回のマージしか経ていない小さなクラスタと関連する類似度よりも相対的に小さくなりやすくなる．その結果，小さなクラスタが相対的にクラスタマージの対象になりやすくなり，均衡性の高いクラスタ分割が作られる．

なお，通常の群平均法に対する更新式 (3) では Sim_{ka}，

Sim_{kb} の係数の合計は $\frac{n_a}{n_c}+\frac{n_b}{n_c}=1$ であり，クラスタマージにより $-\lambda$ の項が増幅されることはない．したがって，類似度行列 A の要素を一律に λ だけ減算しても，クラスタサイズの均衡化はできない．クラスタサイズの均衡化は多観点類似度ならではの機能である．

4. 評価実験

本章では，実文書データをクラスタリングする実験を行い，提案手法である MVS-AVE が，cosine 類似度を用いた通常の群平均法である CS-AVE と比較してほぼ同等の実行時間で分類精度を向上させることができることを示す．さらに，3.5 節で述べたクラスタサイズ均衡化の効果も評価する．

実験プラットフォームは PC（CPU: Intel Core i7-4770 3.4 GHz，OS: Ubuntu 18.04，メモリ 16 GB）である．なお，文書データは多くの次元が 0 の値をとる高次元疎ベクトルであるため，内積演算を以下のように最適化した．$x=(x_1,x_2,\cdots,x_m)$ と $y=(y_1,y_2,\cdots,y_m)$ を 2 つの m 次元ベクトルとすると，その内積は $x^{\mathrm{T}}y=\sum_{i=1}^{m}x_iy_i$ である．ここで x_i，y_i のどちらかが 0 であるならば第 i 次元の乗算は内積に影響しない．この性質に着目し，$x_i=0$ または $y_i=0$ であるときに乗算および加算を省略することで，高次元疎ベクトル用の高速な内積演算を実装した．

4.1 データセット

実験には，**表 1** に示す 18 種類のデータセットを用いる．これらのデータセットは，テキストデータマイニングツール CLUTO 上で提供されている [6]．

表 1 実験で使用する文書データセット
Table 1 Dataset.

データセット	情報源	c	n	m
fbis	TREC	17	2,463	2,000
hitech	TREC	6	2,301	13,170
k1a	WebACE	20	2,340	13,859
k1b	WebACE	6	2,340	13,859
la1	TREC	6	3,204	17,273
la2	TREC	6	3,075	15,211
re0	Reuters	13	1,504	2,886
re1	Reuters	25	1,657	3,758
tr31	TREC	7	927	10,127
reviews	TREC	5	4,096	23,220
wap	WebACE	20	1,560	8,440
la12	TREC	6	6,279	21,604
new3	TREC	44	9,558	36,306
sports	TREC	7	8,580	18,324
tr11	TREC	9	414	6,424
tr12	TREC	8	313	5,799
tr23	TREC	6	204	5,831
tr45	TREC	10	690	8,260

表中において，c はデータセットの持つクラス数，n はデータ数，m は単語数である．データセットにストップワード除去とステミングを適用後，各データを TF-IDF でベクトル化し，さらに単位ベクトルに正規化する．

4.2 他手法との比較

MVS-AVE を文書クラスタリングに適用し，分類精度および実行時間を多観点ではない通常の類似度に基づく群平均法と比較する．比較対象は，cosine 類似度に基づく群平均法 CS-AVE とユークリッド距離に基づく群平均法 EUC-AVE の 2 つである．

