4B1-R-2-8

最近傍探索のための短いハッシュの学習

Learning Short Hash for Nearest Neighbor Search

得居誠也[*1] 佐藤一誠[*2] 中川裕志[*2]
Seiya Tokui Issei Sato Hiroshi Nakagawa

[*1]東京大学大学院情報理工学系研究科
Graduate School of Information Science and Technology, The University of Tokyo

[*2]東京大学情報基盤センター
Information Technology Center, The University of Tokyo

Nearest neighbor search is a fundamental task to retrieve data points which are similar to the given query. Binary hashing is a framework of approximate methods which meets time and space requirements for a large amount of high-dimensional data points. In these methods one converts data points to r-bit binary codes by r hash functions preserving their similarity structure. Recent work shows that hash learning methods enable us to shorten binary codes and improve the precision of nearest neighbor search. However, these methods do not consider the redundant combination of more than two different hash functions. In this paper, we propose a novel hashing method to overcome this redundancy problem by transforming the optimization problem of Spectral Hashing. We show by experiments on real datasets that our method outperforms existing methods in term of code length without decreasing the nearest neighbor search precision.

1. はじめに

最近傍探索とは，データ点の集合 $x^{(1)}, \ldots, x^{(n)} \in \mathcal{X}$ の中から与えられたクエリーデータ $x \in \mathcal{X}$ に最も類似したデータ点の部分集合を見つけ出すタスクである．特に大規模かつ高次元なデータに対して，データを主記憶上で高速かつ高精度に最近傍探索を行う必要がある．

大規模データを小さく圧縮しながら最近傍探索を行う手法として，二値ハッシュの学習が有用であることが近年示されている．ハッシュの学習では，データ集合を二値ベクトル $y^{(1)}, \ldots, y^{(n)} \in \{1, -1\}^r$ に変換する．その際，元のデータ集合における距離関係が二値ベクトル間のハミング距離でおおよそ再現できるように変換することで，二値ベクトルを用いた最近傍探索を実現する．

基本的なハッシュ学習法の一つに Spectral Hashing (SH) [Weiss 09] がある．SH は拡張しやすい定式化になっており，Self-Taught Hashing [Zhang 10] や Anchor Graph Hashing [Liu 11] などの改良がある．これらの手法では，データ集合の非線形な類似構造を捉えることで，比較的短い二値ベクトルによって高精度な最近傍探索を実現する．

本研究では，SH を含め従来のハッシュ学習法に冗長性の問題があることを指摘し，この問題の解決手法を提案する．従来のハッシュ学習における冗長性に関して以下説明する．データ集合において，同じ二値ベクトルに変換される部分集合を**バケット**という．出力される n 個の二値ベクトルを成分ごとに分けたベクトル $y_1, \ldots, y_r \in \{1, -1\}^n$ を考える．従来のハッシュ学習法では，これら r 個のベクトルのそれぞれの情報量と，2 つのベクトル間の相互的な情報量とを最大化するような制約のもとで $y_1, \ldots, y_r$ を求める．ところが 3 つ以上の組み合わせにおける情報量は考慮されないため，バケットが十分に分割されない．バケット内のデータ数が多い場合，きめ細かな近傍探索が不可能となるため，従来手法では高精度な最近傍探索を行うために余計な長さの二値ベクトルが必要であった．

連絡先: 得居 誠也，tokui@r.dl.itc.u-tokyo.ac.jp

実際に SH がバケットの分割に失敗する例を 1 に示した．これは 2 区間 $(-3, -1], [1, 3)$ それぞれに等間隔に分布した 1 次元データに対して，$r = 3$ ビットのコードを SH および提案手法で求めた結果である．ここで各ビットは 1 では実数値で表されており，実際にはこれを正負で二値化したものを出力する．1(a) によると，SH を用いた場合にバケットが 4 つにしか分かれないことがわかる．一方 1(b) のように，提案手法では 3 ビットで区別できる最高の個数である 8 つの小さなバケットが生成されている．

本論文の以降の構成は以下の通りである．まず第 2 節で，提案手法の基礎ともなる Spectral Hashing の定式化を述べる．第 3 節では提案手法について説明する．第 4 節では実データを用いた実験結果を紹介し，従来手法に比べて提案手法がより短い二値ベクトルで高精度な近傍探索を実現することを示す．最後に第 5 節で本研究の結論を述べる．

