F-015

多クラスに対するピボットを用いた大規模データの k-means クラスタリング
Pivot-based k-Means Clustering with Large k for Large-scale Data Sets

服部 正嗣 † 青山 一生 †
Takashi Hattori Kazuo Aoyama

1. まえがき

典型的な k-means クラスタリング法である Lloyd 法は，データマイニングにおける主要なアルゴリズムの一つである [8]．また，Lloyd 法を大規模データに適用するために必要な高速化法も開発されている [4, 6]．大規模データは多数のクラスタから形成されることがあり，その場合，Lloyd 法のパラメータであるクラスタ数（k）は大きく設定すべきである．このような多クラス分割問題の場合，既存高速化法 [4, 6, 5] は時間計算量と空間計算量がともに増大するという問題を生じる．例えば，Elkan 法 [4] は，オブジェクト数（N）とクラスタの重心数（$k < N$）とに対して距離格納用に $O(N{\cdot}k)$ の記憶容量を必要とするため，汎用計算機において Elkan 法を用いて大規模データを多クラスに分割することは困難である．

我々は大規模データの多クラス分割問題を効率的に解くことを可能にする，Lloyd 法の高速化法を提案する．提案法は，新たに導入された少数（$m \ll k$）のピボット p，クラスタ重心 c，オブジェクト x の 3 点に距離の三角不等式を適用することにより，オブジェクトとクラスタ重心との距離計算回数を削減する，提案法が Lloyd 法に対する高速性と小記憶容量 $O((N+k)m)$ とを両立することを，人工データと実画像データとを用いて実験的に示す．

2. 距離計算省略に基づく高速化法

Lloyd 法と同じ解を与える高速化法のうち，特にオブジェクトとクラスタ重心との距離計算の一部を省略することにより高速化を図る三つの手法を概観する．同時に，大規模データを多クラスに分割する場合に起きる，これらの手法の問題点も指摘する．

第一の手法は，与えられたデータから階層的クラスタ（アンカー）をノードとする木構造を構築し，索引として利用することにより，距離計算の一部を省略する Moore 法 [6] である．X_p をアンカー中心 n_p に所属するデータ点集合，$r_{max}(X_p)$ を X_p のうちで n_p から最遠のデータ点までの距離とすると，重心 c とデータ点 $x \in X_p$ の距離の下限値 $d_{LB}(c,x)$ は $d(c,n_p) - r_{max}(X_p)$ で表される．一方，アンカーの中心 n_p に最も近い Lloyd 法のクラスタ重心 $c*$ と，データ点 x の距離の上限値 $d_{UB}(c*,x)$ は $d(c*,n_p)+r_{max}(X_p)$ で算出できる．仮に $d_{UB}(c*,x) > d_{LB}(c,x)$ であるならば，$\forall x \in X_p$ は c を重心とするクラスタに所属することはない．よって $d(c,x)$ の計算を省略できる．このように Moore 法は，アンカー中心とアンカー半径という索引の情報を利用して距離計算回数を削減する反面，データが大規模であるほど木構造である索引を構築するのに要する計算時間が増大するという問題がある．

第二の手法は，Elkan 法は，データ点に最も近いクラスタ重心を決定する際，データ点と重心との距離計算を行う前に，その距離の下限値を用いてデータ点から見た二つの重心の遠近の判定を行うことで不要な距離計算を省略する手法である [4]．データ点 x と重心 c との距離の下限値 $d_{LB}(x,c)$ は，重心座標更新毎に，更新前の距離の下限値から更新前後の重心の移動距離を減算して算出される．データ点 x が所属している中心 $c(x)$ とデータ点 x との距離 $d(x,c(x))$ が $d(x,c(x)) < d_{LB}(x,c)$ を満たせば，データ点 x の所属の変更は起こらないことが分かり，$d(x,c)$ の計算が省略できる．また，省略がなされず，$d(x,c)$ の計算が実行された場合には，距離の下限 $d_{LB}(x,c)$ は $d(x,c)$ に置換される．

