

複数コードブックを用いた直積量子化による近似最近傍探索手法と 特定物体認識への応用

内田 祐介[†] 酒澤 茂之[†]

[†] 株式会社 KDDI 研究所

〒 356-8502 埼玉県ふじみ野市大原 2-1-15

E-mail: [†]{ys-uchida,sakazawa}@kddilabs.jp

あらまし 本研究では、大規模なマルチメディアコンテンツの検索や認識を目的とした、高次元特徴ベクトルの近似最近傍探索手法を扱う。直積量子化を利用した近似最近傍探索手法が注目されているが、従来手法では、異なる分布を持つベクトル集合に対して画一的なコードブックで直積量子化を行っていたため、符号化効率が悪く近似最近傍探索の精度が低下する問題があった。これに対し、任意の数の最適化されたコードブックを作成し、分布に応じて利用するコードブックを切り替えることで、上記の問題を解決する手法を提案する。また、提案手法を特定物体認識に応用した際の考察によって、Bag-of-Visual Words に基づく特定物体認識において、特徴ベクトル間の推定距離によってマッチングを高精度化する際には、Visual Word に属する特徴ベクトル数に対する比率を利用したフィルタリングが有効であることを実験的に示す。

キーワード 近似最近傍探索、ベクトル量子化、直積量子化、クラスタリング、類似画像検索、特定物体認識

1. はじめに

大規模なマルチメディアコンテンツの検索や認識を実現するため、大量の高次元特徴ベクトルに対する効率的な最近傍探索手法の検討が重要な研究課題の 1 つとなっている。最近傍探索手法はこれまで多くの検討がなされており、例えば、二分探索木上で枝刈りを用いた探索を行う kd-tree [1] は低次元ベクトルを対象とした際の最良の選択肢の一つである。また、厳密な最近傍を求めず、確率的に誤りを許すことで効率化を行う近似最近傍探索手法も多く提案されている。ANN [2] では kd-tree 上で誤差を許して枝刈りを行うことで精度と計算量のトレードオフを実現している。しかしながら、上記の手法は、特定物体認識や一般物体認識で広く利用されている SIFT [3] や GIST [4] といった高次元特徴ベクトルを対象とすると、効率的な枝刈りが行えず線形探索と同等程度の計算量が必要になってしまう次元の呪いと呼ばれる問題が発生する。これに対し、複数のハッシュ関数を利用した探索を行う LSH [5] と呼ばれる手法が、高次元特徴ベクトルに対しても応用可能であり、更に精度と計算量の関係が理論的に解析されているため注目を集めている。また、特定物体認識の分野では、理論的な解析はあまり行われていないものの、randomized kd-trees [6, 7] や hierarchical k-means tree [8] 上で priority search [9] を行う探索手法が LSH と比較して経験的に優れた精度を達成することが報告されている [6, 7, 10]。上記の手法を組み合わせ、パラメータを自動的に設定する FLANN と呼ばれる手法も提案されている [10]。

近似最近傍探索を大規模データに適用する場合、精度

と計算量だけでなくメモリ使用量も考慮する必要がある。前述の手法は、特徴ベクトルそのものをメモリ上に保持することを前提としているため、メモリ制約から大規模データに適用することが出来ない場合がある [11]。これを解決するため、高次元ベクトルをよりコンパクトな符号に符号化し、符号間の距離に基づいて近似最近傍探索を行う手法が提案されている。代表的な手法として、ランダム射影を利用する LSH [12] や Hamming Embedding (HE) [13]、主成分分析を用いる Spectral Hashing (SH) [14]、主成分分析後に最適なビット割り当てを行う手法 [15]、直積量子化を用いる手法 [11] がある。これらの中では、直積量子化を用いる手法が精度と符号長（メモリ使用量）のトレードオフの観点から最も優れていることが報告されている [11, 15]。また、精度と計算量のトレードオフにおいても LSH や FLANN と同等以上の性能を示すことが報告されている [11]。

本稿では、直積量子化を用いた近似最近傍探索に着目し、転置インデックスと直積量子化を組み合わせる場合に、分布の異なるベクトル集合に対し同一の直積量子化コードブックを用いて符号化を行うことによる符号化効率の低下問題を指摘する。上記の問題を解決するため、直積量子化時に複数の直積量子化コードブックを切り替えて利用することを提案し、分布の異なるベクトル集合の集合を複数の直積量子化コードブックを介してクラスタリングし、任意の数の最適化された直積量子化コードブックを作成する手法を提案する。また、直積量子化のような特徴ベクトル間の距離を推定する近似最近傍探索手法を、Bag-of-Visual Words (BoVW) [16] に基づく特定物体認識に応用する際の考察を通じて、Visual Word に

表1 本稿で用いる表記

D	特徴ベクトルの次元
N	粗量子化コードブックのサイズ
$C = \{\mathbf{c}_n\}_{n=1}^N$	粗量子化コードブック ($\mathbf{c}_n \in \mathbb{R}^D$)
S	部分ベクトルの分割数
M	直積量子化コードブックの数
L	直積量子化コードブックのサイズ
$\mathcal{D}_m = \{\mathbf{d}_{m,l}\}_{l=1}^L$	直積量子化コードブック ($\mathbf{d}_{m,l} \in \mathbb{R}^{D/S}$)

属する特徴ベクトル数を考慮することにより高精度化が実現できることを示す。

2. 直積量子化を用いた近似最近傍探索

本節では、従来手法である直積量子化を利用した最近傍探索手法 [11] およびその課題について概説する。特に、特定物体認識と相性の良い、直積量子化と転置インデックスと組み合わせることで全探索を回避し効率的な探索が行える手法（[11] では IVFADC と参照）に焦点を当てる。本稿で利用する表記を表1に示す。

