1C1-6

ノイズを含む高次元データにおける K-NN ベースクラスタリング手法

K-NN Based Clustering Methods for High-Dimensional Data Sets with Noise

深田　健太[*1]　Michael E.Houle[*2]　鷲尾 隆[*1]
Kenta Fukata　Takashi Washio

[*1]大阪大学産業科学研究所高次推論方式
Department of Advanced Reasoning, Osaka University. The Institute of Scientific and Industrial Research

[*2]国立情報学研究所
National Institute of Informatics

The needs of efficient image picture query and clustering is rapidly increasing under the development of broadband network and multimedia communication. The recent techniques in the image query and clustering extract thousands of points of interest (POI) from each image, and represent the feature of each POI by a Scale Invariant Feature Transform (SIFT) vector. The similarity among images is evaluated by the matching of SIFT vectors and used for the query and clustering. A crucial issue in the similarity evaluation is the efficient matching of the SIFT vectors. In this report, we assessed some approaches based on a k-NN query technique named "SASH" and compared their efficiency and accuracy with the direct matching by using real world image pictures.

1. はじめに

近年，ブロードバンドネットワークとマルチメディアの普及に伴い，画像検索のニーズが高まっている．現状の多くの画像検索手法では，各画像中で興味を持つ点 (Point of Interest: POI))[5] あるいはその集合に着目して画像同士の類似性を判断するという人間の視覚上の特徴を利用する．代表的な方法では，１枚の画像から視点の違いの影響を受け難く形状特徴の明確な画像上の点を，その最適な縮尺率や回転角度の条件と共に POI として数百〜数千個抽出し，複数画像に亘ってそれらの特徴が比較可能なように周囲の各方向に関する画像強度変化率からなる特徴ベクトルを計算する．このような特徴ベクトルを Scale Invariant Feature Transform (SIFT) ベクトルと呼び，典型的には各ベクトルが 128 次元要素からなる．つまり，画像検索の前処理として，１枚の画像をこのような数百〜数千の SIFT ベクトルの集合に変換する．

２つの画像をマッチングするためには２つの SIFT ベクトル集合同士のマッチングを行う必要があるため，一方の集合内の各 SIFT ベクトルともう一方の集合内の最近接 SIFT ベクトルまでの距離を計算し，それらの集合間の平均距離に基づいて画像検索を行う．現在使用される画像検索手法として，k-d tree[6] とそれを発展させた Best-Bin-First(BBF) アルゴリズム [7] が存在する．k-d tree は探索空間の分割を行うことにより，query を行う際に，その query の存在する空間と隣り合う空間のみを探索し，計算量を減らす手法である．しかし，探索空間が１０次元を超えると，計算量が増大し，完全探索より速度が劣る欠点がある．BBF アルゴリズムは query からより近い隣りの探索空間を優先的に探索し，効率的に近い事象を見つけ出す近似探索手法である．これは 1000 枚程度の画像数であれば効率的な検索を行うことが可能であるが，より多くの画像のより高次元な SIFT ベクトルに対して検索を行うと，計算量の急激な増大や似た距離に存在する SIFT ベクトル数の増加から，正確かつ高速な検索を実行することが困難になる．このことから，現状の SIFT ベクトルのマッチングに基づく大量画像高速検索アルゴリズム研究は不十分であると考えられる．

