Vol.2015-DBS-162 No.19
2015/11/26

L1距離上でのクラス割り当てピボットによる類似検索

小林 えり[1,a)]　斉藤 和巳[1]　池田 哲夫[1]　青山 一生[2]　服部正嗣[2]

概要：ビックデータに対する類似検索の課題として膨大な計算コストがあげられる．この類似検索の対象となる画像，記事データなどにはジャンルやカテゴリなどの“クラス”によってデータが分類されていることが多い．類似検索高速化法一つであるピボット法において，このようなクラス情報を考慮することにより，高性能なピボットの生成が期待できる．本稿では，L1 距離定義に着目したクラス割り当てピボット法を提案する．手書き文字データ，記事データを用いた実験において，クラス情報を考慮しないピボット生成法と比べて，より高速に，同程度の類似検索性能を持つピボットを生成できることを確認した．また，提案法は高次元かつ大規模データにも適用可能なことが分かった．

1. はじめに

近年，Web 上には多量のデータが蓄積されており，そのデータから所望のデータを速やかに見つけ出す技術は重要である．その一つに与えられたクエリオブジェクトに類似するオブジェクトをデータベースから検索するという問題を解く高速類似検索技術がある．この問題の対象となるデータは，多くの場合，高次元特徴ベクトルで表現され，このベクトル間の非類似度として距離公理を満たす距離関数が用いられる．類似検索の高速化には，高次元特徴ベクトル間の距離計算の回数を削減する必要がある．距離計算回数の削減を効率的に行う方法として，ピボット法がある．ピボット法では距離公理の三角不等式から導かれる距離の下限値を用いて厳密な距離計算が不要であるオブジェクトを判別し，計算回数を削減する手法である．本研究では，距離空間の任意のオブジェクトに対して幅広い適用性を持つピボット法に着目する．

効果的なピボット集合を選択する手法として Bustos らはインクリメンタル法を提案している [Bustos 03]．このインクリメンタル法では，より良いピボット集合の指標として与えられたデータベース内の全オブジェクト間の距離の下限値の総和を目的関数として定義し，目的関数を最大化するようなピボットをオブジェクト集合から逐次選択しピボット集合を得る．これに対し，オブジェクト空間の任意の点をピボットとして求める一般化ピボット法も提案されている [Kimura 09, Kobayashi 14]．一般に，オブジェクト集合の中からピボット集合を選択する場合と比較し，オブジェクト空間の任意の点としてピボット集合を求めれば目的関数値の向上が自然に期待できる．特に，実データに L1 距離定義を適用した実験では，目的関数，ピボット集合構築時間，レンジクエリによる類似検索性能の観点で一般化ピボット法は従来法よりも優れた性能を示すことが確認されている [Kobayashi 14]．更に，前述の L1 距離定義と同じ目的関数を用いた場合，“求めるピボット数が 2 つ”であれば対象オブジェクト集合（データベース）のサイズ N に対して，目的関数値の算出に要する計算量を $O(N^2)$ から $O(N \log N)$ に抑えることができる [Kobayashi 13]．

しかしながら，L1 距離定義を用いた一般化ピボット法であっても，大規模高次元データに直接適用することは，次の理由より困難である．文献 [Kobayashi 14] の方法ではピボット集合構築に要する計算量は，アルゴリズム反復回数を T，ピボット数を K，ベクトルの次元数を H，オブジェクト数を N とすると，その計算量は $O(TKHN)$ であり，大規模高次元データの場合，HN が非常に大きな値となる．一方，文献 [Kobayashi 13] の方法は，ピボット数が 2 つの場合に適用できる方法であり，2 つのピボットでは，大規模高次元データのオブジェクト間の距離の下限値と実際の距離との差が大きくなり，距離計算の削減効率が低下する可能性がある．

文献 [Kobayashi 14] の方法を大規模高次元データに適用する素朴な方法の 1 つに，N の代わりにデータベースからランダムサンプリングした比較的小さいサイズのサブデータ集合（学習データ集合）を用いてピボットを生成し，大規模データへの検索高速化を図るアイデアが提案されている

[1] 静岡県立大学 経営情報イノベーション研究科
Graduate School of Management and Information of Innovation, University of Shizuoka
[2] NTT コミュニケーション科学基礎研究所
NTT Communication Science Laboratories
[a)] j14105@u-shizuoka-ken.ac.jp

[Bustos 03, Kobayashi 15A]．一般に，小さいサイズの学習データ集合で生成されたピボット集合による距離計算削減率は低い．即ち，ピボット集合生成の計算コストとその性能との間にはトレードオフの関係がある．そこで，データベース中のオブジェクトが有する何らかの情報に基づいて学習データ集合を選別し，低計算コストと高性能を同時に実現することが望ましい．

類似検索の対象である画像，文書，遺伝子構造などのデータには該当ジャンルや種類などの，“ クラス ”が割り振られていることが多い．例えば記事データには政治，スポーツといった記事のジャンル，写真では風景，人物などといった撮影対象，twitter ™ のツイートではツイート本文に付随されたハッシュタグなどがクラスに該当する．一般に，類似しているオブジェクトペアというのは似た構造，例えばオブジェクトベクトルの場合，2つのベクトルは同じ要素に近似した値を格納した構造を持つため，クエリと同じクラスのオブジェクトが類似オブジェクトとして抽出される可能性が高いことが自然である．

本研究で着目するピボット法は，クエリとデータオブジェクト間の距離計算を枝刈りすることで検索の高速化を図る手法である．以降，枝刈りするという表現を，探索空間を削減する又は探索対象から除外すると意味で用いる．クラス属性を持つデータベースを対象とする場合，距離計算削減率の向上にクラス属性を利用できるピボットが望まれる．クラス属性の有効的な利用法には次のようなものがある．クエリの属するクラス以外のオブジェクトを効率的に枝刈りする．また，同一クラスに関してもより詳細な分類や振り分け困難な潜在的な分類が存在している場合があるため，クラス内オブジェクトを効率的に枝刈りする．クラス内の分類についての具体例としては，記事のスポーツジャンルの場合，“ 野球 ”，“ サッカー ”などのサブジャンルや，“ 野球 ”ジャンルに着目しても “ 高校生野球 ”，“ 社会人野球 ”などより詳細な分類があげられる．