クラスタリングアルゴリズムの分類精度は，正解クラスラベルによるデータ分割 X とクラスタ数を正解クラス数に揃えたクラスタリングアルゴリズムによるデータ分割 Y がどれだけ一致するかで評価する．分割の一致度は正規化相互情報量 $\mathrm{NMI}(X,Y)=\frac{I(X,Y)}{\sqrt{H(X)H(Y)}}$ により評価する．NMI はクラスタリング結果 Y が，正解クラスラベルの分布 X とどれだけ相関を持つかを評価する．NMI の値域は 0 から 1 までで，1 に近いほど X と Y の分割結果が近いことになる．$I(X,Y)=\sum_i\sum_j P(x_i,y_j)\log\frac{P(x_i,y_j)}{P(x_i)p(y_j)}$ は相互情報量であり，$H(X)$ はデータ分割 X のエントロピー $-\sum_i P(x_i)\log P(x_i)$ である．さらに，$P(x_i,y_j)=\frac{\text{クラス } i \text{ に属するクラスタ } j \text{ のデータ数}}{n}$ である．

表 2 左に分類精度を示す．この表では MVS-AVE，CS-AVE，EUC-AVE をそれぞれ “MVS”，“CS”，“EUC” と表記する．18 個中 13 個のデータセットに対して CS-AVE よ

表 2 分類精度および処理時間の評価
Table 2 Classification accuracy and execution time.

データ	分類精度			処理時間（秒）		
	MVS	CS	EUC	MVS	CS	EUC
fbis	**0.570**	0.561	0.042	20.12	20.09	20.11
hitech	**0.250**	0.059	0.026	17.29	17.08	17.11
k1a	**0.556**	0.550	0.080	17.93	17.51	17.63
k1b	**0.710**	0.666	0.011	17.83	17.62	17.72
la1	**0.383**	0.316	0.014	43.39	42.28	42.16
la2	0.374	**0.390**	0.013	37.15	37.03	36.88
re0	**0.312**	0.296	0.039	4.386	4.378	4.355
re1	0.540	**0.568**	0.077	6.158	6.061	6.000
tr31	**0.670**	0.527	0.027	1.624	1.521	1.541
reviews	**0.410**	0.034	0.033	86.70	86.19	87.14
wap	**0.554**	0.539	0.098	5.852	5.652	5.654
la12	0.367	**0.381**	0.009	289.2	288.9	288.3
new3	**0.535**	0.487	0.044	939.1	938.4	946.9
sports	**0.242**	0.106	0.008	692.8	688.3	688.9
tr11	**0.645**	0.633	0.089	0.217	0.197	0.200
tr12	0.472	**0.523**	0.081	0.120	0.107	0.108
tr23	**0.252**	0.223	0.048	0.060	0.050	0.051
tr45	0.363	**0.495**	0.062	0.724	0.685	0.695
平均	**0.456**	0.409	0.045			

りも MVS-AVE のほうが高い分類精度となった．さらに NMI の差は，MVS-AVE が CS-AVE を上回る場合のほうが，CS-AVE が MVS-AVE を上回る場合より平均的に大きい．全データセットに対する平均値は MVS-AVE が 0.456 であり，CS-AVE が 0.409 となり，MVS-AVE のほうが大きくなった．

EUC-AVE の分類精度は他 2 手法に比べて明確に低くなった．この結果は文書データのような高次元疎ベクトルの類似性をユークリッド距離で測るのは効果的でないことを示している．群平均法以外の代表的な凝集型階層クラスタリングである Ward 法でも分類精度を測定したが，全データセットに対する NMI の平均値は 0.030 と EUC-AVE と同様に低い値となった．これは，Ward 法がクラスタ内分散をユークリッド距離に基づいて求めることが原因と考えられる．

次に，表 2 右に実行時間を示す．この表より，MVS-AVE と CS-AVE の実行時間の差はとても小さく，データ数 n，単語数 m が十分大きいデータセットである reviews，la12，news3，sports に対しても 5 秒未満に収まった．

以上より，提案手法は多観点類似度による高い分類精度をほとんどオーバヘッドを増やさずに群平均法に組み込むことに成功したといえる．

4.3 高速化テクニックの効果

3.3 節で述べたように MVS-AVE では，(1) 類似度行列 A の初期化および (2) 更新式の評価を高速化するテクニックを実現している．その効果を実験により確認する．