2.1 直積量子化による符号化と距離推定

直積量子化とは、 D 次元のベクトルを S 個の部分ベクトルに分割し、これらを個別に量子化する手法である。本稿では、 $D \equiv 0 \pmod{S}$ を仮定し、先頭から D/S 次元毎に分割するものとする。 $S = 1$ のときはベクトル量子化、 $S = D$ のときはスカラ量子化と等価となる。ベクトル $\mathbf{x}$ の s 番目の部分ベクトル $\mathbf{x}_s$ を、サイズ L の直積量子化コードブック $\mathcal{D}_s$ で量子化した結果を $l_s \in \{1, 2, \dots, L\}$ とする。この場合、ベクトル $\mathbf{x}$ は $\{l_s\}_{s=1}^S$ と符号化され、この符号は $S \log_2 L$ bit で表現される。[11] では、検索対象であるリファレンスベクトルを直積量子化で符号化しておき、クエリベクトルとリファレンスベクトルの距離の推定値として、クエリベクトルとリファレンスベクトルの符号の距離を計算することを提案している。クエリベクトル $\mathbf{y}$ と符号 $\{l_s\}_{s=1}^S$ の距離を d とすると、 $d^2 = \sum_{s=1}^S \|\mathbf{y}_s - \mathbf{d}_{s,l_s}\|^2$ である。この手法のポイントは、クエリベクトル $\mathbf{y}$ が与えられた際に、部分ベクトル $\mathbf{y}_s$ と対応する直積量子化コードブック $\mathcal{D}_s$ の各符号語 $\mathbf{d}_{s,l}$ との距離 $\|\mathbf{y}_s - \mathbf{d}_{s,l}\|^2$ を予め全て計算し、ルックアップテーブル $T[s][l]$ に保持しておくことで、リファレンスベクトルの各符号語との距離計算をテーブルの参照と加算で効率的に行える点にある。このテーブルを利用すると、前述の推定距離は $d^2 = \sum_{s=1}^S T[s][l_s]$ と求められる。

2.2 転置インデックスとの組み合わせ

前述したように、直積量子化に基づく最近傍探索手法は、転置インデックスと組み合わせることでより効率的な探索を行うことができる [11]。転置インデックスを利用する際の処理手順を図1に示す。コードブックを切り替える処理は後述する提案手法で導入する処理である。符号化時には、リファレンスベクトル $\mathbf{x}$ はまず、サイズ

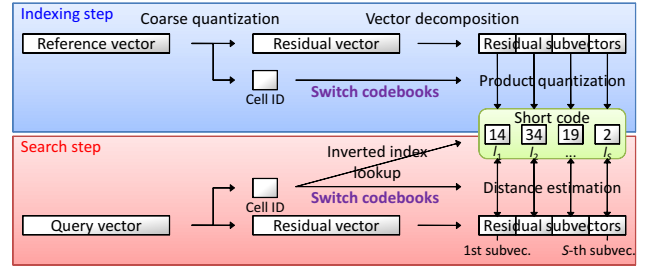


図1 転置インデックスと直積量子化を用いた近似最近傍探索

N の粗量子化コードブック C を用いてベクトル量子化される。ベクトル量子化によってリファレンスベクトル $\mathbf{x}$ が符号語 $\mathbf{c}_n$ に割り当てられたとすると、 $\mathbf{c}_n$ からの残差ベクトル $\mathbf{r} = \mathbf{x} - \mathbf{c}_n$ を求める。この残差ベクトルを部分残差ベクトル（以降単に部分ベクトル）に分割し、前述の直積量子化により符号化を行う。符号化された $\mathbf{x}$ の符号は、符号語 $\mathbf{c}_n$ に対応する転置インデックスの n 番目のリストにベクトルの識別子とともに登録される。探索時には、クエリベクトルはリファレンスベクトルと同様に、粗量子化コードブックでベクトル量子化される。その後、対応する転置インデックスのリストに登録されている符号とのみ距離計算を行うことで全探索を回避することができる。

転置インデックスと直積量子化を組み合わせた手法は効率的な探索を行うことができる一方、符号化を行う直積量子化コードブックに下記のような問題がある。高次元ベクトルを対象とする場合、粗量子化器の各ボロノイセル（以降、単にセル）の形状がお互いに大きく異なるため、セルの代表ベクトルからの差分ベクトルの分布も異なり、その部分ベクトルの分布も異なると考えられる。すなわち、部分ベクトルの分布は粗量子化時に割り当てられるセルに大きく依存する。従来は、部分ベクトルの位置ごとに、全てのセルの部分ベクトルをまとめてクラスタリングし、直積量子化コードブックを作成していた。結果として、全ての分布を包含するような直積量子化コードブックが作成され、符号化効率が低下するという問題が発生する。これに対処するため、各セル、各部分ベクトルの位置ごとに直積量子化コードブックを作成することも考えられるが、直積量子化コードブックを保持するために $N \times S$ に比例したメモリが必要となり、比較的大きな N （例えば [6] では $10K \sim 1M$ ）を利用する大規模な特定物体認識等では利用できない。例えば、 $N=100K$, $L=256$ とした 128 次元の SIFT 特徴ベクトル用の直積量子化コードブックの場合、 $100K \times 256 \times 128 \sim 3G$ Byte のメモリを必要とすることになる。この問題を解決するため、本稿では各セル、各部分ベクトルに対応する部分ベクトル集合を任意の M 個のクラスタにクラスタリングし、各クラスタからそれぞれ 1 つの直積量子化コードブックを作成し、セルおよび部分ベクトルの位置によって利用する直積量子化コードブックを切り替える

手法を提案する．提案手法において $M \ll N \times S$ となるような M を利用することで，現実的なメモリ使用量で複数の直積量子化コードブックを利用しつつ，符号化効率を改善することができ，最終的な近似最近傍探索の精度を改善することができる．既存手法と提案手法の直積量子化コードブック作成法の違いを図2に示す．