本研究では，大規模高次元データベースでの類似検索高速化を目的に，データオブジェクトにクラス情報が付与されているデータベースを対象として，クラス割り当てピボットによる類似検索法を提案する．クラス割り当てピボットとは，クラス情報を考慮して生成されたピボットを指し，具体的には，クラス j 以外のクラスに属するオブジェクトを効果的に枝刈りするためのピボットと，クラス j 内のオブジェクトを効果的な枝刈りするためのピボットを指す．また，提案法では文献 [Kobayashi 13] で提案された高速化アルゴリズムを採用しており，クラス毎に高速に最適なピボット集合を求めることができる．

本稿の構成は以下となる．まず，類似検索問題について説明する．次に，一般化ピボット法について述べる．次いで，クラス割り当て一般化ピボット法について述べる．そして，使用データ等の実験設定について述べる．その後，実データを用いた，提案法の性能評価を報告する．最後に，本研究をまとめ及び，今後の課題について述べる．

2. 類似検索問題

類似検索問題には様々な問題設定がある [Zezula06, Samet06]．例えば，与えられた空間に対する解の性質，即ち厳密解か近似解かの設定や問い合わせ方法，クエリから k 番目までのオブジェクトを検出する k-NN クエリ問題，またはクエリからある一定の距離以内のオブジェクトを検出するレンジクエリかの設定などがある．本稿ではクエリから一定のレンジ内にあるオブジェクトの厳密解を求めるレンジクエリ問題を扱う．なお，適切なレンジ距離を設定することにより，レンジクエリ問題を k-NN クエリ問題へ拡張することも可能である．

レンジクエリ問題は，オブジェクト集合 $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ とクエリ $\mathbf{q}$ とレンジ r が与えられたとき，$\mathbf{q}$ と $\mathbf{x}_n$ の距離 $d(\mathbf{x}_n, \mathbf{q})$ が r 以下となるようなオブジェクト集合を求める問題である．本稿では，この問題を解くのに要する計算時間を短縮するためにピボット法を用いる．

3. 一般化ピボット法

ピボット法は，オブジェクト間の距離計算回数を削減し検索を高速化させるために，一部のオブジェクトを選定してピボット集合を求める．例えば，Bustos らの提案した局所最適選択法（local optimum selection）や逐次選択法（incremental selection）では，目的関数を最大化するようなピボット集合を最適なピボット集合として求める．Bustos らの手法では，与えられたオブジェクト集合の中からピボットを逐次選択するのに対し，一般化ピボット法では，オブジェクト空間の任意の点をピボットとして生成する．本稿では一般化ピボット法に着目する．

今，二つのオブジェクト集合 $Y, Z \subset X$ が与えられたとき，ピボット集合 P による一般化ピボット法での目的関数 $\mathcal{F}(P; Y, Z)$ は以下のように定義される．

$$P = \arg\max_{P} \mathcal{F}(P; Y, Z) \tag{1}$$

$$\mathcal{F}(P; Y, Z) = \sum_{\mathbf{y} \in Y} \sum_{\mathbf{z} \in Z} D(\mathbf{y}, \mathbf{z}; P \subset \boldsymbol{R}^H) \tag{2}$$

$$D(\mathbf{y} \in \boldsymbol{Y}, \mathbf{z} \in \boldsymbol{Z}; P \subset \boldsymbol{R}^H) = \max_{1 \le k \le |P|} |d(\mathbf{y}, \mathbf{p}_k) - d(\mathbf{z}, \mathbf{p}_k)| \tag{3}$$

$\boldsymbol{R}^H$ は H 次元のユークリッド空間で表現されるオブジェクト空間を指す．ここで，ピボット集合を X ではなく，$\boldsymbol{R}^H$ より選択している点に留意されたい．なお，文献 [Kobayashi 15A] では $Y \cap Z = \emptyset$ での場合を想定してしたのに対し，本稿では Y, Z は任意のオブジェクトより構築される集合であり，$Y = Z = X$ となる場合（単一のオブジェクト集合のみで目的関数を最大化する場合）は文献 [Kobayashi 14] の方法に帰着するので自然な拡張である．

式 3 の max 関数内は，オブジェクトペア $\{\mathbf{y},\mathbf{z}\}$ の距離に対する，ピボット $\boldsymbol{p}_k$ から各オブジェクトまでの距離を用いて算出した下界値を意味する．従って，式 1 は $\boldsymbol{p}_1,\boldsymbol{p}_2,\cdots,\boldsymbol{p}_K$ を用いて算出した最大下界値である．レンジクエリ問題と距離の最大下界値との関係を明確にするために，最大下界値が r より大きいオブジェクト集合を $E(r)=\{\mathbf{x}_n \mid D(\mathbf{y},\mathbf{z};P)>r\}$ と定義する．集合 Z をクエリオブジェクト集合と見なすならば，$E(r)$ に属するオブジェクト集合の要素については，ピボットから求まる下限値によりオブジェクト z との実距離を計算することなくレンジ外であると判断できるため，レンジクエリ問題の解を求めるまでの時間の短縮が期待できる．

4. クラス割り当て一般化ピボット法

本稿で提案するクラス割り当て型ピボット法ではクラス毎に自身のクラス以外のオブジェクトを枝刈りするピボットと，自身のクラス内オブジェクトを枝刈りするピボットを生成する．ここで，ピボット数 K が 2 個までであれば，目的関数計算に文献 [Kobayashi 13] で提案されている高速化アルゴリズムを適用できる．

クラス数を J とし，提案法で用いるピボット集合は $\bar{P}=\{\bar{P}_1,\cdots,\bar{P}_J,\bar{P}_{J+1},\cdots,\bar{P}_{2J}\}$ である．$\bar{P}_j$ はクラス j $(1\le j\le J)$ とその他クラスのオブジェクトを効率的に枝刈りするピボット集合であり，$\bar{P}_{J+j}$ はクラス j 内のオブジェクトを効率的に枝刈りするピボット集合を指す．よって，クラス j に割り当てられるピボット数は，自クラスと他クラスの枝刈り用で 2 つ $(|\bar{P}_j|=2)$，自クラス同士の枝刈り用で 2 つ $(|\bar{P}_{J+j}|=2)$ の計 4 つとなる．即ち，$|\bar{P}_j|=2$，$|\bar{P}|=4J$ である．

提案法の対象となるデータベースはクラスによる分類を有し，クラス j に属するオブジェクトのみから成るサブオブジェクト集合を X_j とすると，$X=\cup_{j=1}^{J}X_j$ と表すことが出来る．ここで，オブジェクトの所属するクラスは単一であり，複数のクラスに属することはないとする $(X_j\cap X_{j'}=\emptyset\ (j\ne j'))$．