4.3.1 類似度行列の初期化

類似度行列 A の初期化に関しては，総和ベクトル D の代わりにすべての i に対して $d_i^T D$ を事前計算することで m 次元の内積計算回数を $\frac{1}{3}$ に減らせることを説明した．このテクニックがクラスタリングの実行時間に与える影響を調べるため，

- 手法 1：全データの総和ベクトル D を事前計算し，$\mathrm{MVS}(d_i, d_j)$ を計算するときに，$d_i^T D$，$d_j^T D$ および $d_i^T d_j$ の m 次元ベクトルの内積を 3 回計算する手法
- 手法 2：全データペア間で cosine 類似度を事前計算後に，すべての i に対して $d_i^T D$ を事前計算することで，$\mathrm{MVS}(d_i, d_j)$ を m 次元ベクトルの内積を 1 回計算するだけで算出する手法

の 2 手法を実装し，類似度行列の初期化にかかる時間を比較した．結果を**表 3** にまとめる．

手法 1 は手法 2 と比べて，内積の計算回数が 3 倍なので，処理時間もおよそ 3 倍になることを予想したが，実験結果では最大で 30 倍以上まで増加するケースもあった．この現象は，高次元疎ベクトルを想定した内積演算の最適化によって引き起こされたと考えられる．データセットの個々のデータ d_i は疎ベクトルであるが，総和ベクトル $D = \sum_i d_i$ は必ずしも疎ではない．このため，$d_i^T d_j$ の内積計算には最適化が有効に働くが，$d_i^{\mathrm{T}} D$，$d_j^{\mathrm{T}} D$ の内積計算では最適化の効果は限定的になると考えられる．$\mathrm{MVS}(d_i, d_j)$ を計算するときに，手法 2 では $d_i^T d_j$ しか計算しないのに対し，手法 1 では $d_i^T d_j$，$d_i^{\mathrm{T}} D$ および $d_j^{\mathrm{T}} D$ を計算するため，実行時間の差が 3 倍以上に広がったと考えられる．

表 3 類似度行列初期化の処理時間
Table 3 Processing time to initialize similarity matrix A.

データ	処理時間（秒）		データ	処理時間（秒）	
	手法 1	手法 2		手法 1	手法 2
fbis	18.09	3.372	reviews	334.6	13.29
hitech	62.76	3.194	wap	19.43	1.266
k1a	67.15	2.909	la12	768.7	24.38
k1b	67.03	2.909	new3	2,809	84.87
la1	155.0	5.746	sports	1,153	42.75
la2	127.5	5.138	tr11	1.315	0.146
re0	6.308	0.437	tr12	0.697	0.083
re1	9.680	0.605	tr23	0.334	0.043
tr31	9.138	0.747	tr45	4.41	0.434

手法 1 では CS-AVE と同等の実行時間を達成するのは困難であり，MVS-AVE と CS-AVE が同等の時間になるという結果は，すべての i に対して $d_i^T D$ を事前計算するテクニックによるといえる．

4.3.2 更新式評価の高速化

3.3.2 項では，クラスタ a，b をマージ後に新クラスタ c と既存クラスタ k とのクラスタ間類似度を算出する際に，事前計算された $D_a^{\mathrm{T}} D_b$ を用いて類似度行列の更新式 (7) を内積計算を行わずに高速に評価する手法を提案した．この手法の効果を実験的に評価するため，

- 手法 1：更新式評価時に D_a と D_b の内積計算を $O(m)$ で行う手法
- 手法 2：$D_a^{\mathrm{T}} D_b$ を事前計算し，更新式を $O(1)$ で評価する高速化手法

の 2 手法を実装し，類似度行列初期化後のクラスタのマージにかかる時間を比較した．結果を**表 4** にまとめる．全データセットに対して，手法 2 のほうが手法 1 より処理時間が短くなっており，提案した更新式の評価法が MVS-AVE の高速性に確かに寄与していることを確認できた．