3. 複数コードブックを用いた直積量子化

上記の問題を解決するため，本稿では任意の M 個の直積量子化コードブックを利用した近似最近傍探索手法を提案する．直積量子化による符号化処理および探索処理における提案手法と既存手法の違いは，図1に示すように，粗量子化時のセルと部分ベクトルの位置によって直積量子化コードブックを切り替える部分のみである．基本的に，直積量子化コードブックの数は提案手法のほうが多くなるが，探索時のルックアップテーブルを作成する処理は，各部分ベクトルが対応する直積量子化コードブックのみについてテーブルを作成するため，探索時に必要な計算量は既存手法と同一となる．以下では，提案する直積量子化コードブックの作成法を記述する．

3.1 複数直積量子化コードブックの作成法

複数の直積量子化コードブックの作成法を説明する．まず，訓練ベクトル集合をサイズ N の粗量子化器で量子化し残差ベクトルを求め，残差ベクトルを S 個の部分ベクトルに分割することで，部分ベクトル集合の集合 $\{\mathcal{R}_{s,n}\}$ を作成する．ここで $\mathcal{R}_{s,n}$ は，粗量子化器においてセル n に割り当てられた訓練ベクトルの残差ベクトルの， s 番目の部分ベクトルの集合である．この $N \times S$ 個の部分ベクトルの集合を，類似した分布を持つ M 個のクラスタにクラスタリングし，それぞれから直積量子化コードブックを作成することが課題となる．部分ベクトルの分布を仮定し，分布間の距離によってクラスタリングを行うことも考えられるが，量子化誤差を軽減するという明らかな目的関数があるため，本稿では k -means のように，繰り返し処理によってクラスタリングおよび直積量子化コードブックの作成を行うことを提案する．具体的には，下記の手順となる．

[Step 1] 全ての部分ベクトル集合 $\mathcal{R}_{s,n}$ に対し，ラベル $\hat{m}_{s,n} \in \{1, 2, \dots, M\}$ を付加する初期化を行う．

[Step 2] $m = 1, 2, \dots, M$ について，ラベル m が付加された部分ベクトル集合の和集合 $\bigcup_{\hat{m}_{s,n}=m} \mathcal{R}_{s,n}$ に属する部分ベクトル全てを用いてクラスタリングを行い，直積量子化コードブック $\mathcal{D}_m$ を作成する．

[Step 3] 各部分ベクトル集合 $\mathcal{R}_{s,n}$ について， M 個の直積量子化コードブック $\mathcal{D}_m$ それぞれで $\mathcal{R}_{s,n}$ に属する部分ベクトル全てを量子化した際，量子化誤差の合計が最も小さくなる m でラベル $\hat{m}_{s,n}$ を更新する．

[Step 4] 収束するまで Step 2 と Step 3 を繰り返す．

上記により， M 個の直積量子化コードブック $\{\mathcal{D}_m\}_{m=1}^M$ が作成される．また，最終的に得られるラベル $\hat{m}_{s,n}$ は，粗量子化器においてセル n に割り当てられたベクトルの差分ベクトルの s 番目の部分ベクトルは， $\hat{m}_{s,n}$ 番目の直積量子化コードブックで符号化されるという情報であり，リファレンスベクトルの符号化時および探索時に直積量子化コードブックを切り替えるために利用される．

3.2 初期化手法

本稿では，直積量子化コードブックセット作成時の Step 1 において，2 種類のラベルの初期化手法を検討する．1 つは，単純に各差分部分ベクトル集合に対しランダムなラベル $m \in \{1, \dots, M\}$ を与える手法である．もう1 つは， k -means++ アルゴリズム [17] を参考とした初期化手法である． k -means++ アルゴリズムでは，各入力サンプルベクトルについて，これまで選択された代表ベクトル集合のうち，最も近い代表ベクトルまでの距離の二乗に比例する確率で次の代表ベクトルを選択する^(注1)という処理を，所望のクラスタ数だけ代表ベクトルが選択されるまで繰り返し行う．その後，全てのサンプルベクトルについて，最も距離の近い代表ベクトルのラベルを付加することで初期化を行う．この k -means++ アルゴリズムを，複数の直積量子化コードブック作成時の初期化に応用すると，下記の手順となる．

[Step 1] ランダムに部分ベクトル集合を1つ選択し，クラスタリングすることで直積量子化コードブックを1つ作成する．

[Step 2] 各部分ベクトル集合について，これまで作成された直積量子化コードブック全てで量子化を行い，それらのうち最小となる量子化誤差を求める．

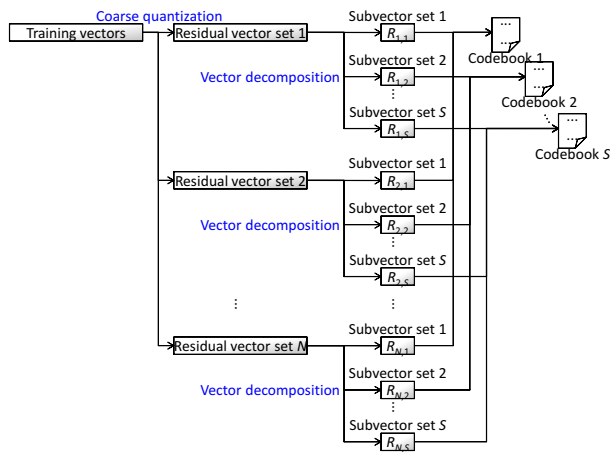
[Step 3] Step 2 で求めた各部分ベクトル集合の最小量子化誤差に比例する確率で部分ベクトル集合を1つ選択し，クラスタリングを行うことで新たな直積量子化コードブックを作成する．