2. Spectral Hashing

提案手法の基本にもなっている Spectral Hashing について定式化を述べておく．SH では，データ集合の 2 点間の類似度を要素とする行列 $A \in \mathbb{R}^{n \times n}$ を用いて次の最適化問題を解く．

$$\min_Y \frac{1}{2} \sum_{ij} \left\| y^{(i)} - y^{(j)} \right\|^2 A_{ij} = \mathrm{tr}(Y^\top L Y) \qquad (1)$$
$$\text{s.t. } Y^\top \mathbf{1} = 0,\ Y^\top Y = nI.$$

ここで $Y = (y^{(1)}, \ldots, y^{(n)})^\top$ は出力となる実行列であり，その成分を正負で二値化したものの各行が変換後の二値ベクトルとなる．また，L は A を重み行列とするグラフのグラフラプラシアンであり，A の行和を並べた対角行列 D を用いて $L = D - A$ と定義される．$\mathbf{1}$ はすべての成分が 1 であるようなベクトルである．最適化問題 (1) の目的関数は，類似したデータ対が近い二値ベクトルに変換されるように作られている．$Y = (y_1, \ldots, y_r)$ のようにビットごとの列で分けて考えると，**バランス制約** $Y^\top \mathbf{1} = 0$ は各ビットの情報量を最大化し，

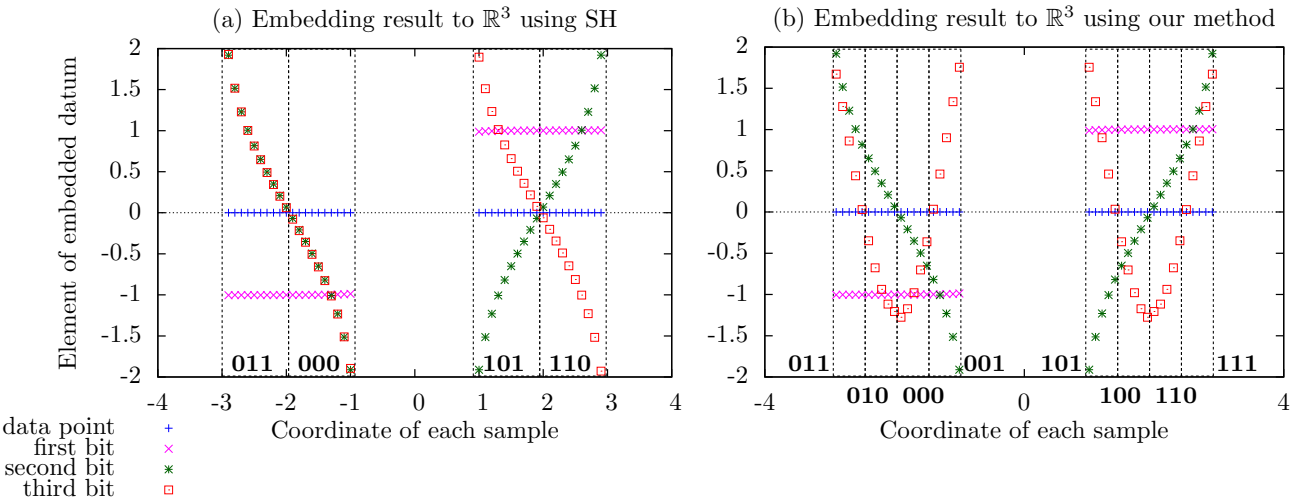

図 1: 1 次元人工データ (青) を SH (a) と提案手法 (b) を用いて $\mathbb{R}^3$ に埋め込んだ結果．正および負の成分はそれぞれ $1, -1$ に変換される．縦長の箱がバケットを表す．(a) ではデータは 4 つのバケットにしか分かれない．一方，(b) では提案手法によりデータが 8 つの小さなバケットに分割されていることがわかる．

直交制約 $Y^\top Y = nI$ は 2 つのビット間の相互的な情報量を最大化するために用いている．(1) の大域的最適解は，L の固有値 $0 = \sigma_0 \leq \sigma_1 \leq \cdots \leq \sigma_{n-1}$ に対応する固有ベクトルを $\mathbf{1} = u_0, u_1, \ldots, u_{n-1}$ とおいたとき，$Y = \sqrt{n}(u_1, \ldots, u_r)$ となる．