式 2 を用いて，クラス割り当て型一般化ピボット法（以下，L1CPGM 法と呼ぶ）にて最大化する目的関数は次のようになる．

$$\mathcal{G}(\bar{P};X)=\sum_{j=1}^{J}\big(\,\mathcal{F}(\bar{P}_j;X-X_j,X_j)+\mathcal{F}(\bar{P}_{J+j};X_j,X_j)\,\big)\quad(4)$$

式 4 の $\mathcal{F}(\bar{P}_j;X-X_j,X_j)$，$\mathcal{F}(\bar{P}_{J+j};X_j,X_j)$ の最適化及びピボット更新に関しては $F(P,Y,Z)$ を基に次節にて説明する．

4.1 高速化アルゴリズムの導入

文献 [Kobayashi 13] では，単一のオブジェクト集合 $(\mathcal{F}(P;X,X))$ でのアルゴリズムが記載されている．ここでは，複数のオブジェクト集合を用いる必要があるため，前述のアルゴリズムの目的関数を $\mathcal{F}(P;Y,Z)$ に拡張する．

ここで，オブジェクト $\mathbf{y}$ とオブジェクト集合 Z 内の全オブジェクトを，ピボット $\mathbf{p}_1$ とオブジェクト間の距離とピボット $\mathbf{p}_2$ とオブジェクト間の距離の和の値で降順ソートしたときの順位を $S_1(\mathbf{y},Z)$ とし，$\mathbf{y}$ と Z 内の全オブジェクトを，オブジェクトと $\mathbf{p}_1,\mathbf{p}_2$ 間の距離の差の値で降順ソートしたときの順位を $S_2(\mathbf{y},Z)$ とする．このとき，式 2 は次のようになる．

$$\begin{aligned}\mathcal{F}(P;Y,Z)\ =&\sum_{\mathbf{y}\in Y}\Big(S_1(\mathbf{y},Z)+S_2(\mathbf{y},Z)-|Z|\Big)d(\mathbf{y},\mathbf{p}_1)\\&+\sum_{\mathbf{y}\in Y}\Big(S_1(\mathbf{y},Z)-S_2(\mathbf{y},Z)\Big)d(\mathbf{y},\mathbf{p}_2)\\&+\sum_{\mathbf{z}\in Z}\Big(S_1(\mathbf{z},Y)+S_2(\mathbf{z},Y)-|Y|\Big)d(\mathbf{z},\mathbf{p}_1)\\&+\sum_{\mathbf{z}\in Z}\Big(S_1(\mathbf{z},Y)-S_2(\mathbf{z},Y)\Big)d(\mathbf{z},\mathbf{p}_2)\\=&\sum_{k=1}^{2}\Big(\sum_{\mathbf{y}\in Y}c_k(\mathbf{y},P)d(\mathbf{y},\mathbf{p}_k)+\sum_{\mathbf{z}\in Z}c_k(\mathbf{z},P)d(\mathbf{z},\mathbf{p}_k)\Big)\quad(5)\end{aligned}$$

$c_k(\mathbf{y},P)$ は $d(\mathbf{y},\mathbf{p}_k)$ に対する係数とし，同様に，$c_k(\mathbf{z},P)$ は $d(\mathbf{z},\mathbf{p}_k)$ に対する係数である．

ここで，$d(\mathbf{y},\mathbf{p}_k)$ の係数 $c_k(\mathbf{y},P)$ は，$k=1$ の場合，$S_1(\mathbf{y},Z)+S_2(\mathbf{y},Z)-|Z|$ であり，$k=2$ の場合，$S_1(\mathbf{y},Z)-S_2(\mathbf{y},Z)$ となる．同様にして，$d(\mathbf{z},\mathbf{p}_k)$ の係数 $c_k(\mathbf{z},P)$ は，$k=1$ の場合，$S_1(\mathbf{z},Y)+S_2(\mathbf{z},Y)-|Y|$ であり，$k=2$ の場合，$S_1(\mathbf{z},Y)-S_2(\mathbf{z},Y)$ となる．

貪欲法による式 4 の目的関数の最大化には，ピボット数が 2 個の場合，オブジェクト数を N とすると目的関数算出にかかる計算量は $O(N^2)$ となるが，これに対し，文献 [Kobayashi 13] の高速化アルゴリズムの場合, $O(NlogN)$ の計算量で済む．

4.2 ピボット更新

本研究では，L1 距離定義に基づく一般化ピボット法に着目する．L1 距離定義に着目するのは, 大規模写真画像のコンテンツデータベースである CoPhIR(Content-based Photo Image Retrieval) にて採用されているように，MPEG-7 の一般記述において L1 距離は優れた性能を示すことが報告されているからである．次元数を H，オブジェクトベクトルの h 次元の値を $x_{n,h}$ とすると，L1 距離定義によるオブジェクトペア $\mathbf{x}_n,\mathbf{x}_m$ 間の距離は以下の式で定義される．

$$d(\mathbf{x}_n,\mathbf{x}_m)=\sum_{h=1}^{H}|x_{n,h}-x_{m,h}|\qquad(6)$$

ここで，オブジェクト集合 Y,Z を結合させた集合を $B=\{\mathbf{b}_1,\cdots,\mathbf{b}_{|Y|},\mathbf{b}_{|Y|+1},\cdots,\mathbf{b}_{|Y|+|Z|}\}$，オブジェクト集合 Y,Z 内の各オブジェクトとピボット $\mathbf{p}_k$ との距離 $d(\mathbf{y},\mathbf{p}_k)$（または $d(\mathbf{z},\mathbf{p}_k)$）と対応する係数 $c_k(\mathbf{y},P)$(または $c_k(\mathbf{z},P)$) をまとめた係数の集合を
$C_k=\{c_k(\mathbf{y}_1,P),\cdots,c_k(\mathbf{y}_{|Y|},P),c_k(\mathbf{z}_1,P),\cdots,c_k(\mathbf{z}_{|Z|},P)\}$
とする．
$\mathbf{b}_1$ は $\mathbf{y}_1$ を指し，$\mathbf{b}_1$ に対応する係数は $c_k(\mathbf{y}_1,P)$ となるた