[Step 4] M 個の直積量子化コードブックが作成されるまで Step 2 と Step 3 を繰り返す．

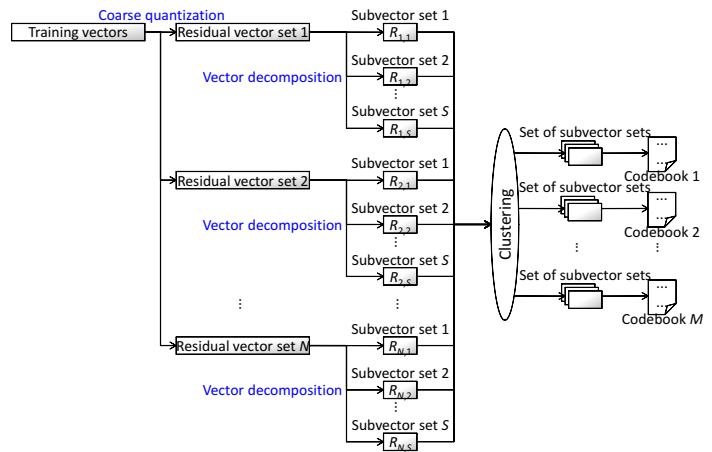
[Step 5] 全ての差分ベクトル集合について，量子化を行った際に最も量子化誤差が小さくなる直積量子化コードブックのラベルを付加する．

なお，Step 2 で求める最小の量子化誤差は，1 つ前の繰り返しでの最小値を保持しておき，新たに作成された直積量子化コードブックでの量子化誤差と比較することで効率的に算出できる．

(注1)：正確には，距離の二乗に比例した確率で代表ベクトルを選択することを複数回繰り返し，全体の量子化誤差を最小にする代表ベクトルを選択するという処理を行なっている．



(a) 既存手法における直積量子化コードブック作成法



(b) 提案手法における直積量子化コードブック作成法

図2 既存手法および提案手法での直積量子化コードブック作成法

4. 評価実験

4.1 データセットおよび実験環境

提案手法の精度検証および既存手法 [11] との比較のため、下記の公開データセットを用いて評価実験を行う。

- ANN.SIFT1M^(注2)
- Flickr60K^(注3)

ANN.SIFT1Mからは100万個のリファレンス（テスト）ベクトルおよび1万個のクエリベクトルを利用し、Flickr60Kからはコードブック作成時の訓練ベクトルを最大400万個利用する。ANN.SIFT1Mにも訓練ベクトルは含まれているが、サイズが小さいため上記のデータを利用する。実験には、OS: Windows 7 64-bit, CPU: Core i7-940 2.93GHz, Memory: 24G Byte の計算機を利用する。

本節の実験では、特徴ベクトルとして128次元のSIFTを利用し、[11]で妥当なパラメータとされている、粗量子化コードブックサイズ $N = 1024$ 、部分ベクトルの分割数 $S = 8$ 、直積量子化コードブックサイズ $L = 256$ を利用する。この場合、直積量子化によって生成される符号長は $S \log_2 L = 64$ bit となる。粗量子化コードブックは、提案手法および既存手法ともに、400万個の訓練ベクトルから作成した同一のものを利用し、以降では直積量子化コードブックを単にコードブックと参照する。なお、SIFTは特徴領域を 4×4 のブロックに分割し、それぞれから8次元の方向ヒストグラムを作成しており、部分ベクトルの分割数 $S = 8$ は、2つのブロックのヒストグラムを1つの部分ベクトルとして分割していることに相当する。

4.2 初期化法の評価

本節では、3.2節で提案した2種類の初期化法について比較評価を行う。実験では、 $M = 64$ とし、訓練ベクトル

400万個からコードブックを作成する。図3にランダムにラベルを与える初期化法 (base) と k -means++アルゴリズムを参考とした初期化法 (k -means++) について、繰り返し数を20回としてコードブックを作成することを10回行い、各ステップまでに必要とした処理時間と各ステップ終了時の量子化誤差の平均 (ave) と最小値 (min) を計測した結果を示す。ここで量子化誤差は、直積量子化を用いてベクトル間の距離を推定した際の誤差の上界となっている [11]。

図3より、 k -means++を参考とした初期化法では、 k -means++と同様に初期化に処理時間が必要であるものの、ランダムにラベルを与える初期化法と比較して収束が早く、収束時の量子化誤差も低くなるのが分かる。 k -means++を参考とした初期化法は M に比例した処理時間を必要とする。各繰り返しでより時間を必要とする処理は、部分ベクトル集合を最適な直積量子化コードブックに割り当てる処理であり、これも直積量子化コードブック数 M に比例する時間を必要とする。そのため、どの M においても初期化と繰り返しの必要な処理時間の比はほぼ一定となり、どの M でも k -means++を参考とした初期化法が有効であると予想される。以降の実験では、全てのコードブックは k -means++を参考とした初期化法を用いて作成したものを利用する。

4.3 異なる特性を持つベクトルを用いた考察

本節では、部分ベクトルの分布が分割された部分ベクトルの位置 s に (a) 依存する特徴ベクトル、(b) 依存しない特徴ベクトル、それぞれについて提案手法の有効性を検証する。(a)の例としてHessian-Affine [18] 特徴領域をSIFTで記述した特徴ベクトル (HesAff-SIFT)、(b)の例として画像をグリッド分割しSIFTで記述 [19] した特徴ベクトル (Grid-SIFT) を利用する。HesAff-SIFTは、(1) Hessian-Affineにより検出される領域がBlob領域である、(2) 特徴点を中心とするガウス窓によって各画素の特徴ベクトルへの貢献に重みが付けられている、という2つ