重心座標更新毎に再計算が必要となる重心移動距離の計算回数 k は，省略される可能性のある距離計算の回数 $N \times (k-1)$ よりも十分小さいため，距離の下限値を用いた遠近の判定により距離計算の一部が省略されると高速化につながる．その反面，Elkan 法は距離の下限値格納用に $O(N{\cdot}k)$ の記憶容量を必要とするため，N, k が共に大きな大規模データを多クラスに分割する用途には適さない．

第三の方法は，Elkan 法と比較して記憶容量を抑制することができる Hamerly 法である [5]．Hamerly 法は，データ点とそのデータ点が所属していないクラスタ重心との距離の下限値を求める際に，データ点から第二近傍の重心との距離を初期値として，以降第一近傍の重心を除く $k-1$ 個の重心のうち，重心更新前後で最も大きく移動した重心の移動距離を減算する方法で算出する．この算出法では，k の増大に伴って距離の下限値がより緩くなるため，多クラス分割問題には適さない．

3. 提案法

提案法は，各オブジェクト（データ点 x とも呼ぶ）が所属するクラスタを決定するために必要な x とクラスタ重心 c との距離 $d(x,c)$ の計算の要否を，その下限値 $d_{LB}(x,c)$ を用いて判定する．効率的な下限値計算のために，ユークリッド空間の m 個の点をピボット p として導入し，距離 $d(p,x)$ と距離 $d(p,c)$ とから三角不等式を用いて下限値 $d_{LB}(x,c)$ を算出することにより，不要な距離計算を省略する．

3.1 距離計算の省略法

クラスタリングのある段階において，データ点 x が重心 $c(x)$ に所属していたとする．重心の座標の更新に伴って x の所属する重心が別の重心 c に変更されるかを判定するためには，$d(x,c(x)), d(x,c)$ の大小の比較が必要である．

本手法では，$d(x,c)$ の距離計算を行う前に，ピボット

†日本電信電話株式会社 NTT コミュニケーション科学基礎研究所

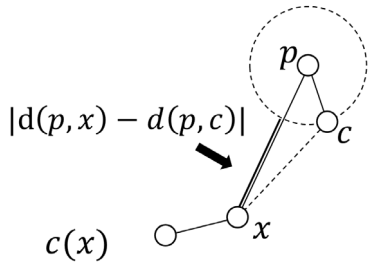

図 1: 距離 $d(x,c)$ の下限値 $d_{LB}(x,c) = |d(p,x) - d(p,c)|$. $d_{LB}(x,c) > d(x,c(x))$ ならば $d(x,c)$ の計算は不要である．

p を用いて算出した $d(x,c)$ の距離の下限値 $d_{LB}(x,c) = |d(p,x) - d(p,c)|$ と $d(x,c(x))$ との大小比較を行う．図 1 に示す通り，この際，$d(x,c(x)) < d_{LB}(x,c)$ であれば，$d(x,c)$ を計算することなく所属の変更が起こらないことが分かる．

3.2 ピボット選出法

上述の方法で省略できる距離計算の省略回数を最大化するようなピボット選出法が望まれる．例えばピボット p をデータ点 x，重心 c のいずれか一方の近くに配置し，距離の下限値 $d_{LB}(x,c)$ を大きくする方法が考えられる．

提案法は以下に示す方針に基づいてピボットを選出する．データ点と重心の距離計算の省略は，複数のピボットのうちいずれかにより達成する．ピボットあたりの距離計算の省略の効果を大きくするために，ピボット同士は互いの近くに配置しない．順次にピボットを選出する場合，既存のピボットにより省略できていないデータ点と重心との距離計算を省略できるようにピボットを配置する．

提案法は，Lloyd 法と同様の手順を 1 反復のみ行い，得られた重心のうち，所属するデータ点の数が最大の重心を第一のピボット p_1 として採用する．2 個目以降のピボット $p_j (j = 2, 3, \cdots, m)$ として $argmax_{c \in C \setminus P}(distance(c) \cdot count(c))$ となる重心を再帰的に選出する．ただし，P, C をそれぞれ既存ピボット，重心の集合としたとき $distance(c)$ は重心 c と最も近い既存のピボットとの距離を表し，$count(c)$ は重心 c について P で距離計算が省略できていない回数を表す．$count(c)$ の算出を **Algorithm1** に示す．ただし，$X(c)$ は重心 c に所属するデータ点の集合である．