め，係数集合 C_k は $C_k = \{c_k(\mathbf{b}_1, P), \cdots, c_k(\mathbf{b}_{|Y|+|Z|}, P)\}$ のように書き換えることができる．よって，式 5 の目的関数は以下のように書き換えることができる．

$$\begin{aligned}\mathcal{F}(P;Y,Z) =& \sum_{k=1}^{2}\Big(\sum_{\mathbf{y}\in Y} c_k(\mathbf{y},P)d(\mathbf{y},\mathbf{p}_k) + \sum_{\mathbf{z}\in Z} c_k(\mathbf{z},P)d(\mathbf{z},\mathbf{p}_k)\Big)\\ =& \sum_{k=1}^{2}\sum_{n=1}^{|B|} c_k(\mathbf{b}_n,P)d(\mathbf{b}_n,\mathbf{p}_k)\\ =& \sum_{k=1}^{2}\sum_{h=1}^{H}\sum_{n=1}^{|B|} c_k(\mathbf{b}_n,P)|b_{n,h}-p_{k,h}|\\ =& \sum_{k=1}^{2}\sum_{h=1}^{H}\mathcal{F}(p_{k,h}) \qquad (7)\end{aligned}$$

$\mathcal{F}(p_{k,h})$ はピボット $\mathbf{p}_k$ での h 次元での目的関数を表し，以下のように定義される．

$$\mathcal{F}(p_{k,h}) = \sum_{n=1}^{|B|} c_k(\mathbf{b}_n,P)|b_{n,h}-p_{k,h}| \qquad (8)$$

式 7 で定義した目的関数の最大化は，各 $p_{k,h}$ に対し，独立に式 8 を最大化できることが分かる．文献 [Kobayashi 14] より，式 8 の目的関数の最大値は区分線形式の交点，つまりはオブジェクト集合 B 内のオブジェクトの h 次元の値 $b_{n,h}$ のどれかで与えられる．すなわち，ピボット $\mathbf{p}_k$ の第 h 座標値の更新式は以下となる．

$$p_{k,h} = \underset{\{\mathbf{b}_{n,h} \mid 1\le n\le |B|\}}{\arg\max} \mathcal{F}(b_{n,h}) \qquad (9)$$

以下に $\mathcal{F}(P;Y,Z)$ を最適するピボット集合構築アルゴリズムを記述する．

step1 初期化：ピボット集合 P をランダムに生成する．

step2 距離計算：任意の n と k のペアに対して $d(\mathbf{y}_n,\mathbf{p}_k)$ 及び $d(\mathbf{z}_n,\mathbf{p}_k)$ を式 6 で計算する．

step3 係数計算：ピボット p_1 と p_2 との距離の和および距離の差でオブジェクト $\mathbf{y}_n$ と集合 Z を，$\mathbf{z}_n$ と集合 Y をそれぞれ降順ソートし，$S_1(\mathbf{y},Z), S_2(\mathbf{y},Z)$ 及び $S_1(\mathbf{z},Y), S_2(\mathbf{z},Y)$ を計算する．

step4 ソート :Y,Z を結合したオブジェクト集合 B を，各次元 h ごとにオブジェクトの座標値 $b_{n,h}$ で昇順ソートする．

step5 ピボット更新：任意の k と h のペアに対して $p_{k,h}$ を式 9 で求める．

step6 判定：目的関数値 $\mathcal{F}(P;Y,Z)$ が向上しなければピボット集合 P を出力して終了，さもなければ，step2 へ戻る．

4.3 L1CPGM 法への拡張

$\mathcal{F}(P;Y,Z)$ でのアルゴリズムを基に，式 4 は以下のように書き換えることができる．

$$\begin{aligned}\mathcal{G}(\bar{P};X) =& \sum_{j=1}^{J}\big(\,\mathcal{F}(\bar{P}_j;X\text{–}X_j,X_j) + \mathcal{F}(\bar{P}_{J+j};X_j,X_j)\,\big)\\ =& \sum_{j=1}^{J}\Big(\sum_{k=1}^{2}\sum_{n=1}^{|B_j|} c_k(\mathbf{b}_j n,\bar{P}_j)d(\mathbf{b}_j n,\bar{\mathbf{p}}_k)\\ &+\sum_{k=1}^{2}\sum_{n=1}^{|B_{(J+j)}|} c_k(\mathbf{b}_{(J+j)}n,\bar{P}_{(J+j)})d(\mathbf{b}_{(J+j)}n,\bar{\mathbf{p}}_k)\,\Big)\\ =& \sum_{j=1}^{J}\Big(\sum_{k=1}^{2}\sum_{h=1}^{H}\mathcal{F}(\bar{p}_{jk,h}) + \sum_{k=1}^{2}\sum_{h=1}^{H}\mathcal{F}(\bar{p}_{(J+j)k,h})\Big) \qquad (10)\end{aligned}$$

ここで，B_j，B_{J+j} に関して，B_j は $\mathcal{F}(\bar{P}_j;X\text{–}X_j,X_j)$ より，$B_j = \{b_1,\cdots,b_{|X_j|},\cdots,b_{|X|}\}$ であり，
$B_{(J+j)}$ は $\mathcal{F}(\bar{P}_{J+j};X_j,X_j)$ より，$B_{(J+j)} = \{b_1,\cdots,b_{|X_j|}\}$ となる．ピボット $\bar{\mathbf{p}}_{jk}$ はピボット集合 $\bar{P}_j$ の第 k 番目ピボットを示し，$\bar{p}_{jk,h}$ はピボット $\bar{\mathbf{p}}_{jk}$ の第 h 座標値を示す．

L1CPGM 法では，クラス j 毎に，$\mathcal{F}(\bar{P}_j;X\text{–}X_j,X_j)$ 及び，$\mathcal{F}(\bar{P}_{J+j};X_j,X_j)$ に対して $F(P,Y,Z)$ と同等のアルゴリズムを適用させ，式 10 の最適化及びピボット更新を行う．