(注2) : <http://corpus-texmex.iris.fr/>

(注3) : <http://lear.inrialpes.fr/~jegou/data.php>

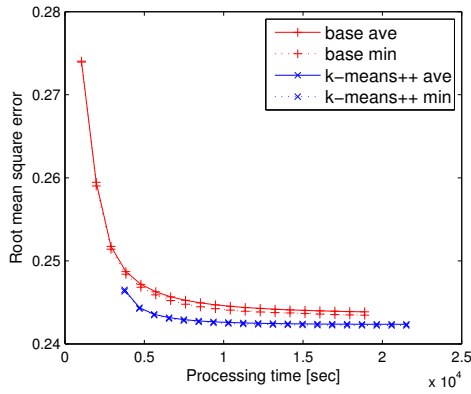


図3 初期化手法の比較

の理由から、部分ベクトルの分布が部分ベクトルの位置によって異なる。他方、Grid-SIFT は理論的には全ての部分ベクトルの分布は同一^(注4)である。

表2に、HesAff-SIFTとGrid-SIFTそれぞれ100万個のベクトルを用いて、提案手法で $M=1,8$ としてコードブックを作成した場合(Prop.)と、既存手法(Conv.)で作成した場合の、量子化誤差を示す。提案手法 $M=1$ では、全ての部分ベクトル集合から1つのコードブックを作成し、既存手法では、部分ベクトルの位置それぞれに対し1つ、合計8つのコードブックを作成することになる。提案手法 $M=8$ は、部分ベクトルの位置によらず全ての部分ベクトル集合を8つのクラスタにクラスタリングし、それぞれのクラスタから、合計8つのコードブックを作成することになる。表2において提案手法 $M=1$ と既存手法を比較すると、予想されるように、位置によって部分ベクトルの分布が変わるHesAff-SIFTを対象とした場合には、部分ベクトルの位置毎にコードブックを作成することが有効であることが分かる。一方、位置によらず部分ベクトルの分布が同一であるGrid-SIFTを対象とした場合には、部分ベクトルの位置毎にコードブックを作成することは有効でないことが分かる。また、提案手法 $M=1$ と提案手法 $M=8$ を比較すると、HesAff-SIFTとGrid-SIFTどちらに対してもコードブック数を増やすことで量子化誤差を軽減できていることが分かる。特にHesAff-SIFTでは、部分ベクトルの位置毎にコードブックを作成する既存手法と比較しても、更に量子化誤差を軽減できていることから、位置によって部分ベクトルの分布が変わるような特徴ベクトルを対象としても提案手法が有効であることが分かる。

図4に、提案手法 $M=8$ について、それぞれの位置の $N=1024$ 個の部分ベクトル集合が、8つのコードブックそれぞれに割り当てられている割合を示す。HesAff-SIFTの場合は、同一位置の部分ベクトル集合が同一のコード

表2 異なる特性を持つベクトルを対象とした際の量子化誤差

	Prop. $M=1$	Conv.	Prop. $M=8$
HesAff-SIFT	0.2774	0.2715	0.2594
Grid-SIFT	0.2870	0.2852	0.2740

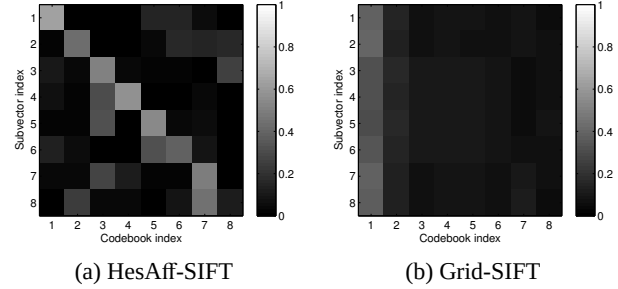


図4 各部分ベクトルの各コードブックへの割り当て率

ブックに割り当てられている割合が大きいものの、異なる位置の部分ベクトル集合もある程度コードブックを共有していることが分かる。一方、Grid-SIFTの場合は、部分ベクトルの分布が位置によらないため、同様の割り当て率となることが分かる。

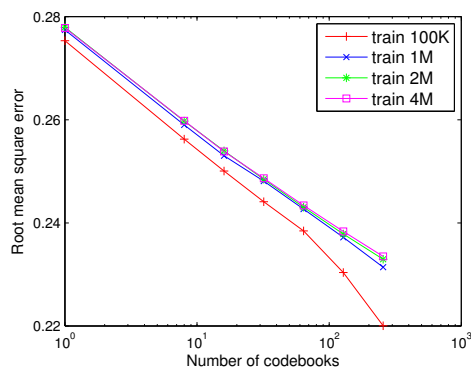
4.4 コードブック数と量子化誤差

本節では、コードブック数と量子化誤差について考察を行う。訓練データセットの先頭からそれぞれ10万個、100万個、200万個、400万個の訓練ベクトルを利用し、 $M=1,8,16,\dots,256$ としてコードブックを作成した。また、それらのコードブックを用いてテストデータ内の100万個のベクトルを量子化し、量子化誤差を計測した。 $M=1,8,16,\dots,256$ について、図5(a)にコードブック作成時の量子化誤差を、図5(b)にテストデータの量子化誤差を示す。図5より、訓練データおよびテストデータともに、コードブック数の対数に比例して量子化誤差が減少することが分かる。また、提案手法では M に比例する数の訓練ベクトルが必要であるため訓練ベクトル数が十分でない場合には、過学習が見られる。最も極端な、各セル、各部分ベクトルの位置ごとにコードブックを作成する場合では、 $N \times S = 1024 \times 8$ 個のコードブックを作成することになり、提案手法において $M=256$ とした場合の32倍の訓練ベクトルを必要とする。このことから、提案手法は精度とメモリ使用量のトレードオフだけではなく、精度と必要な訓練ベクトル数とのトレードオフも実現していると言える。