4.4 クラスタサイズ均衡化の効果

3.5 節で述べたクラスタサイズの均衡化手法が有効に働くかを評価した．以下では均衡化手法を導入した MVS-AVE のことを Balanced MVS-AVE と呼び，BMVS-AVE と記述する．クラスタサイズが均衡化しているかどうかは $\text{Fairness Index} = \frac{n^2}{K \sum_{i=1}^{K} n_i^2}$ により評価する．ここで K はクラスタ数，n_i は第 i 番目のクラスタのデータ数である $(1 \leq i \leq K)$．Fairness Index は K 個のクラスタが均等に $\frac{n}{K}$ ずつデータを保持するときに最大値 1 となり，値が

表 4 クラスタのマージの処理時間
Table 4 Processing time for cluster merging.

データ	処理時間（秒）		データ	処理時間（秒）	
	手法 1	手法 2		手法 1	手法 2
fbis	17.98	16.61	reviews	79.52	72.67
hitech	15.20	13.86	wap	4.875	4.481
k1a	15.53	14.77	la12	274.0	263.4
k1b	16.04	14.68	new3	894.5	849.7
la1	39.82	37.19	sports	656.1	647.9
la2	34.24	31.63	tr11	0.153	0.056
re0	4.075	3.888	tr12	0.072	0.028
re1	5.668	5.475	tr23	0.060	0.011
tr31	1.285	0.820	tr45	0.539	0.256

表 5 クラスタサイズに関する Fairness Index，太字は MVS-AVE を上回ったことを表す
Table 5 Fairness Index with respect to the size of clusters.

	Fairness Index			分類精度	処理時間（秒）
	MVS	BMVS		BMVS ($\lambda = \frac{2}{n}$)	
データ		$\lambda = \frac{2}{n}$	$\lambda = \frac{10}{n}$		
fbis	0.299	**0.322**	**0.325**	0.584	20.19
hitech	0.483	**0.492**	**0.567**	0.255	17.18
k1a	0.254	0.254	0.251	0.556	17.89
k1b	0.343	0.343	**0.467**	0.710	17.86
la1	0.373	**0.377**	**0.526**	0.376	42.64
la2	0.285	**0.384**	**0.395**	0.466	37.17
re0	0.453	0.453	**0.474**	0.312	4.389
re1	0.258	0.258	**0.356**	0.540	6.132
tr31	0.561	0.561	**0.577**	0.670	1.598
reviews	0.385	0.382	0.385	0.420	86.30
wap	0.291	0.291	**0.297**	0.554	5.747
la12	0.272	**0.373**	**0.372**	0.426	288.8
new3	0.314	0.314	0.314	0.535	945.6
sports	0.189	**0.191**	**0.368**	0.238	685.6
tr11	0.507	0.507	**0.510**	0.645	0.216
tr12	0.232	**0.297**	**0.878**	0.553	0.121
tr23	0.465	**0.472**	**0.651**	0.260	0.060
tr45	0.156	**0.291**	**0.487**	0.558	0.742
平均				0.481	

大きいほどクラスタサイズが均衡化していることになる．BMVS におけるパラメータは $\lambda = \frac{2}{n}$，$\frac{10}{n}$ の 2 つの値を試した．

表 5 左にクラスタ数 K をクラス数 c に設定したときの結果を示す．この表では MVS-AVE，BMVS-AVE をそれぞれ “MVS”，“BMVS” と表記する．$\lambda = \frac{2}{n}$ のときには，BMVS-AVE は 9 個のデータセットに対して MVS-AVE より Fairness Index が上回り，“reviews” に対してのみ Fairness Index が下回った．残りのデータセットに対してはクラスタリング結果が MVS-AVE から変化せず Fairness Index も不変であった．λ を $\frac{10}{n}$ に増やすと 15 個のデータセットに対して MVS-AVE より Fairness Index が上回るようになった．このように λ を増やすにつれて Fairness Index が増加する傾向が確認され，BMVS-AVE がクラスタサイズを均衡化することが確認できた．