3.3 必要記憶容量

提案法と有力な既存高速化法である Elkan 法とがそれぞれ必要とする記憶容量を比較し，クラスタ数が大きいほど提案法が効率的であることを示す．

提案法と Elkan 法がそれぞれ必要とする記憶容量とその内訳を表 1，2 に示す．ただし，D はデータ点集合のユークリッド空間上の次元数を表す．

$D \ll k$ の場合，Elkan 法において最も記憶容量を必要とするのは重心とデータ点の距離の下限値格納用の記憶領域でその容量は $O(N \cdot k)$ である．それに対して，提案法は重心とデータ点の距離の下限値をピボットを経由して算出する．そのため，$O(N \cdot k)$ の代わりにピボットと重心との距離およびピボットとデータ点との距離格納用にそれぞれ $O(m \cdot k)$，$O(m \cdot N)$ を合わせぬ $O((N+k)m)$ を必要とする．これは，$m \ll k < N$ であるため，$O(N \cdot k)$ と比較して小記憶容量となる．

Algorithm 1 $count(c)$ の算出

```
initiate count(c) = 0 for all c ∈ C
for c ∈ C do
  for x ∈ X(c) do
    for c' ∈ C \ {c} do
      flag = 0
      for p ∈ P do
        if d(x,c) <| d(p,c') − d(p,x) | then
          flag = 1
          break
        end if
      end for
      if flag = 0 then
        count(c) = count(c) + 1
        count(c') = count(c') + 1
      end if
    end for
  end for
end for
```

表 1: 提案法の必要記憶容量

記憶内容	データ型	記憶容量
データ点座標	double	$N \times D$
データ点の所属重心	int	N
重心座標	double	$k \times D$
重心に所属するデータ点数	int	k
ピボット座標	double	$m \times D$
ピボットと重心との距離	double	$m \times k$
ピボットとデータ点との距離	double	$m \times N$

4. 実験

提案法を人工データと実画像データに適用し，Lloyd 法と比較して省略できる距離計算の割合を示す．

4.1 データセット

表 3 に使用した人工データおよび実画像データを示す．人工データとして，8，16，32 次元の超球面に一様ランダムにそれぞれ 10^6 個のデータ点を配置して作成した 3 つのデータ点集合 rand8，rand16，rand32 を用いた．実画像データとして，Amsterdam Library of Object Images (ALOI)[1] の画像 24000 枚および画像 SNS サービス flickr[2] より収集した画像 100 枚を基に，それぞれの画像の SIFT 特徴量の特徴点（128 次元）をデータ点として作成した二つのデータ点集合 ALOI と flickr を用いた．

表 2: Elkan 法の必要記憶容量

記憶内容	データ型	記憶容量
データ点座標	double	$N \times D$
データ点の所属クラスタ	int	N
更新前重心座標	double	$k \times D$
更新後重心座標	double	$k \times D$
データ点の所属する重心	int	N
重心とその最近傍重心との距離	double	k
更新前後の重心移動距離	double	k
重心とデータ点との距離の上限値	double	N
重心とデータ点との距離の下限値	double	$N \times k$
所属重心の変更判定フラグ	char	N

表 3: データセット

種別	データ名	次元数	データ点数
人工データ	rand8	8	1000000
	rand16	16	1000000
	rand32	32	1000000
画像データ	ALOI	128	7674723
	flickr	128	225776

4.2 距離計算の省略率

上述の人工データおよび実画像データに対して分割数 k を 1000 および 2000 で提案法を適用した場合に，Lloyd 法と比較して省略された距離計算の割合をそれぞれ表 4,5,6,7 に示す．ピボットの選択方法の評価のため，提案法でピボットを選択した場合に加えて，ピボットを k means++[3] の初期点選択同様の方法で選択した場合（kmpp），Lloyd 法と同様の手順を 1 反復のみ行い得られたクラスタの最大クラスタより降順に所属データ点数の多いクラスタの重心をピボットとして採用した場合（size）を併記する．