最適化問題 (1) を解いて得られたベクトル $y_1, \ldots, y_r$ に対して，これらを正解ラベルとして r 個のサポートベクターマシン (SVM) を学習することで，未知のクエリーに適用可能なハッシュ関数を得ることができる．この枠組みは Self-Taught Hashing (STH) [Zhang 10] として知られている．本稿で紹介する提案手法でもこの枠組みを用いる．

SH における類似度行列 A の選択肢として二つ紹介しておく．一つは重み付き k 近傍グラフの重み行列 [Zhang 10] である．もう一つはアンカーグラフの類似度行列 [Liu 11] である．特に後者を用いたアルゴリズムを Anchor Graph Hashing (AGH) という．どちらも疎行列であり，Lanczos 法 [Watkins 10] などを用いることでグラフラプラシアンの $r+1$ 番目までの固有ベクトルを $O(kn)$ の時間，空間計算量で求めることができる．特にアンカーグラフについてはグラフの構築もデータ数 n に対して線形の計算量で行うことができる．

3. 提案手法

節 1 節で述べたように，SH やその拡張である STH，AGH にはバケットが十分に分割されない問題がある．この問題は，SH の最適化問題 (1) がベクトル $y_1, \ldots, y_r$ のうち 1 つおよび 2 つの組み合わせに対してしか制約を置かないことに起因する．直交制約はどの 2 つのビットも大域的には異なる分布を持つことを保証するが，バケットはすべてのビットの組み合わせによって決定されるため，これを細かく分割するためには 3 つ以上のビットの組み合わせも考慮した制約が必要となる．提案手法では $y_1, \ldots, y_r$ を 1 つずつ求める．その際，すでに決定されたビットの組み合わせをすべて考慮し，各バケットを確実に分割するような制約を入れることで，細かいバケット分割を達成する．本節ではバケットを分割しつつ SH と同様の目的関数を最小化するような最適化問題を定式化し，従来手法とほとんど同じ計算量でこれを解く手法を提案する．

3.1 局所最適化ハッシュ

$y_1, \ldots, y_{k-1}$ が求まった状態で $y = y_k$ を最適化することを考える．この時点での各バケット S に対して，その指示ベクトル $\mathbf{1}_S = (i \in S \text{ ならば } 1, \text{そうでなければ } 0)_i$ を 2 ノルムで正規化したものを**バケットベクトル**と呼ぶ．バケットベクトルを並べた行列を $B \in \mathbb{R}^{n \times b}$ とおく．ここで b はこの時点でのバケットの個数である．y が各バケットを分割するには，**バケット分割制約** $B'y = 0$ が成り立てば良い．そこで y を以下の最適化問題の解として求める．

$$\min_y y^\top L y \quad \text{s.t. } \|y\|^2 = n,\ B^\top y = 0. \tag{2}$$

最適化問題 (2) は次のように固有値問題に帰着される．$\hat{A} = I - L = D^{-1/2} A D^{-1/2}$ とおく．このとき $y^\top L y = \|y\|^2 - y^\top \hat{A} y = n - y^\top \hat{A} y$ が成り立つので，正規化制約下での $y^\top L y$ の最小化は $y^\top \hat{A} y$ の最大化と等価である．さらにバケット分割制約 $B^\top y = 0$ を用いて目的関数を

$$y^\top \hat{A} y = y^\top (I - BB^\top) \hat{A} (I - BB^\top) y$$

と変形できる．ここで $I - BB^\top$ は $B^\top$ の核空間への射影行列であることに注意する．最終的に最適化問題は次のように変形される．

$$\max_y y^\top (I - BB^\top) \hat{A} (I - BB^\top) y \quad \text{s.t. } \|y\|^2 = n. \tag{3}$$

バケット分割制約 $B^\top y = 0$ はもはや必要ない．なぜならば，この制約を満たさない y は (3) に現れる射影行列の掛け算においてノルムが減少してしまい，目的関数の最適解になり得ないので，(3) の最適解は自らバケット分割制約を満たすからである．最適化問題 (3) の最適解は $(I - BB^\top)\hat{A}(I - BB^\top)$ の最初の固有ベクトルであり，Lanczos 法を用いれば A の非ゼロ要素数に線形な計算量で求めることができる．このアルゴリズムを局所最適化ハッシュという．