なお参考までに，$\lambda = \frac{2}{n}$ に設定した BMVS-AVE による分類精度と実行時間を表 5 右に示す．全データセットに対する NMI の平均値が 0.481 となり，分類精度が MVS-AVE を上回った．実行時間は表 2 に示した MVS-AVE とほぼ同じであり，MVS-AVE より実行時間が短くなったデータセットもあった．BMVS-AVE と MVS-AVE を比べたときに，BMVS-AVE の余計なオーバヘッドは類似度行列の初期化時に $\widehat{\mathrm{MVS}}(d_i, d_j \mid \lambda) = \mathrm{MVS}(d_i, d_j) - \lambda$ のように定数 λ を引くことだけである．この引き算は $\mathrm{MVS}(d_i, d_j)$ 自体の算出よりはるかにオーバヘッドが小さい．このため，定数 λ を引くオーバヘッドは時間計測の誤差やクラスタリング結果の違いによる実行時間の差違に隠蔽されたと考えられる．

5. 関連研究

近年，異なるセンサから取得された複数のヘテロな特徴を持つデータを対象とするマルチビュークラスタリングが注目を集めている [7], [8]．ここでのビューとは特徴検出器のことである．マルチビュークラスタリングの目的は，個々のビュー（から得られた特徴）に対するクラスタリング結果を統合してより適切なクラスタリング結果を得ることである．本論文で取り扱う多観点類似度 MVS はマルチビュークラスタリングとはまったく異なる．MVS は複数のクラスタリング結果を統合するものではなく，類似度を測る基準点を移動しデータ分布に適応的な類似度を得るアプローチである．

多観点類似度 MVS に関する関連研究は以下がある．Yan ら [9] は MVS を用いた半教師ありクラスタリングアルゴリズムを開発した．Ravoori ら [10] は我々の研究と同様に MVS を群平均法を組み込んだと主張している．しかし，文献 [10] では MVS を群平均法にどう組み込んだかが説明されておらず，通常の群平均法が提案手法として紹介されている．その結果，我々の論文の 3 章に相当する内容が欠落しており，我々の論文における異なるクラスタに属するデータ間で MVS を計算するための拡張（式 (5)）や類似度行列の更新式 (7) は提案されていない．さらに計算量の理論的な解析や実行速度の実験評価がなされていない．

最後に文書クラスタリングに対する類似度に関する研究に言及する．文献 [11] では，特徴ベクトルの一部の次元だけを選んで cosine 類似度を計算する手法を提案している．ここでは類似度計算の対象ペアに応じてどの次元を選択するかを適応的に変えている．文献 [12] では 2 つの特徴ベクトル間で cosine 類似度を計算する際に，片方のベクトルだけが非零である次元に対してはペナルティとしてマイナスの定数を割り当てる手法を提案した．これらの手法はクラ

スタリング精度に関しては通常の cosine 類似度を凌駕するが，計算が複雑化するため実行時間が遅くなる．

本研究では，凝集型階層クラスタリングにおいて実行時間を増やさずに cosine 類似度を上回るクラスタリング精度を達成した点で価値がある．

6. 結論

本研究では，Nguyen らが提唱した多観点類似度 MVS を階層クラスタリングの群平均法に組み込んだ手法 MVS-AVE を提案した．MVS を単純に群平均法に組み込むと群平均法の時間計算量が増加する状況において，MVS-AVE では，類似度行列の初期化およびマージ後のクラスタ間類似度の更新を高速化し，通常の群平均法と同等の計算量 $O(mn^2 + n^2 \log n)$ を達成した．そして，実文書データを用いた実験により，MVS-AVE が通常の群平均法より高い分類精度を，わずかな計算時間の増加で達成できることを示した．

さらに，通常の群平均法とは異なり，MVS-AVE では，初期化後の類似度行列の全要素を一律に定数 λ 減らすだけで，クラスタサイズの均衡性をコントロールできる．今後の課題としては，クラスタサイズの均衡化を制御するパラメータである λ の適切な設定方法を明らかにすることがあげられる．