5. 実験設定

5.1 実験データ

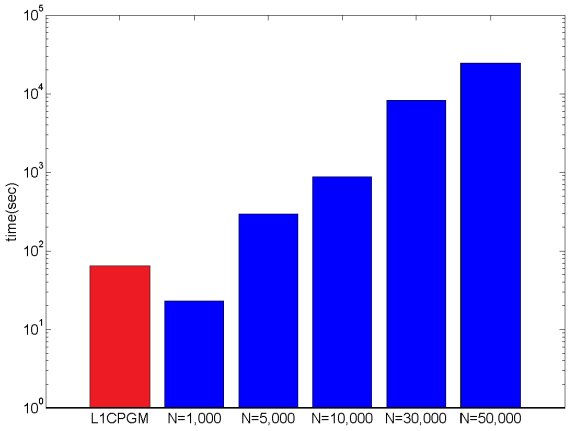
実験データとして YahooNews の記事データと MNIST を用いた．

YahooNews では，各記事を形態素解析して得られた単語頻度ベクトルをオブジェクトベクトルとする．記事データセットの総オブジェクト数 (総記事数) は 324,528，総出現ターム種類数（異なり単語数：ベクトルの次元数）は 91,522 である．記事の平均ターム種類数（平均異なり単語数：有効次元数）は 77 であり，構成ターム数（出現単語数：ベクトルの要素の総和）は 121 である．YahooNews データは高次元空間で表現されるが，各オブジェクトベクトルは有効次元数が小さく，データ全体は非常にスパースな構成である．YahooNews は “ 国内 ”, “ 経済 ”, “ エンタメ ”, “ 生活 ”, “ 地域 ”, “ サイエンス ”, “ スポーツ ”, “ 世界 ” の 8 つのジャンルに分類され，一つの記事が複数のジャンルを持つことはない．本実験では，ジャンルをクラスとし，単語頻度ベクトルは TFIDF による前処理を行ったものを使用する．

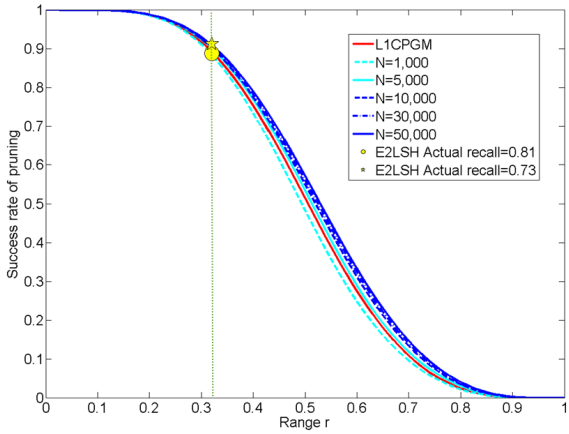
MNIST は手書き文字認識用データベースであり，0〜9 のいずれかの手書きアラビア数字画像 60,000 からなる．各オブジェクト（画像）は画素数が縦横 28 × 28=784，各画素が 0〜255 の 256 階調グレースケールで表現されており，それぞれ 0〜9 の数字ラベルが付与されている．実験では，10 種の数字ラベルをクラスとし，各オブジェクトの特徴量を 784 個の画素値を要素とする 784 次元ベクトルとした．