SIFT ベクトルマッチング問題は大規模次元大量ベクトルデータに関する類似ベクトル検索問題である．直接検索を用いると，画像の SIFT ベクトル本数を平均 n 本とすると，$o(n^2)$ の計算時間複雑さとなる．この計算時間を短縮する方法として，ピボットを利用した Vantage Point Tree[1][2] やボロノイ図を利用した Voronoi Tree[1][3] が代表的である．Vantage Point Tree はピボットと呼ばれる複数の基準ベクトルから他のベクトルまでの距離が，ある一定値よりも近いか遠いかで２分木を作成し，それを利用して必要な空間のみを探索する手法であり，Voronoi Tree は幾つかの基準ベクトルを結ぶ直線の垂直２等分線の作成を階層的に繰り返すことで空間を区切り，必要な空間のみを探索する手法である．それぞれ計算時間複雑さは検索データ作成が $o(n \log n)$，検索時 $o(\log n)$，記憶容量複雑さは $O(n)$ である．これに対して近年 SASH 法 [4] と呼ばれる新しい検索手法が提案されており，その計算時間複雑さは検索データ作成が $o(n \log n)$，検索時 $o(\log n)$，記憶容量複雑さは $O(n)$ と前記手法と同様である．しかし，実際の計算速度や検索精度は SASH 法が非常に優れていることが明らかになっている．この SASH 法については３章にて詳細を述べる．本研究では，画像を対象としたノイズを含む高次元データの k-NN ベースクラスタリング手法開発の準備として，SASH 法を用いた k-NN 類似 SIFT ベクトル検索に基づいた高速画像検索方法を検討する．その中でも特に，SASH 法で類似画像を検索するための画像間距離評価方法を検討することを目的とし，その詳細を４章に，またそれによる実験結果を５章に述べる．

連絡先: 大阪大学産業科学研究所
〒 567–0047 大阪府茨木市美穂ヶ丘 8-1
E-mail: kenta-f@ar.sanken.osaka-u.ac.jp

2. 従来の画像検索手法

本節では従来の画像検索手法の概要について述べる．

2.1 SIFT ベクトル抽出

以下の 4STEP を行うことで，画像データ内で縮尺や回転などに対して不変な場所を特定し，その場所の本質的特徴を表す SIFT ベクトルを作成する．

1. 縮尺率及び画像位置上の極値の発見

   はじめに，適切な粒度の画像上の縮尺率や位置の様々な組み合わせについて，ガウス関数差分を適用し，縮尺率や位置の差異に対して不変である潜在的に興味深い画像上の極値点を効率的に特定する．

2. POI の特定

   それぞれの極値点に対して，内挿補間によってより詳細な縮尺率と位置を推定する．そして，更に，画像が表す物体形状の特徴が明確でない低コントラストな極値点や濃淡境界線上の極値点を除いて，明確な対象形状を表す Point Of Interest (POI) を決定する．

3. 方向の割り当て

   画像各所が歪みなどによって回転している可能性を考慮し，画像強度が最も変化する方向を求める．常にこの方向に次ステップで求める POI の特徴ベクトルの座標系を統一することで，POI 付近の画像回転歪の影響を受け難くする．

4. POI の SIFT ベクトル

   各 POI について上記方向に合わせた座標系で周囲 4×4 の格子点上のそれぞれ 8 方向に関する画像強度の変化率から成る，128 次元 SIFT ベクトルを求める．

この処理を行うことにより，画像から位置や縮尺などに対して不変的で形状が明確な場所を見つけ出し，更にその場所の回転を補正した SIFT ベクトルを特徴ベクトルとして得る．一般に，SIFT ベクトルは各画像から数百〜数千本取り出される．

2.2 画像検索アルゴリズム

画像検索を行う際，はじめに検索対象とする各画像から SIFT ベクトルを抽出し，データベースに貯蔵する．新しいイメージを検索する際にはそのイメージから SIFT ベクトルを抽出し，画像間の SIFT ベクトル群のユークリッド距離に基づいて検索を行う．その代表的な方法に，以下の 2 つがある．

1. Keypoint matching

   対象画像から作られた SIFT ベクトル群の中から query 画像の SIFT ベクトルと最も近い SIFT ベクトルを持つ画像を特定することで検索を行う．ここでは，SIFT ベクトル間のユークリッド距離が最も小さいものを最も近い画像であると判定する．ユークリッド距離は直接計算法を用いて計算する．