謝辞 本研究は JSPS 科研費 JP18K11311 の助成を受けたものである．

参考文献

[1] Nguyen, D.T., Chen, L. and Chan, C.K.: Clustering with multiviewpoint-based similarity measure, *IEEE Trans. Knowledge and Data Engineering*, Vol.24, No.6, pp.988–1001 (2012).
[2] Dhillon, I.S. and Modha, D.S.: Concept Decompositions for Large Sparse Text Data Using Clustering, *Mach. Learn.*, Vol.42, No.1-2, pp.143–175 (2001).
[3] Sokal, R.R. and Michener, C.D.: A statistical method for evaluating systematic relationships, *University of Kansas Science Bulletin*, Vol.38, pp.1409–1438 (1958).
[4] Fujiwara, Y. and Koga, H.: Multiviewpoint-Based Agglomerative Hierarchical Clustering, *Database and Expert Systems Applications* (*DEXA*), pp.325–340, Cham, Springer International Publishing (2019).
[5] Shi, J. and Malik, J.: Normalized cuts and image segmentation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.22, No.8, pp.888–905 (2000).
[6] Karypis, G.: CLUTO – A clustering toolkit, Technical Report, Minnesota Univ. Minneapolis Dept. of Computer Science (2002).
[7] Cai, X., Nie, F. and Huang, H.: Multi-View K-Means Clustering on Big Data, *International Joint Conference on Artificial Intelligence*, pp.2598–2604 (2013).
[8] Tao, H., Hou, C., Liu, X., Liu, T., Yi, D. and Zhu, J.: Reliable Multi-View Clustering, *AAAI Conference on Artificial Intelligence*, pp.4123–4130 (2018).
[9] Yan, Y., Chen, L. and Nguyen, D.T.: Semi-supervised clustering with multi-viewpoint based similarity measure, *The 2012 International Joint Conference on Neural Networks* (*IJCNN*), pp.1–8 (2012).
[10] Ravoori, D.T. and Chen, Z.: Multi-View Meets Average Linkage: Exploring the Role of Metadata in Document Clustering, *Int. J. Inf. Retr. Res.*, Vol.5, No.2, pp.26–42 (2015).
[11] D'hondt, J., Vertommen, J., Verhaegen, P.-A., Cattrysse, D. and Duflou, J.R.: Pairwise-adaptive dissimilarity measure for document clustering, *Information Sciences*, Vol.180, No.12, pp.2341–2358 (2010).
[12] Lin, Y., Jiang, J. and Lee, S.: A Similarity Measure for Text Classification and Clustering, *IEEE Trans. Knowledge and Data Engineering*, Vol.26, No.7, pp.1575–1590 (2014).

藤原 勇二

1993 年生．2018 年電気通信大学大学院情報理工学研究科情報・ネットワーク工学専攻博士前期課程修了．修士（工学）．2018 年株式会社科学情報システム入社，現在に至る．

古賀 久志 （正会員）

1971 年生．1995 年東京大学大学院理学系研究科情報科学専攻修了．1995 年富士通研究所入社．博士（理学）．2003 年電気通信大学大学院情報システム学研究科講師．現在，電気通信大学大学院情報理工学研究科准教授．専門は類似検索アルゴリズム．現在はストリームデータ処理の研究に従事．

付　　録

A.1 定理 1 の証明

$\mathrm{MVS}(d_i, d_j | d_i \in S_k, d_j \in S_c) = \frac{1}{n-n_k-n_c} \sum_{d_h \in S \setminus (S_k \cup S_c)} (d_i - d_h)^{\mathrm{T}}(d_j - d_h)$ なので，クラスタ k と c 間の類似度 Sim_{kc} は式 (A.1) となる．

$$Sim_{kc} = \frac{1}{n_k n_c (n - n_k - n_c)} \sum_{d_i \in S_k} \sum_{d_j \in S_c} \sum_{d_h \in S \setminus (S_k \cup S_c)} (d_i - d_h)^{\mathrm{T}}(d_j - d_h). \tag{A.1}$$