表 4: 距離計算の省略率（人工データ，k=1000）

	rand8	rand16	rand32
提案法 (m=10)	0.965150	0.227216	0.002373
提案法 (m=20)	0.986116	0.342502	0.006568
kmpp(m=10)	0.971364	0.275838	0.004431
kmpp(m=20)	0.988465	0.436921	0.009481
size(m=10)	0.967178	0.227216	0.002365
size(m=20)	0.985814	0.348131	0.005116

4.3 議論

図 2 は，各データセットからランダムにデータ点を二つ選び，その距離を記録する試行を 10^6 回行ったとき，横軸の距離に存在したデータ点の組数を縦軸に描画した両対数グラフである．人工データについては，ユークリッド空間における次元数が増加するにつれて，大多数のデータ点の組み合わせについてデータ点間の距離が狭い値域に存在するようになること，すなわち任意の二組のデータ点の距離の差が小さな値をとることが分かる．この場合，ピボットをユークリッド空間中のどこに選択

表 5: 距離計算の省略率（人工データ，k=2000）

	rand8	rand16	rand32
提案法 (m=10)	0.980361	0.252404	0.003030
提案法 (m=20)	0.992483	0.382263	0.007453
kmpp(m=10)	0.976924	0.324889	0.005206
kmpp(m=20)	0.992405	0.488326	0.012401
size(m=10)	0.979945	0.244210	0.003005
size(m=20)	0.992006	0.382588	0.007109

表 6: 距離計算の省略率（実画像データ，k=1000）

	ALOI	flickr
提案法 (m=10)	0.384577	0.302678
提案法 (m=20)	0.467527	0.367825
kmpp(m=10)	0.278002	0.186614
kmpp(m=20)	0.352539	0.248114
size(m=10)	0.390338	0.282291
size(m=20)	0.474255	0.364270

しても，ピボット p を用いて算出するデータ点 x と重心 c との距離の下限値 $|\, d(p,x) - d(p,c) \,|$ は小さな値となり，データ点 x と重心 c との距離 $d(x,c)$ の差異が大きいため，距離計算の省略が行われなくなる．実際，表 4, 5 に示す通り，人工データについては，いずれのピボット選択方法を用いても次元数が上昇するにつれて距離計算の省略率は低下する．特に rand32 については，ほとんど距離計算の省略はなされていない．

一方で，2 種の実画像データのユークリッド空間上における次元数は 128 であり，rand32 と比較してさらに次元数が高くなっているにも関わらず，表 6, 7 に示す通り一定の距離計算を省略できている．これは，データ点間距離の分布のピークは人工データのそれと比較して遠距離にある一方で，近距離に存在するデータ点の組数が人工データに比較して多いためと考えられる．この場合，ピボットを適切に選択することで重心 c とデータ点 x の一方に近く，他方に遠く配置することができる．このとき，距離の下限値 $|\, d(p,x) - d(p,c) \,|$ と距離 $d(x,c)$ との差異は小さくなり，結果として，距離計算を省略できる可能性が高まる．

以上のように，提案法（およびピボット選択に kmpp, size を用いた方法）は対象データセットのユークリッド空間における次元数が低いか，高い次元数を持っていても，ユークリッド空間中におけるデータ点の分布が一様

表 7: 距離計算の省略率（実画像データ，k=2000）

	ALOI	flickr
提案法 (m=10)	0.412797	0.309934
提案法 (m=20)	0.494782	0.395635
kmpp(m=10)	0.313532	0.185929
kmpp(m=20)	0.357798	0.251454
size(m=10)	0.415503	0.309607
size(m=20)	0.493651	0.390160