2. Efficient nearest neighbor indexing

   画像と画像をマッチングするためには 2 つの SIFT ベクトル集合同士のマッチングを行う必要があるため，一方の集合内の各 SIFT ベクトルともう一方の集合内の最近接 SIFT ベクトルまでの距離を計算し，それらの集合間の平均距離に基づいて画像検索を行う．128 次元のような，高次元データにおける 2 つのベクトルの正確な nearest neighbor を測るアルゴリズムとしては，直接計算法より高速なアルゴリズムは実用的に存在しない．そこで，高速な近似検索アルゴリズムである Best-Bin-First(BBF)[7] が用いられることが多い．BBF は探索空間の分割を行い，隣り合う空間の中で検索画像に近い空間を優先して探索することで近似的に類似の画像を見つけ出す手法である．

3. SASH 法

SASH 法 [4] はランダムサンプリングを用いて SASH 構造を構築し，高速で近似的に高精度な k-nearest neighbor (k-NN) 類似事例検索を行う手法である．

3.1 SASH 構築

SASH 法は階層状にランダムにサンプリングされたノードを展開し，それと同時にトップノードから順に下のノードへリンクを張る処理を繰り返すことで，図 1 に示すような SASH 構造を構築する．ただし，root ノードから辿れない孤児ノードを作らないために，root ノードを除いたノードは必ず親ノードを持つように構成する．SASH の各レベルのノード数はトップの 1 個から始めてレベルを降りる毎に 2 倍となり，最下レベルでは [n/2] 個となる．

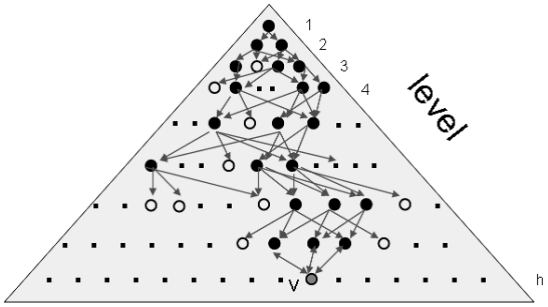

図 1: SASH 構造

以下に SASH 構造を構築するアルゴリズムを述べる．SASH は $SASH_1$(レベル 1），$SASH_2$（レベル 2），$\cdots$ の順に再帰的に構築される．ここでは，$SASH_{l-1}$ が与えられたとき，$SASH_l$ を構築する方法を示す．

1. l=2 のときは root ノードが親となり，root ノードはレベル 2 の全てのノードを子供に持つ．$l \neq 2$ のとき，以下のステップを全て行う．

2. レベル l のノード v 各々について，距離の評価指標を用いて，レベル 1 からレベル l-1 まで，各レベルにおいて 1 つ上位レベルで選択されたノードの子ノードの中から v に近い順に p 個のノードを選択する，ということを繰り返す．そして，レベル l-1 で選択された p 個のノードを v の仮の親とする．本研究では距離の指標として，ユークリッド距離を用いる．v の上述の l-1 レベルの仮の親にリンクを張る．

3. レベル l-1 の各ノードからリンクが張られたレベル l のノードにエッジを展開する．ただし，距離の指標を用いて，レベル l-1 の各ノードから近い順にレベル l の c 個のノードまでエッジを展開する．

4. レベル l の全てのノードが親を持てば次のレベルの SASH 構築を行う．親を持たないノードが存在したとき，そのノードが持てる親の数 p を倍にして親を探索する．全ての子供が親を持つと，次のレベルの SASH 構築を行う．

事例数が n のとき，SASH 構築の計算時間複雑さは $o(nlog_2 n)$ に抑えられる．また，本研究においては下位ノード 1 個当たりの親数を p=4 とし，親が持つことができる子供の最高数を c=16 とする．

3.2 SASH による高速近似 k-NN Query

ある query(q) が与えられると，SASH 構築時と同様にして，各レベル毎に事例 q に近い k 個の事例を探索し，$P_l(q,k_l)$(l=1,..,h) を作成する．ここで，h は SASH のレベルの最大数である．そして，$P_l(q,k_l)$(l=1,..,h) の中から q に近い k 個の事例を解として抽出する．このように SASH 法を用いて近似類似事例を探索する方法 [4] の計算時間複雑さは $o(log_2 n)$ となる．