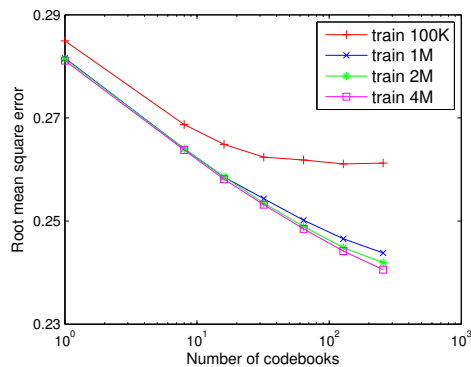
4.5 近似最近傍探索精度の検証

本節では、提案手法および既存手法について、近似最近傍探索の精度を検証する。評価指標として Recall@R [11,15]を利用する。これは、直積量子化を用いて推定されたクエリベクトルとリファレンスベクトルの距離でリファレンスベクトルをソートした際に、上位 R 件にクエリの最近傍となる正解リファレンスベク

(注4)：Grid-SIFTにおいてもガウス窓による重み付けが用いられることが多いが、本実験では検証のため重み付けを行わずに特徴ベクトルを記述している。



(a) 量子化コードブック作成時の量子化誤差



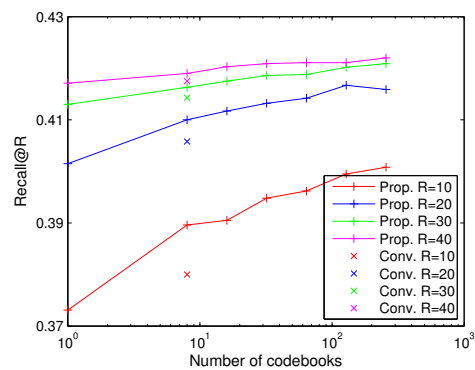
(b) テストデータの量子化誤差

図5 訓練データセットおよびテストデータセットに対する量子化誤差

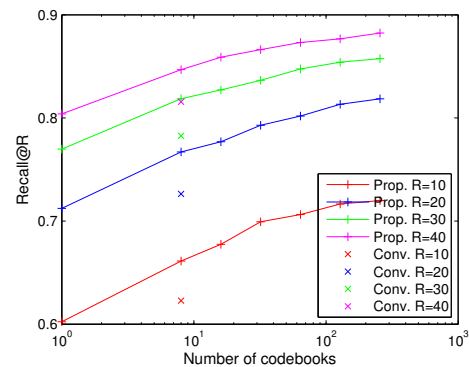
トルが含まれる割合である。ここでは、粗量子化時にクエリを最近傍の符号語に割り当ててのではなく、複数の近傍の符号語に割り当て、転置インデックスにおいて対応する複数のリストを探索することで精度を向上させる multiple assignment を導入する。割り当てられる近傍数を w とする。図6に提案手法 (Prop.) と既存手法 (Conv.) それぞれの、(a) $w = 1$ および (b) $w = 16$ における $R = 10, 20, 30, 40$ での Recall の値を示す。図6より、提案手法では、 M の対数にほぼ比例して精度が改善できていることが分かる。また、4.3節の結果からも予想されるように、同一数のコードブックを利用した場合 ($M = 8$) でも、提案手法が既存手法から大きく精度を改善できていることが分かる。例えば、 $w = 16, R = 10, M = 64$ の場合の精度は、既存手法 0.623 に対し提案手法 0.706 であり、13%の精度改善を実現している。この場合、コードブックを保持するために必要なメモリ量は 33KB から 262KB に増加するが、データベースのサイズ 12MB の 2%程度であり、データセットのサイズが大きくなれば無視できるサイズとなる。以上の考察により、提案手法は精度とメモリ使用量の優れたトレードオフを実現し、同一メモリ使用量の場合でも、既存手法に対し優位性があると言える。

5. 特定物体認識への応用

本節では、提案手法を、局所特徴を用いた特定物体認



(a) $w = 1$



(b) $w = 16$

図6 提案手法および既存手法の近似最近傍探索精度

識へ応用した際の考察を行う。ベースラインとして、[11]でも参照されており、近年注目を集めている BoVW と HE を用いた手法 [13] (以降 BoVW+HE) を取り上げる。BoVW+HE のポイントは以下の通りである。元々の BoVW [16] では同一の Visual Word (VW)^(注5) に割り当てられた局所特徴は全てマッチしたものと扱う。一方、BoVW+HE では、各局所特徴ベクトルが割り当てられた VW の代表ベクトルからの差分ベクトルをハミング空間に射影した符号語を利用し、同一の VW に割り当てられ、かつ符号語間のハミング距離が一定以下のもののみがマッチしたと扱う (フィルタリングを行う) ことで、マッチングの再現率を保ちつつ、適合率を高めている。

以降では、直積量子化を介して推定された局所特徴ベクトル間の距離を、上記のハミング距離の代わりとしてフィルタリングに利用することを検討する。近似最近傍探索における精度では、HE よりも直積量子化を用いた手法が優れていることが示されており [11]、特定物体認識においても、提案手法により精度が改善されることが期待される。

5.1 実験環境

画像データセットとして、本分野で広く利用されている Recognition Benchmark Images^(注6) データセットを利用

(注5) : VW はこれまでの表記における粗量子化の符号語に相当する。

(注6) : <http://www.vis.uky.edu/~stewe/ukbench/>

する。上記データセットは、2,550 個のオブジェクトを異なる 4 方向から撮影した画像、合計 10,200 枚から構成されている。各画像をクエリとし、オリジナルの画像を含む 4 枚の画像を正解として検索を行った際の Mean Average Precision (MAP) により性能評価を行う [13]。