ここで $S_c = S_a \cup S_b$ なので，式 (A.1) は

$$\frac{1}{n_k n_c (n - n_k - n_c)} \sum_{d_i \in S_k} \left\{ \sum_{d_j \in S_a} \sum_{d_h \in S \setminus (S_k \cup S_a \cup S_b)} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) + \sum_{d_j \in S_b} \sum_{d_h \in S \setminus (S_k \cup S_a \cup S_b)} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) \right\} \tag{A.2}$$

と書ける．

一方で，Sim_{ka} と Sim_{kb} は定義より次式で表される．

$$Sim_{ka} = \frac{1}{n_k n_a (n - n_k - n_a)} \sum_{d_i \in S_k} \sum_{d_j \in S_a} \sum_{d_h \in S \setminus (S_k \cup S_a)} (d_i - d_h)^{\mathrm{T}}(d_j - d_h). \tag{A.3}$$

$$Sim_{kb} = \frac{1}{n_k n_b (n - n_k - n_b)} \sum_{d_i \in S_k} \sum_{d_j \in S_b} \sum_{d_h \in S \setminus (S_k \cup S_b)} (d_i - d_h)^{\mathrm{T}}(d_j - d_h). \tag{A.4}$$

式 (A.2)，(A.3)，(A.4) より，Sim_{kc} は Sim_{ka} と Sim_{kb} を用いて式 (A.5) のように表現できる．

$$\begin{aligned} Sim_{kc} = {} & \frac{n_a(n - n_k - n_a)}{n_c(n - n_k - n_c)} Sim_{ka} + \frac{n_b(n - n_k - n_b)}{n_c(n - n_k - n_c)} Sim_{kb} \\ & - \frac{1}{n_k n_c (n - n_k - n_c)} \left\{ \sum_{d_i \in S_k} \sum_{d_j \in S_a} \sum_{d_h \in S_b} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) + \sum_{d_i \in S_k} \sum_{d_j \in S_b} \sum_{d_h \in S_a} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) \right\}. \end{aligned} \tag{A.5}$$

ここで，各クラスタの総和ベクトル $D_a = \sum_{d_i \in S_a} d_i$ を用いることで，

$$\sum_{d_i \in S_k} \sum_{d_j \in S_a} \sum_{d_h \in S_b} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) = n_b D_k^{\mathrm{T}} D_a - n_a D_k^{\mathrm{T}} D_b - n_k D_a^{\mathrm{T}} D_b + n_k n_a n_b.$$

であり，

$$\sum_{d_i \in S_k} \sum_{d_j \in S_b} \sum_{d_h \in S_a} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) = n_a D_k^{\mathrm{T}} D_b - n_b D_k^{\mathrm{T}} D_a - n_k D_b^{\mathrm{T}} D_a + n_k n_a n_b.$$

である．よって，

$$\sum_{d_i \in S_k} \sum_{d_j \in S_a} \sum_{d_h \in S_b} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) + \sum_{d_i \in S_k} \sum_{d_j \in S_b} \sum_{d_h \in S_a} (d_i - d_h)^{\mathrm{T}}(d_j - d_h) = -2 n_k D_a^{\mathrm{T}} D_b + 2 n_k n_a n_b. \tag{A.6}$$

式 (A.6) を式 (A.5) に代入すると，

$$Sim_{kc} = \frac{n_a(n - n_k - n_a)}{n_c(n - n_k - n_c)} Sim_{ka} + \frac{n_b(n - n_k - n_b)}{n_c(n - n_k - n_c)} Sim_{kb} + \frac{2(D_a^{\mathrm{T}} D_b - n_a n_b)}{n_c(n - n_k - n_c)}. \tag{A.7}$$

定理 1 は式 (A.7) において，$n_c = n_a + n_b$ と書き換えることで得られる．