4. 画像間距離計算法

本研究では，画像間の距離を計算する手法として以下の 4 つを考案した．

4.1 SIFT ベクトル直接法

各画像 i(i=1,..,h) は m_i 本の SIFT ベクトル s_{ig}(g=1,..,m_i) で表される．ここで，h は画像の枚数，m_i は各画像毎の SIFT ベクトル数を表す．データ画像 i と query 画像 q 間の距離を測るために，まず，画像 i の各 SIFT ベクトルと画像 q の全 SIFT ベクトルのユークリッド距離を計算する．画像 i の各 SIFT ベクトルに対して，これらのユークリッド距離の最小値を求め，それの平均を取ることで画像 i から画像 q までの距離 d_{iq} とする．同様に画像 q の各 SIFT ベクトルから画像 i の全 SIFT ベクトルへのユークリッド距離を計算し，その最小値の平均を取ったものを画像 q から画像 i までの距離 d_{qi} とする．そして，画像 i と画像 q の距離をその幾何平均を取った $D_{iq} = \sqrt{d_{iq} * d_{qi}}$ とする．このように画像距離を求める際，SIFT ベクトル同士のユークリッド距離を全計算する方法が SIFT ベクトル直接法である．

4.2 SIFT ベクトル SASH 法

SIFT ベクトル SASH 法は対象画像 i の各 SIFT ベクトルと query 画像 q の全 SIFT ベクトルのユークリッド距離を計算する際に，SASH を用いて，SIFT ベクトルの k-NN フィルタリングを行うことで計算量を減らす手法である．初めに，各画像毎に SASH を構築する．次に画像 i の全 SIFT ベクトル s_{ig} 毎に，画像 q の SASH に対して，query することで k-NN SIFT ベクトルの集合を得る．同様に，画像 q の全 SIFT ベクトルから画像 i の k-NN SIFT ベクトル集合も得る．これらを元に 4.1 節と同様にして画像距離 D_{iq} を求める．本研究では SASH に対して query を行う際の k-NNquery を k=1 とする．

4.3 SIFT・代表ベクトル法

予め前処理において，各画像について SASH を用いて相互にあまり似てない SIFT ベクトルを選び，それらをその画像の代表ベクトルとする．そして，実際に画像 q で query する際に，前処理で求めておいた各画像の代表ベクトル集合と query 画像 q の全 SIFT ベクトル集合の間で，直接法と同様にベクトル間最小距離の平均距離を求める手法が SIFT・代表ベクトル法である．画像 i の代表ベクトルを求める前処理では，画像 i の SIFT ベクトル集合について SASH を構築する．そして，画像 i の全 SIFT ベクトル集合から，ランダムにある SIFT ベクトル s_{ig} を選び，画像 i の SASH の k-NN query を行う．これにより，画像 i 内で s_{ig} と似た SIFT ベクトルを k 本抽出する．この k 本の SIFT ベクトルの代表を query に用いた s_{ig} とする．次に今求められた k 本の SIFT ベクトルを除いた画像 i の SIFT ベクトルの集合から，再び代表ベクトルを選択する．先行する全ての代表ベクトルに類似する SIFT ベクトルを除いて，以後，代表ベクトルを選ぶことを繰り返して m_i/k 本の代表ベクトルを抽出する．この代表ベクトルと query 画像 q の全 SIFT ベクトルで直接法同様の距離の計算を行う．本研究では k=2 及び 5 の場合に実験評価を行った．

4.4 代表ベクトル法

4.3 節で求めた方法で画像 i，query 画像 q の両方から代表ベクトルを抽出する．この代表ベクトル同士で直接法による距離計算を行う方法が代表ベクトル法である．本研究ではここでも k=2 及び 5 の場合に実験評価を行った．

5. 実験結果

本研究では自転車や猫といった人間の目から見て特徴的な対象が含まれる 8 枚の写真画像を用いた．それらの画像をそれぞれ SIFT ベクトルに変換し，４章の様々な方法で画像距離を計算し，その時間や精度の比較を行った．それぞれの画像に対する SIFT ベクトル数は画像１:2483，画像２:4555，画像３:606，画像４:550，画像 5:3613，画像 6:659，画像 7:3066，画像 8:1350 であった．本稿では画像１と画像２～８の間で各手法によって距離計算を行った時間や精度の比較結果を示す．