5.2 比較手法

比較手法として，データオブジェクトのクラス情報を考慮せずに最適ピボットを生成する，文献 [Kobayashi 14] に

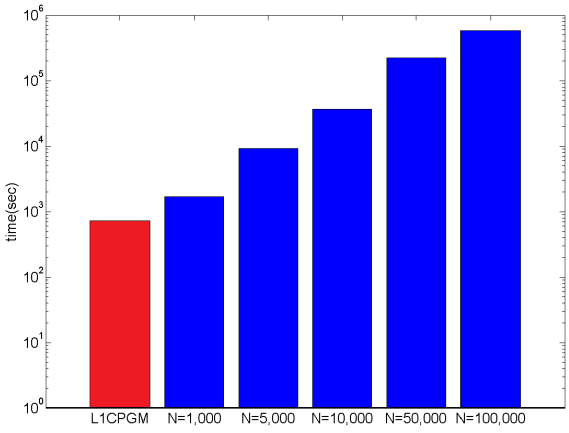


（a）ピボット集合構築時間（秒）

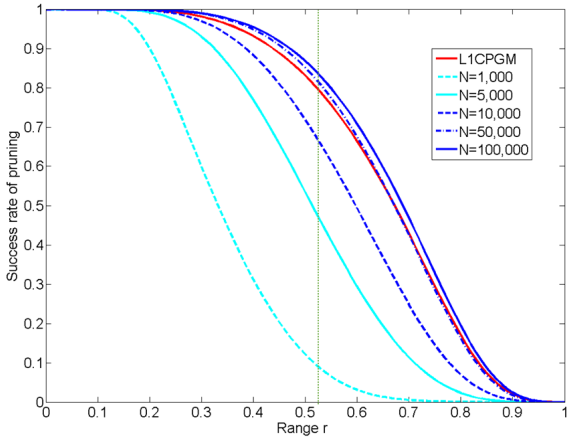


（b）枝刈り成功率

図 1　MNIST での結果

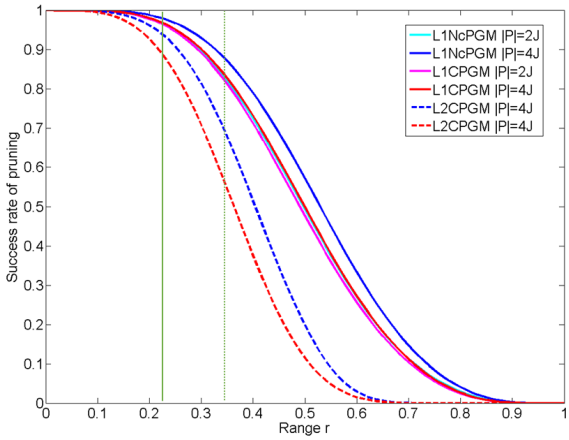


（a）ピボット集合構築時間（秒）

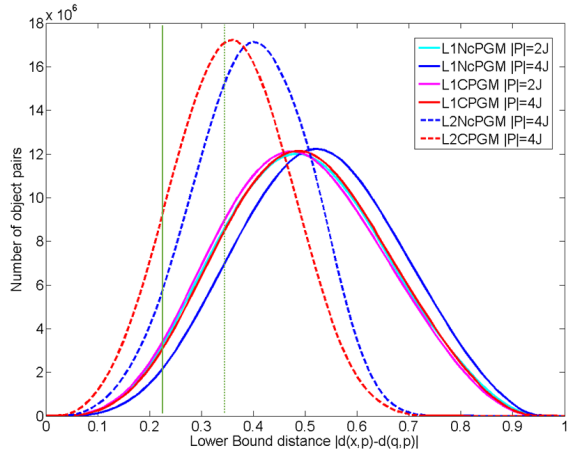


（b）枝刈り成功率

図 2　YahooNews での結果

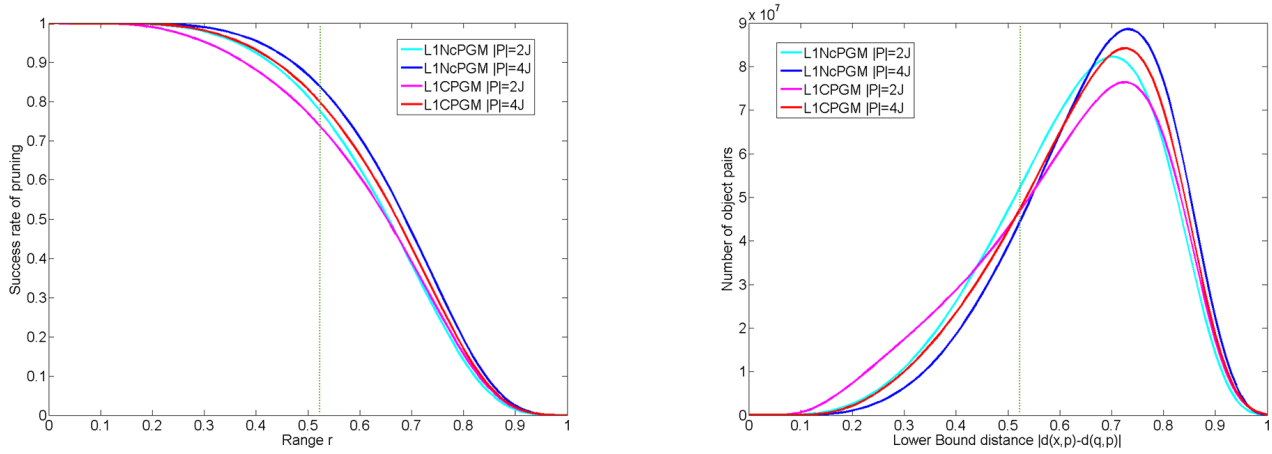


（a）枝刈り成功率



（b）LB 分布

図 3　ピボット数変化による枝刈り性能の評価（MNIST）



（a）枝刈り成功率　　（b）下限値分布

図 4　ピボット数変化による枝刈り性能の評価（YahooNews）

て提案されている L1 距離定義に基づく一般化ピボット法 (以下 L1NcPGM 法) を用いる．L1NcPGM 法にて最適化する目的関数は以下の式で定義される．

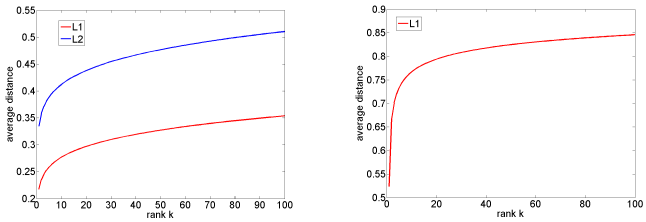
$$P_N = \arg\max_P \mathcal{F}(P) \tag{11}$$

$$\mathcal{F}_N(P;X,X) = \sum_{n=1}^{|X|-1} \sum_{m=n+1}^{|X|} D(\mathbf{x}_n, \mathbf{x}_m; P \subset \boldsymbol{R}^H) \tag{12}$$

提案法ではジャンル数 J より，ピボット数 K は決定されるが，L1NcPGM 法ではピボット数は使用者が予め設定する必要がある．本実験では提案法に合わせてピボット数を $K = 4J$ とする．

なお，提案法（L1CPGM 法）と従来法（L1NcPGM 法）は最適ピボット集合構築のアプローチが異なるだけで，計算に用いるメモリコストは同量であることに注意されたい．

5.3　評価指標



（a）MNIST　　（b）YahooNews

図 5　クエリに対する k-NN 距離の平均値

提案法は厳密解を求める方法であるため，評価指標として高速性を評価する枝刈り成功率を用いる．本実験では，ピボットの性能を評価する評価クエリ集合 Q として，オブジェクト集合 X から 10,000 個のオブジェクトをランダムサンプリングにより選択し，残りを検索用として $\hat{X} = X \setminus Q$ とする．即ち，$Q \subset X$，$Q \cap \hat{X} = \emptyset$，$|Q| = 10000$ である．レンジ距離 r にて，$|d(x \in \hat{X}, p) - d(q \in Q, p)| \leq r$ 満たす，ピボット集合 P で距離計算を省略することができたオブジェクトの集合を $E(r)$ とすると，枝刈り率は $E(r)/|\hat{X}|$ で表される．

本論文では，高次元大規模データベースでの類似検索への適用性を検証するため，データサイズ X が 6 万，約 32 万と比較的大規模かつ，次元数 H が 728，約 9 万と高次元なデータベースでの枝刈り成功率により提案法の有用性を示す．

6.　実験結果

本実験では，ピボット構築時間とそのピボットでの枝刈り成功率を用いて評価する．図 1(a)，図 2(a) にピボット集合構築に要した時間（秒単位）を示し，図 1(b)，図 2(b) に構築したピボット集合での枝刈り性能を示す．なお，図 1(a) と図 2(a) のピボット集合構築時間の縦軸は対数表示であることに注意されたい．枝刈り成功率を示す図 1(b)，図 2(b) では，横軸がレンジ距離 r であり，縦軸が r での枝刈り成功率 $E(r)/|\hat{X}|$ である．

枝刈り成功率に関して，着目すべきレンジ距離の参考に，k-NN クエリ問題を用いる．図 5 はクエリ $\mathbf{q} \in Q$ とデータベース $\hat{X}$ 内のオブジェクトによる k 近傍オブジェクトとの距離の平均値をプロットした図である．横軸にランク k を，縦軸にクエリと k 番目に近傍なオブジェクトとの距離の平均値をとり，例えば，図 5(b) より，YahooNews においてクエリと最近傍オブジェクトとの平均 L1 距離は約 0.52 であることが分かる．

従来法（L1NcPGM 法）は集合 $\hat{X}$ のオブジェクト全てを用いて目的関数を最大化させるピボットを生成するため，集合 $\hat{X}$ のサイズが大きい場合，ピボット生成に多大な計算コストを要する．そのため，$\hat{X}$ よりランダムサンプリングした学習データ集合 $X' \subset \hat{X}$ を用いてピボット集合を構築する．