実装や実験条件は [13] を踏襲し、マッチングのスコアは VW の Inverse Document Frequency (IDF) の二乗で重み付けを行い、マッチング後の各画像のスコアは BoVW の L2 ノルムで正規化されることとする。また VW の数は 20K とし、BoVW+HE で利用する符号長は直積量子化を用いる場合と同じく 64bit とした。

5.2 実験結果と考察

図 7 (a-d) に、提案手法 ($M = 256$)、既存手法 [11] について様々な基準および閾値の値を基にフィルタリングを行った際の精度 (MAP) を示す。また、図 7 (e) に、BoVW+HE の精度を示す。直積量子化を用いてフィルタリングを行う際、最も単純には、直積量子化を介して推定された距離そのものを閾値としてフィルタリングを行うことが考えられるが、予想に反して BoVW+HE よりも精度が低いという結果が得られた (図 7 (a))。これは図 8 (a) に示すように、サイズの異なるセルに対し同一の閾値でフィルタリングを行うと、小さなセルではほとんどフィルタリングが行われず、大きなセルではほとんどの特徴ベクトルがフィルタリングされてしまう、セル間の不公平性が問題であると予想される。特徴ベクトル間の距離に正規分布を仮定し、各セルにおいて平均と分散で正規化を行うことである程度改善される (図 7 (b)) が、図 9 から分かるように、セルによってベクトルの密度が違うため十分に不公平性が解消されたとは言えない。一方、[11] では、距離でソートした際の k 近傍のみマッチさせており、全てのセルにおいて同数の特徴ベクトルがマッチするためセル間の公平性は保たれており、BoVW+HE を上回る精度を達成している (図 7 (c))。しかしながら、図 8 (b) に示すように、特徴ベクトルの密度が高いセルでは、距離に換算すると非常に近い特徴ベクトルしかマッチしないことになり、ノイズ等による特徴ベクトルの変化に対応出来なくなる問題がある。本稿では、上記の考察から、距離でソートした際に一定のパーセンタイル内にあるものをマッチさせることを検討する。これは基準としてユークリッド距離を利用した場合と k 近傍を利用した場合の中間の効果があり、上記の問題が解消できると期待される。実験により、実際に精度を改善でき、マッチした特徴ベクトルについて、 p パーセンタイルにあたる特徴ベクトルに $\exp(-p^2/\sigma^2)$ ($\sigma = 0.2$) の重みを付けることで更に精度が改善できることが確認できた (図 7 (d))。どちらの場合でも提案手法は既存手法よりも優れた精度を達成しているが、特定物体認識では大量の特徴ベクトルの探索結果を統合しているため、単一ベクトルの近似最近傍探索精度で確認されたほどの精

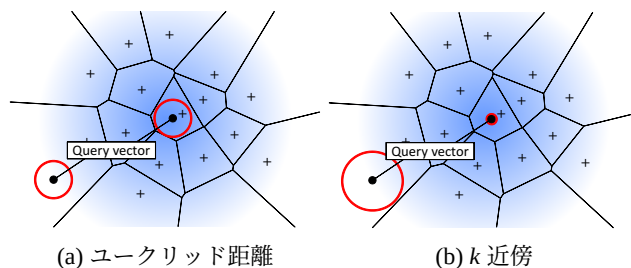


図 8 ユークリッド距離および k 近傍を利用した場合のクエリベクトルにマッチする特徴ベクトルの範囲

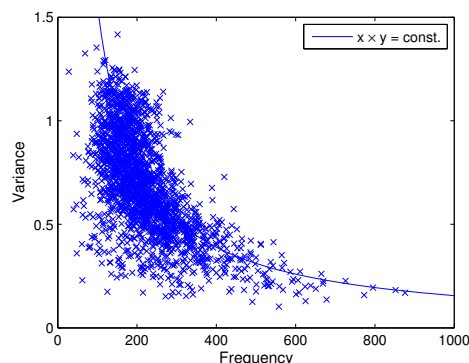


図 9 400 万個の訓練ベクトルから 20K 個の VW を作成した際の、VW に属するベクトル数 (横軸) と VW の代表ベクトルからの平均二乗誤差 (縦軸) を、ランダムに選択した 2K 個の VW についてプロットした。平均二乗誤差はだまかなセルの大きさに相当し、属するベクトル数が多い VW ほどセルが小さい傾向にあることが分かる。これは k -means クラスタリングの特性による

度改善の効果は見られなかった。

以上の考察より、BoVW を用いた特定物体認識において、特徴ベクトル間の推定距離によってマッチングを高精度化する際には、VW に属する特徴ベクトル数に対する比率を利用したフィルタリングが有効であると言える。また、このことは BoVW+HE に下記のような新たな解釈を与えることができる。BoVW+HE では、直交射影および中央値を用いた閾値処理により、特徴ベクトルを 0 と 1 が等確率かつ無相関に出現するバイナリ列に符号化している。このとき、同じ VW に属し、クエリベクトルからのハミング距離がある閾値以内である特徴ベクトルの数は、VW に属する特徴ベクトル数に対してほぼ一定の割合となる。これにより、BoVW+HE は図らずも^(注7)上記の比率を利用したフィルタリングを行っており、そのことも高精度な認識に貢献していると考えられる。

6. む す び

本稿では、直積量子化を利用した近似最近傍探索手法に対し、分布の異なるベクトル集合に対し同一の直積量子化コードブックを用いることにより符号化効率が低下する問題を指摘した。更に、コードブックを分布に応じ