5.1 画像検索時間

今回の基準とする画像 1 はすでに SASH 構築や代表ベクトル化がなされており，新しく画像 2～8 を検索することを想定した実験を行った．初めに画像 2～8 を各々SIFT ベクトルに変換する．その後，手法毎に距離計算に要した時間の結果が表１である．ただし，単位は全て秒である．

表 1: 手法の検索時間比較

	画像 2	画像 3	画像 4	画像 5	画像 6	画像 7	画像 8
手法 1	11	1.6	1.5	8.8	1.7	7.5	3.5
手法 2	20.2	5.8	5.4	17.2	6.1	15.4	8.8
手法 3	5.2	0.86	0.78	4.6	0.98	4.0	1.6
手法 3’	2.2	0.53	0.52	1.7	0.55	1.5	0.88
手法 4	11.3	0.84	0.72	8.2	0.94	6.9	2.3
手法 4’	5.5	0.39	0.34	3.9	0.42	3.3	1.1

ここで手法 1 は SIFT ベクトル直接法，手法 2 は SIFT ベクトル SASH 法，手法 3 は SIFT・代表ベクトル法 (k=2)，手法 3’ は SIFT・代表ベクトル法 (k=5)，手法 4 は代表ベクトル法 (k=2)，手法 4’ は代表ベクトル法 (k=5) である．例えば右下の 1.1 という値は手法 4’ を用いると，画像 1 と画像 8 の計算時間は 1.1 秒であったことを示す．これらの値は新しいある query 画像 1 枚に対して，似た画像 1 枚を検索する際に必要となる計算時間である．

5.2 画像クラスタリング計算時間

画像 1～8 が対象画像として蓄積されており，SASH 構築や代表ベクトル化がなされていると考え，その上でクラスタリングを行う際，手法毎に要する計算時間の結果が表２である．ただし，単位は全て秒である．

表 2: 手法のクラスタリング時間比較

	画像 2	画像 3	画像 4	画像 5	画像 6	画像 7	画像 8
手法 1	11	1.6	1.5	8.8	1.7	7.5	3.5
手法 2	15.9	4.1	4	14.1	4.6	12.7	7.1
手法 3	5.2	0.86	0.78	4.6	0.98	4.0	1.6
手法 3’	2.2	0.53	0.52	1.7	0.55	1.5	0.88
手法 4	2.4	0.28	0.25	1.9	0.3	1.6	0.61
手法 4’	0.34	0.031	0.032	0.23	0.047	0.18	0.078

5.3 画像距離精度

４章の４つの手法を用いて各画像間の画像距離を計算し，その距離に基づいて各画像同士の近さ順位付けを行った．SIFT ベクトル直接法によって導かれた距離が正しい画像距離順位であると考え，その順位とその他の手法により導かれた画像同士

の近さ順位の差を調べるため，以下のスピアマンの順位相関係数 ρ を用いた．

$$\rho = 1 - \frac{6\sum D^2}{N(N^2-1)}$$

ここで，D は 2 つの順位間で同じ画像の順位の差，N は順位付け対象の画像数である．また画像 8 枚中で 1 枚は基準に用いたため，N=7 である．各画像 1 ～ 8 を各々基準に用いた場合について，SIFT ベクトル直接法と他の手法の順位相関係数 ρ を求め，それを全基準画像について平均を計算したものが各手法の精度と考えられる．SIFT ベクトル直接法と他の手法の順位相関値を表 3 にまとめる．

表 3: 手法の順位相関値

	手法 2	手法 3	手法 3’	手法 4	手法 4’
手法 1	1	0.977	0.9018	0.7946	0.7723

6. 結果考察

手法 1 と手法 2 を比較すると，検索とクラスタリングの両方において，SASH を用いた手法 2 が手法 1 より計算時間を要する結果となった．これは SASH 自体の構築に計算時間を要することと，query を n 回行う際に SASH を用いることでプログラムの複雑な処理に計算時間を必要としたためと考えられる．つまり，SIFT ベクトルが 128 次元とあまり高次元ではないため，SASH の利点を十分活かせなかったと考えられる．しかし，順位相関値が 1 であることから，SASH の精度が非常に高いことを確認できた．