一方，提案法（L1CPGM 法）は，大規模データベースへの適用を前提としているため，評価クエリ Q を除く全データオブジェクトを用いてピボットを生成し，評価検証を行う．

6.1 ピボット集合の構築時間とその枝刈り性能

6.1.1 MNIST での性能

図 1 に MNIST でのピボット集合構築の要した時間と構築されたピボット集合による枝刈り性能を示す．L1NcPGM 法では，性能比較のため，ピボット生成の学習データ集合 X' のサイズ N を $1,000$，$5,000$，$10,000$，$30,000$，$50,000$ と変えた．

図 1(a) より，N=1,000 を除く全てのパターンの L1NcPGM 法よりも，L1CPGM 法の方が高速にピボット集合を構築できていることがわかる．図 1(b) より，そのピボットの性能は $N = 5,000$ での L1NcPGM 法と同程度の性能である．性能に着目すれば，データサイズ N が大きいパターンでの L1NcPGM 法の方が提案法より優っているが，N の増加に伴い構築時間は飛躍的に増加するため，大規模データベースへの適用は厳しいと分かる．つまり，L1CPGM 法は L1NcPGM 法よりも短い時間で，高い枝刈り性能を持つピボットを生成することが期待できる．

次に，ピボット法以外の類似検索高速化手法と比較し，提案法の有用性を示す．本実験では、近年広く利用されている locality-sensitive hashing(LSH) の典型である Exact Euclidean LSH (E2LSH) を比較対象法とする。図中の★，●は E2LSH による枝刈り成功率を示す．E2LSH にて設定するレンジ距離 r は文献 [Aoyama 11, Andoni 06] より，$r = 0.65$ と設定した．しかしながら，文献 [Aoyama 11, Andoni 06] と本稿では距離の正規化及び，距離定義が異なるため，レンジ距離を適切な値に変換し直す必要がある．文献 [Aoyama 11 ,Andoni 06] での距離の正規化は $0 \leq d(,) \leq 2$ であるのに対し，本実験では $0 \leq d(,) \leq 1$ であるため，レンジ距離は $(r/\sqrt{2})$ となる．さらに，このレンジ距離 [Aoyama 11 ,Andoni 06] は L2 空間での設定のため，L1 空間での距離に直す必要がある．図 5(a) より，L1 距離と L2 距離の比は，およそ $L1 : L2 = \sqrt{2} : 1$ と判断でき，本実験では $r = (0.65/\sqrt{2})/\sqrt{2} = 0.325(\sim 0.32)$ とする．

E2LSH の AC (Actual recall)=0.73 での枝刈り成功率を★，AC=0.81 での枝刈り成功率を●でを示す．図 1(b) から，r=0.32 での L1CPGM 法と E2LSH との枝刈り成功率を比較するとほぼ同程度であることが分かる．E2LSH は近似アルゴリズムであり，この場合はその精度が 0.73 と 0.81 であるのに対して，L1CPGM 法は厳密アルゴリズムであり，精度は 1.00 である．このように提案法は近似アルゴリズムと同等の枝刈り成功率，即ち，高速性を達成する優れた方法である．ただし，E2SLH は L2 空間でのアルゴリズムのため，厳密な比較をする場合，L1 定義に基づく LSH と比較する必要があることに注意されたい．

6.1.2 YahooNews での結果

図 2 に YahooNews でのピボット集合構築の要した時間と構築されたピボット集合による枝刈り性能を示す．YahooNews では，学習データ集合 X' のサイズ N を $1,000$，$5,000$，$10,000$，$30,000$，$50,000$，$100,000$ と変えた．

図 2(a) より，L1CPGM 法は最も高速にピボット集合を構築できており，図 2(b) より，その性能は，$N = 50,000$ での L1NcPGM 法とほぼ同性能である．なお，$N = 50,000$ の L1NcPGM 法でのピボット構築時間は L1CPGM 法の 800 倍以上の計算時間要しており，提案法が如何に高速な手法であるかが分かる．図 5(b) の k-NN クエリより，YahooNews でのクエリと最近傍オブジェクトとの平均 L1 距離は 0.52 であり，$r = 0.52$ での性能に着目すると，1CPGM 法の枝刈り成功率は 80%以上であり，類似検索の高速化が期待できる．

L1NcPGM 法について，こちらも学習データ数 N の増加に伴い，枝刈り性能は向上するものの，構築時間は飛躍的に増加するため，大規模データベースへの適用が厳しいと分かる．また，全体的に MNIST に比べて膨大な計算時間を要しているが，これは次元数の大きさ（MNIST:784, YahooNews: 91,522）が影響していると考えられる．

以上より，L1CPGM 法は高次元大規模データに対して高速に，高性能なピボット集合を構築することが期待できると分かった．

6.2 提案法のピボット種類と枝刈り成功率との関係

次に，ピボットの種類による枝刈り性能への影響を評価する．つまり，L1CPGM 法の自クラス同士の枝刈り用ピボット集合の枝刈り性能への貢献度の検証を行う．よって，L1CPGM 法のピボット数 K が $K = 2J$ のものと $K = 4J$ の 2 パターンを評価し，それぞれ最適化する目的関数は $\mathcal{G}(\bar{P}; X)$，$\sum_{j=1}^{J} \mathcal{F}(\bar{P}_j; X - X_j, X_j)$ であることに注意されたい．L1NcPGM 法に関してもピボット数変化による性能評価への影響を検証するため，L1CPGM 法に合わせて $K = 2J, 4J$ のものを評価する．

6.2.1 MNIST での性能

図 3 に MNIST での結果を示す．MNIST のみ，L1, L2 両距離定義での結果も示す．L1NcPGM 法の学習データ集合のサイズは $N = 50,000$ とし，L2 定義での結果は L2CPGM 法，L2NcPGM 法共にピボット数は $K = 4J$ の結果のみ記載する．グラフはそれぞれ，L1 定義の結果は実線で，L2 定義の結果は点線で示し，赤で CPGM 法（$K = 4J$）を，ピンクで CPGM 法（$K = 2J$）を，青で NcPGM 法（$K = 4J$）を，水色で NcPGM 法（$K = 2J$）での結果を示す．

まず，L1CPGM 法の結果より，MNIST では $K = 2J$ と $4J$ 間に大きな性能差が見られないため，自クラス同士のピボットによる枝刈り性能への貢献度は低いと分かる．