(注7)：本来の目的は特徴ベクトル間の距離を近似することであった。

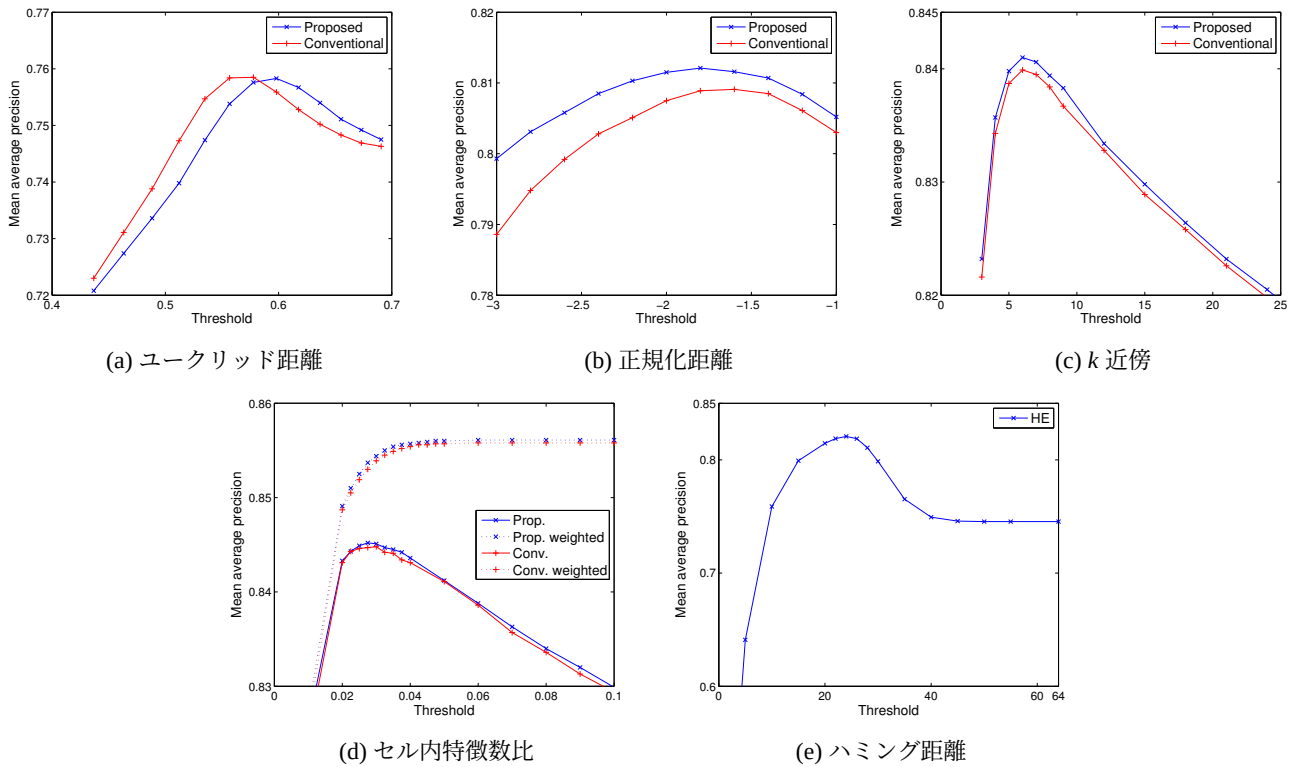


図7 異なる指標を閾値としてフィルタリングを行った際の特定物体認識精度

て切り替えて利用することを提案し、任意の数のコードブックを繰り返し処理によって最適化しながら作成する手法を提案した。評価実験により、提案手法は既存手法と同等の速度およびメモリ使用量で大幅な精度改善を実現し、精度とメモリ使用量の優れたトレードオフを実現できることを確認した。また、直積量子化を利用した近似最近傍探索を特定物体認識に応用した際の考察によって、特徴ベクトル間の推定距離によってマッチングを高精度化する際には、VWに属する特徴ベクトル数に対する比率を利用したフィルタリングが有効であることを実験により確認した。

文 献

- [1] J.H. Friedman, J.L. Bentley, and R.A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. on Mathematical Software*, vol.3, no.3, pp.209–226, 1977.
- [2] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A.Y. Wu, "An optimal algorithm for approximate nearest neighbor searching in fixed dimensions," *JACM*, vol.45, no.6, pp.891–923, 1998.
- [3] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol.60, no.2, pp.91–110, 2004.
- [4] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *IJCV*, vol.42, no.3, pp.145–175, 2001.
- [5] A. Andoni, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Proc. of FOCS*, pp.459–468, 2006.
- [6] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," *Proc. of CVPR*, pp.1–8, 2007.
- [7] C. Silpa-Anan and R. Hartley, "Optimised kd-trees for fast image descriptor matching," *Proc. of CVPR*, pp.1–8, 2008.
- [8] D. Nistér and H. Stewénus, "Scalable recognition with a vocabulary tree," *Proc. of CVPR*, pp.2161–2168, 2006.
- [9] J. Beis and D.G. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," *Proc. of CVPR*, pp.1000–1006, 1997.
- [10] M. Muja and D.G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," *Proc. of VISAPP*, pp.331–340, 2009.
- [11] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. on PAMI*, vol.33, no.1, pp.117–128, 2011.
- [12] M.S. Charikar, "Similarity estimation techniques from rounding algorithms," *Proc. of STOC*, pp.380–388, 2002.
- [13] H. Jégou, M. Douze, and C. Schmid, "Improving bag-of-features for large scale image search," *IJCV*, vol.87, no.3, pp.316–336, 2010.
- [14] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Proc. of NIPS*, pp.1753–1760, 2008.
- [15] J. Brandt, "Transform coding for fast approximate nearest neighbor search in high dimensions," *Proc. of CVPR*, pp.1815–1822, 2010.
- [16] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," *Proc. of ICCV*, pp.1470–1478, 2003.
- [17] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," *Proc. of SODA*, pp.1027–1035, 2007.
- [18] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L.V. Gool, "A comparison of affine region detectors," *IJCV*, vol.60, no.1-2, pp.43–72, Nov. 2005.
- [19] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," *Proc. of CVPR*, pp.524–531, 2005.