次に手法 1 と手法 3，手法 3’ を比較する．基準とした画像 1 はすでに SASH 構築と代表ベクトル化が済んでいる．そのため，検索を行うために必要な処理は，画像 2～8 の SIFT ベクトルと画像 1 の代表ベクトルの距離計算のみである．このことから，直接法と比べて計算回数を 1/k 倍にすることができ，計算時間も約 1/k 倍とすることが可能である．この手法では，前処理は画像を SIFT ベクトルに変換するのみであるため，画像検索と画像クラスタリングに要する計算時間は等しくなる．このように前処理をほとんど必要としないため，この手法は検索に向いている．ただし，k の値を大きくすると順位相関値が下がる．つまり，精度が劣化するため，k の値をどのように決めるかが重要となる．

最後に手法 1 と手法 4，手法 4’ を比較する．画像検索を行う際の計算時間を比較すると，距離計算を行う画像 2～8 の SIFT ベクトル数により違いのある結果となった．検索を行う際は画像 2～8 の SIFT ベクトルを代表ベクトル化する処理に計算時間を必要とする．代表ベクトルを作成するためには画像 2～8 の SIFT ベクトル自体を SASH 化し，反復的に query を行う必要がある．上述したように SASH の構築と SASH に対して query を行う際に計算時間を要するため，画像 2～8 の中で SIFT ベクトル数が少ないものは短い時間で代表ベクトル化できるのに対し，SIFT ベクトル数が多いものは長い計算時間を必要とする．そのため，SIFT ベクトル数が多い画像は直接法より時間を要する結果になった．しかし，k=5 を用いた手法 4’ においては SIFT ベクトル数が多い画像においても，SASH に対して query を行う回数を減らすことができるため，高速化できた．また，画像のクラスタリングを行う際は事前に代表ベクトル化を行うことで，計算回数を減少できる利点のみを利用するため，非常に高速化できる．ただし，検索精度は劣化する．

7. おわりに

本研究において，2 画像間の距離を測る様々な方法を提案し，精度を維持しつつ，より効率的な画像検索やクラスタリングを行う手法を検討した．課題として，より効果的に SASH を用いることにより，計算時間と精度の向上を図る必要がある．また，本研究では画像 2 枚の距離計算を行う方法を提案したが，大量の画像データが存在するときに，効率的な画像検索，画像クラスタリングを行うことができる手法を考案する必要がある．

参考文献

[1] Edgar Chavez, Gonzalo Navarro, Ricardo Baeza-Yates, Jose Luis Marroquin, ACM Computing Surveys (CSUR), Volume 33 , Issue 3, pp.273-321(2001)

[2] P.Yianilos, Excluded middle vantage point forests for nearest neighbor search in general metric spaces. In proc. 4th ACM-SIAM Symposium on Discrete Algorithms (SODA’93), pp.311-321(1993)

[3] F.Dehen, H.Nolteimer, Voronoi trees and clustering problems. Information Systems, 12(2): pp171-175(1987)

[4] Michael E.Houle, Jun Sakuma, Fast Approximate Similarity Search in Extremely High-Dimensional Data Sets, Proceedings of the 21st International Conference on Data Engineering (ICDE’05), pp619-630(2005)

[5] David G.Lowe, Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, 60, 2, pp.91-110(2004)

[6] Friedman.J.H., Benteley.J.L, Finkel.R.A, An algorithm for finding best matches in logarithmic expected time, ACM Transactions on Mathematical Software, 3(3), pp.209-226(1977)

[7] Beis.J, Lowe.D.G, Shape indexing using approximate nearest-neighbour search in high dementional spaces, In Conference on Computer Vision and Pattern Recognition, pp1000-1006(1997)