一方，L1NcPGM 法では $K = 2J$ と $4J$ 間にはレンジ距離によっては 5%以上の性能差が見られた．しかしながらピボット数の増加に伴い，構築時間も増加するため，妥当なピボット数を見極めることが求められる．

次に，L1 距離定義と L2 距離定義での性能差を評価すると，L1CPGM 法，L1NcPGM 法共に L1 定義での結果の方が高い枝刈り成功率を示している．クエリと最近傍オブジェクトとの平均距離でそれぞれの性能を比較するため，L1 定義の場合 $r = 0.22$ に，L2 定義の場合 $r = 0.34$ に着目すると，L1 定義での両手法とも 95%以上の成功率であり，ほとんどの距離計算が枝刈りできてしまうのに対し，L2CPGM 法では 58%，L2NcPGM 法では 71% であり，大きな性能差が生じた．

距離定義による性能差にはピボットによるクエリとオブジェクトとの下限値 $|d(x,p) - d(q,p)|$ の分布が関係していると考えられる．図 3(b) より，L2CPGM，L2NcPGM 法による分布の最頻値は $r = 0.34$ に対してわずかに右寄りであるのに対し，L1CPGM，L1NcPGM 法による分布の最頻値は $r = 0.22$ よりもだいぶ右寄りであるため，高い枝刈り成功率が望める．つまり，L1 距離定義の方が，L2 距離定義よりも下限値がよりタイト（実距離に近い）であるため，枝刈り成功率に差が生じたのである．

L1，L2 距離定義による枝刈り成功率性能の差に関して，文献 [Kobayashi 15B] にて YahooNews を用いて，データ全体が非常にスパースな構成であるデータセットでの検証を行ったが，本研究では MNIST を用いて，オブジェクトベクトルの有効次元数が大きく，データ全体が密な構成であるデータセットに対しても，L1 距離定義の方が高い枝刈り成功率を期待できることを確認した．

6.2.2 YahooNews での性能

図 4 に YahooNews での枝刈り成功率を示す．赤で L1CPGM 法（$K = 4J$）を，ピンクで L1CPGM 法（$K = 2J$）を，青で L1NcPGM 法（$K = 4J$）を，水色で L1NcPGM 法（$K = 2J$）での結果を示す．

図 4（a）より，L1CPGM 法 ($K = 4J$) と L1CPGM 法 ($K = 2J$) の性能差はレンジ距離によっては 10%以上の差が見られ，自クラス同士枝刈り用のピボットが効果的に作用しているのが分かる．MNIST では性能差が見られなかったのに対し，YahooNews で性能差が見られた要因として，クラス i に該当するサブオブジェクト集合 X_i が考えられる．MNIST での各 X_j 内オブジェクト間の平均距離は約 0.75（分散：0.11）に対し，YahooNews での各 X_j 内オブジェクト間の平均距離は約 0.98（分散：0.022）であり，クラス X_j 内でもほとんどの記事が非類似記事であることが分かる．つまり，YahooNews の場合，類似記事（最近傍記事との距離：約 0.52）と非類似記事の差が顕著であり，ピボットが学習しやすい構造（特徴的なオブジェクト分布を持つ構造）であるため，同クラス同士の枝刈り用ピボットの枝刈り効果がより強くなったと考えられる．

L1NcPGM 法に関して，こちらも $K = 2J$ と $4J$ 間にはレンジ距離によっては 5%以上の性能差が見られた．

下限値の分布に関して，両手法とも分布が右寄りになり，最頻値は $r = 0.52$ よりも右に位置するため，高い枝刈り成功率が望める．

7. おわりに

本研究では，大規模高次元データベースでの類似検索高速化を目的に，クラス割り当てピボットによる類似検索法を提案した．実データを用いて，高次元かつ大規模データに対して高速に高性能なピボット集合を構築でき，提案法の有効性を確認できた．今後は他のデータセットでの検証，他の類似検索手法との比較を行うと共に 4.1 節で述べた計算時間コストの理論値を実験的に検証する．

謝辞 本研究は，総務省 SCOPE(No.142306004)，及び，科学研究費補助金基盤研究 (C)(No. 26330138) の助成を受けた．

参考文献

[Bustos 03] B. Bustos, G. Navarro, and E. Chávez: “ Pivot Selection Techniques for Proximity Searching in Metric Spaces, ”Pattern Recognition Letters, Vol.24, No.14, pp. 2357-2366, (2003).

[Zezula 06] P. Zezula, G. Amato, V. Dohnal, and M. Batko: “ Similarity Search: The Metric Space Approach, ” Springer, (2006).

[Samet 06] H. Samet: “ Foundations of Multidimensional and Metric Data Structures, ” Morgan Kaufman, (2006).

[Kimura 09] 木村 学，斉藤 和巳，上田 修功: “ 効率的な類似検索のためのピボット学習法, ” 情報処理学会論文誌, Vol.50, No.8, pp.1883-1891, (2009).

[Aoyama 11] K.Aoyama, K.Saito, H.Sawada, N.Ueda: “ Fast Approximate Similarity Search Based on Degree-Reduced Neighborhood Graphs, ”Proc. of the 17th International Conference on Knowledge Discovery and Data Mining (KDD2011), pp.1055-1063, (2011).

[Kobayashi 13] 小林 えり，伏見 卓恭，斉藤 和巳，池田 哲夫: “ ソートを利用したＬ１ノルム総和計算によるピボット選択の高速化, ”第 12 回情報科学技術フォーラム（FIT2013）, (2013).

[Kobayashi 14] E. Kobayashi, T. Fushimi, K. Saito and T. Ikeda: “ Similarity Search by Generating Pivots Based on Manhattan Distance, ” PRICAI 2014, (2014).

[Kobayashi 15A] 小林 えり，斉藤 和巳，池田 哲夫，青山 一生，服部 正嗣: “ クエリ分布を考慮した類似検索の高速化, ”第 29 回人工知能学会全国大会（JSAI2015）, (2015).

[Kobayashi 15B] 小林 えり，斉藤 和巳，池田 哲夫，青山 一生，服部 正嗣: “ クエリ分布を考慮した一般化ピボット法の距離定義による特性評価, ”第 12 回情報科学技術フォーラム（FIT2015）, (2015).

[Andoni 06] A. Andoni, M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni: “ Locality-sensitive hashing using stable distributions, ” In T. Darrell and P. Indyk and G. Shakhnarovich eds., Nearest Neighbor Methods in Learning and Vision: Theory and Practice, pp. 55-67, MIT, (2006).