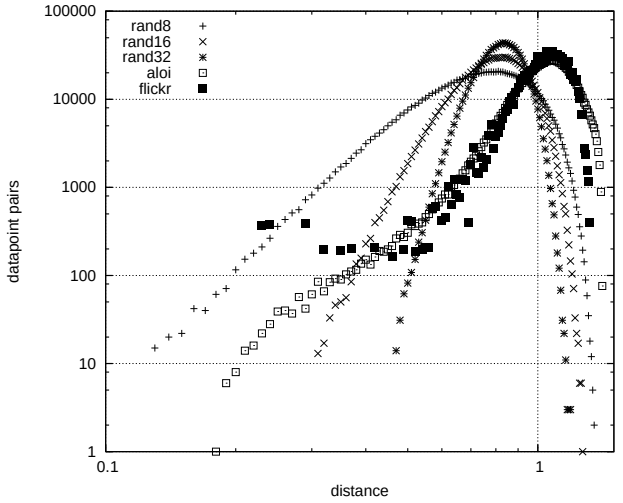

図 2: データ点間の距離分布

でなければ高速化が行えるといえる．

分割数 k については，いずれのピボット選択法を用いても 1000 の場合より 2000 の場合の方が省略率が上昇している．これは，分割数 k が大きいほどデータ点が所属する重心とデータ点との距離 $d(x, c(x))$ が小さくなり，距離の三角不等式 $d(x, c(x)) <| d(p, x) - d(p, c) |$ が成立しやすくなるためと考えられる．

ピボット数 m については，各ピボット選択法ともに 10 よりも 20 の場合の方が省略率は上昇している．重心とデータ点の距離の下限について，ピボットを利用する方法はピボット数 m，データ数 N，重心数 k に対して $m \times (N + k)$ 個の距離を記憶する記憶領域を必要とする．高速化のためには運用する計算機の記憶容量に応じてピボット数 m を選択するべきである．最もデータ点数が多い ALOI についてピボット数 m，分割数 k をそれぞれ 20，2000 としたとき，提案法は 11GB の記憶容量を必要とした．なお，Elkan 法を分割数 $k = 2000$ で ALOI に適用した場合，重心とデータ点との距離の下限値の記憶のためだけに 121.6GB の記憶容量を必要とする．

ピボット選択方法について，提案法と比べ kmpp は人工データを対象とした際の距離計算の省略率が高い．これは kmpp がユークリッド空間中に万遍なく m 個のピボットを選択するため，一様な分布を持つデータセットに適しているからであると考えられる．提案法と比べ，size は所属データ点数の多い重心を上位からピボット数 m 分選ぶという簡易な選択法であるにも関わらず，提案法と同等の省略率を記録した．これは今回のデータセットにおいて所属データ点数の多い重心が比較的遠距離に存在しており，提案法の既存ピボットとの距離が大きな重心ほど選ばれやすくなる選択基準によって選ばれた重心多くの場合一致したからであると考えられる．

5. 関連研究

距離計算の省略以外の Lloyd 法を高速化するアプローチとしては下記が挙げられる．大規模データの一部をサンプリングして近似結果を高速に得る手法の中で，Sculley は既存法と比較して Lloyd 法との異なりが小さいサンプリング手法を提案している [7]．Arthur らは，重心の初期座標の与え方を工夫することで，収束する解の質が向上するとともに，結果的に収束も早まることを示している [3]．これらの方法は提案法と併用することができる．

6. おわりに

ピボットを用いて大規模データを多クラスに分割する Lloyd 法の高速化法を提案した．提案法は，従来の高速化法と比較して次のような利点がある．(1)Moore 法と異なり索引を必要とせず，(2)Elkan 法では必要な記憶容量が大きく適用困難である多クラス分割の場合であっても，必要な記憶容量を汎用計算機が具備する程度の小記憶容量に抑制でき，(3)Hamerly 法と異なりクラス数が増大して効率的に距離計算の省略が行われる．人工データと画像データとを用いた実験により，提案法によって分割対象のデータセットのユークリッド空間における次元数が低いか，高い次元数を持っていても，ユークリッド空間中におけるデータ点の分布が一様でなければ Lloyd 法と比較して 30-50%の距離計算を省略できることを確認した．

謝辞

実験遂行に御協力頂いた日本システムウエア株式会社の藤本裕文氏に深く感謝致します．

参考文献

[1] Amsterdam Library of Object Images (ALOI). `http://aloi.science.uva.nl/`.

[2] flickr. `https://www.flickr.com/`.

[3] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[4] C. Elkan. Using the triangle inequality to accelerate k-means. In *ICML*, volume 3, pages 147–153, 2003.

[5] G. Hamerly. Making k-means even faster. In *SDM*, pages 130–140. SIAM, 2010.

[6] A. W. Moore. The anchors hierarchy: Using the triangle inequality to survive high dimensional data. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, UAI'00, pages 397–405, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[7] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM, 2010.

[8] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip, et al. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2008.