3.2 弱い分割

以上の方法で逐次的に $y_1, \ldots, y_r$ を求める場合，途中でバケットが完全に部活された場合にそれ以上ビットを増やせないという問題がある．このとき行列 B は置換行列となり，制約 $B^\top y = 0$ を満たす y は零ベクトルのみとなってしまう．この場合，提案手法が短い二値ベクトルでバケットを細かく分割できていることになるが，少しビット長が長くても良いから精度を上げたいという場合もある．以下ではコード長と精度のトレードオフを調整するための提案手法の拡張を紹介する．

局所最適化ハッシュでは，1 ビットずつ最適化問題 (2) を解いてハッシュ値を決定していった．解は固有ベクトル問題 (3) を解くことで得られた．ここで 2 番目以降の固有ベクトルは，局所的には最初の固有ベクトルと同じ割り当てを行う可能性があるが，大域的には直交する．そこで 2 番目以降の固有ベクトルも一部使うことで，ビット数を増やす代わりに大域的な近傍構造をもとのデータ集合により近づけることができる．局所最適化ハッシュの各ステップにおいて $(I - BB^\top)\hat{A}(I - BB^\top)$ の上位 q 個の固有ベクトルを用いる場合，これは次の最適化問題を解いていることになる．

$$\min_{Y_k} \operatorname{tr}(Y_k^\top L Y_k) \qquad (4)$$
$$\text{s.t. } Y_k^\top Y_k = nI,\ B^\top Y_k = O.$$

ここで $Y_k = (y_{(k-1)q+1}, \ldots, y_{kq-1})$ は k 番目のステップで求めるハッシュ値の行列である．最適化問題 (4) では一度に求めるビットの間に直交制約を置いている．$q = r$ と取った場合には SH と等価な問題となり，(4) は SH と局所最適化ハッシュの中間的な問題と考えることができる．

4. 実験による評価

提案手法である局所最適化ハッシュと従来手法である STH および AGH とを実験により比較する．提案手法における類似度行列としては k 近傍グラフとアンカーグラフを用いてそれぞれ実験を行った．これらは LOH および ALOH という名前で結果を表示した．LOH および STH では線形 SVM を用いてハッシュ関数を学習した．AGH では [Liu 11] で提案されている手法でハッシュ関数を学習し，ALOH ではアンカーとの類似度を特徴とした線形 SVM を学習した．AGH については [Liu 11] にある拡張である AGH-2 も比較した．

評価指標として，訓練データにおけるバケットの平均データ数と，最近傍探索精度を用いた．後者について，各クエリーに対して訓練データにおける 100 近傍を正解データとして，ビット列を用いた場合の 10 近傍における正解データの割合を計算し，全クエリーに対する平均を精度として用いた．

データセットとして **MNIST** と **RCV1** を用いた．**MNIST**[*1] は数字画像認識のデータセットで，各データは 784 次元のグレースケールのピクセル値で表されている．これを 69,000 個の訓練データと 1,000 個のテストデータに分割して実験を行った．**RCV1** [Lewis 04] は文書分類のデータセットで，各データは 47,236 次元の疎な Bag-of-words ベクトルで表されている．これを 518,571 個の訓練データと 15,564 個のテストデータに分割して実験を行った．

各データセットに対する実験結果を 2，3 に示した．**MNIST** は密な特徴ベクトルからなり，非線形な類似度構造を持っているため，単純な線形 SVM を用いる LOH や STH と比べて

∗1 http://yann.lecun.com/exdb/mnist/

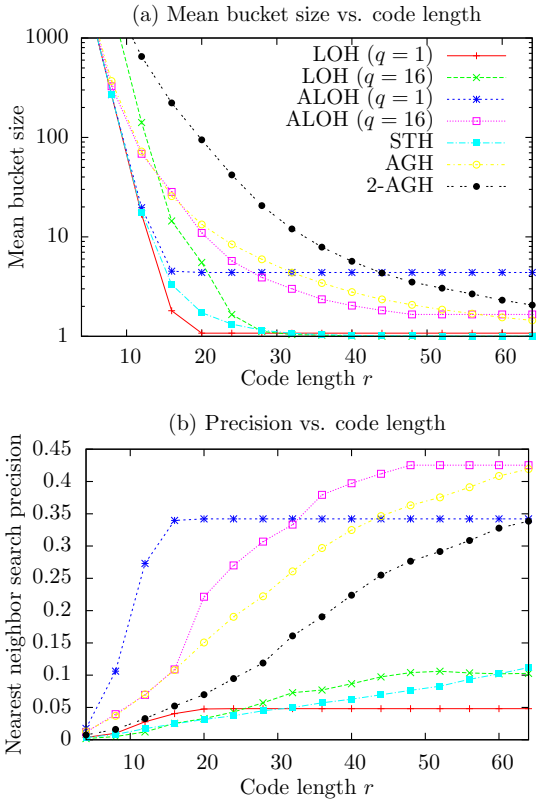

図 2: **MNIST** に対する実験結果．(a) バケット内の平均データ数．(b) 上位 10 データの近傍探索精度．

アンカーを利用した ALOH や AGH の方が精度が高かった．一方，**RCV1** は疎な特徴ベクトルからなるため，LOH が高い精度を達成している．特に **MNIST** における ALOH と **RCV1** における LOH は，非常に短い二値ベクトルで小さなバケットと高い近傍探索精度を達成している．弱い分割法である $q = 16$ の設定においては，LOH，ALOH 共に非常に小さなビット数では精度が落ちるものの，ビット数を中程度に増やしたときの精度は $q = 1$ の場合に勝っており，トレードオフの調節に成功している．

5. おわりに

本稿では省メモリな大規模最近傍探索を実現するためのハッシュ学習法について，Spectral Hashing を中心とした従来の手法ではバケットが十分に分割されない問題があることを指摘した．提案した局所最適化ハッシュは，明示的なバケット分割制約を導入することでこの問題に対処している．実際に画像と文書のデータセットに適用することで，局所最適化ハッシュが非常に短い二値ベクトルを用いて高精度な最近某探索を実現することを示した．また，局所最適化ハッシュの分割を緩やかにすることで，コード長と精度のトレードオフを調節することができることを示した．今後の課題として，そのままではメモリ上で処理できないような大規模データセットでの実験を行いたいと考えている．

参考文献

[Lewis 04] Lewis, D. D., Y., Y., Rose, T. G., and Li, F.: RCV1: A New Benchmark Collection for Text Cate-

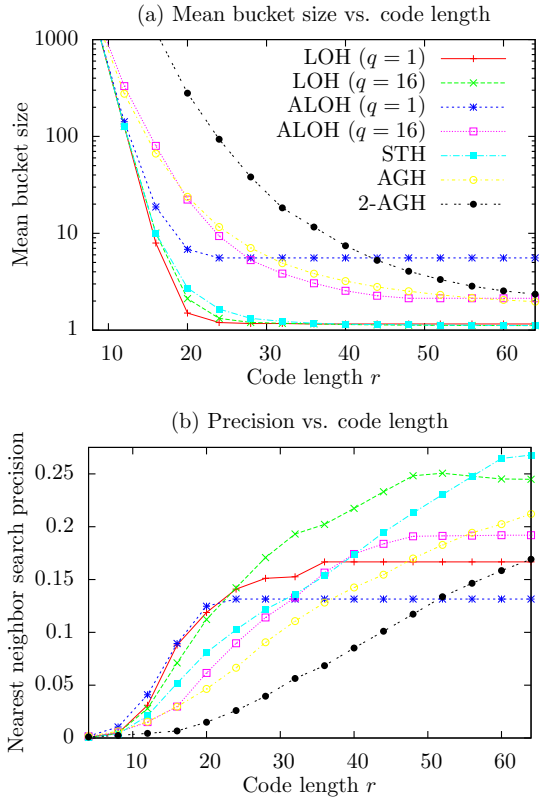

図 3: **RCV1** に対する実験結果．(a) バケット内の平均データ数．(b) 上位 10 データの近傍探索精度．

gorization Research, *Journal of Machine Learning Research*, Vol. 5, pp. 361–397 (2004)

[Liu 11] Liu, W., Wang, J., Kumar, S., and Chang, S.-F.: Hashing with Graphs, in *Proceedings of the 28th International Conference on Machine Learning*, pp. 1–8 (2011)

[Watkins 10] Watkins, D. S.: *Fundamentals of Matrix Computations*, Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts, Wiley, third edition edition (2010)

[Weiss 09] Weiss, Y., Torralba, A., and Fergus, R.: Spectral Hashing, in *Advances in Neural Information Processing Systems 21*, pp. 1753–1760 (2009)

[Zhang 10] Zhang, D., Wang, J., Cai, D., and Lu, J.: Self-Taught Hashing for Fast Similarity Search, in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 18–25 (2010)