4L-05

# GPU による直積量子化を用いた近似 k 近傍探索の高速化

対比地 恭平† 天笠 俊之‡
筑波大学 情報学群 情報科学類† 筑波大学 計算科学研究センター‡

## 1 序論

直積量子化[1]は，ベクトル量子化を応用したデータ圧縮技術である．直積量子化はメモリや計算のコストを抑えられることから，近似 k 近傍探索に用いられ，顔認証やテキスト検索，深層学習など多くの分野で活用することができる．また，シンプルな仕組みであることから， GPU による高速化手法も提案されている[2][3]．

現在最も性能の良い Johnson らの手法[3]は，GPU による直積量子化を用いた高速な探索を実現している．しかし大規模なデータに対しては依然として莫大なメモリ消費量が必要となる．

そこで本稿では，GPU を用いた大規模なデータに対する探索の高速化手法を提案する．具体的には，量子化したデータベースに対して GPU で扱いやすいように前処理を施すことで，探索の高速化を目指す．

## 2 関連研究

本研究に関連する 2 つの先行研究について述べる．Wieschollek らの研究[2]では，データを 2 段階の量子化によって圧縮し，木構造で表すことを提案した．木構造によって枝刈りが可能となり，高速に探索をすることができる．しかし，メモリの消費量が大きいため，大規模なデータには適さない．

Johnson らの研究[3]では，GPU のメモリ階層に着目し，高速な k 選択アルゴリズムを提案した．これと直積量子化を組み合わせることにより，従来手法よりも高速な探索が可能となり，比較的大規模なデータも扱うことができる．しかし 10 億を超えるような，より大規模なデータに対しては複数の GPU を用いて処理しているため，さらにメモリコストを抑える必要がある．

**GPU-accelerated approximate k-nearest neighbor search using product quantization**
Kyohei Tsuihiji†
Toshiyuki Amagasa‡
†College of Information Science, University of Tsukuba
‡Center for Computational Sciences, University of Tsukuba

## 3 前提知識

### 3.1 ベクトル量子化

ベクトル量子化は，主に近似 k 近傍探索に使われる，代表的なデータ圧縮技術である．データ $\boldsymbol{x} \in \mathbb{R}^D$ が与えられたとき，ベクトル量子化によって $\boldsymbol{x}$ は以下のようにコードブック $C = \{\boldsymbol{c_i}\,;\, i = 0, 1, \ldots, k-1\}$ 中の最近傍ベクトル(コードワード)で表される．

$$\boldsymbol{x} \rightarrow q(\boldsymbol{x}) = \underset{\boldsymbol{c} \in C}{\operatorname{argmin}}\, d(\boldsymbol{x}, \boldsymbol{c}) \qquad (1)$$

ここで $d(\boldsymbol{x}, \boldsymbol{c})$ は $\boldsymbol{x}$ と $\boldsymbol{c}$ の距離を表している．

ベクトル量子化の利点は主に 2 つある．1 つは，高次元のベクトルを $\log_2 k$ ビットに圧縮することができ，メモリコストを抑えられる点だ．もう 1 つの利点は，探索時に対称距離計算(SDC)や非対称距離計算(ADC)によって高速に 2 点間の距離計算をすることができる点だ．SDC はクエリとデータベースの両方が量子化を行なっている場合，ADC はデータベースだけ量子化を行なっている場合に使われる．

ベクトル量子化を用いた探索は，メモリや計算のコストを抑えることができるが，高い精度を得るにはより多くの情報量（ビット数）が必要となる．するとコードブックのサイズが大きくなり，メモリや計算のコストが大きくなるため，画像のような高次元データには適さない．

### 3.2 直積量子化

直積量子化は，高次元ベクトルをいくつかの低次元ベクトルの直積と考え，各部分ベクトルにおいてベクトル量子化を適用する(図 1)．つまり，データは $\boldsymbol{x} = [\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^m]$，コードブックは

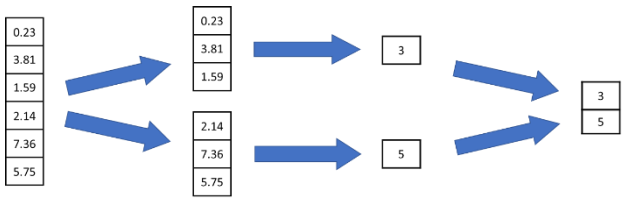


図 1: 直積量子化の流れ



Copyright ©2021 Information Processing Society of Japan.
All Rights Reserved.

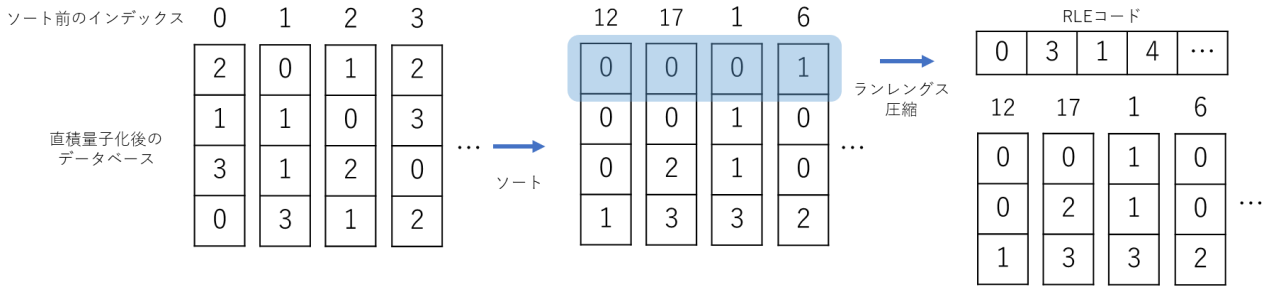


図 1: データベースに対する前処理

$C = [C^1 \times C^2 \times \ldots \times C^m]$と表される．各部分空間において式(1)を適用すると，$\boldsymbol{x}$は最終的に以下のように量子化される．

$$\boldsymbol{x} \rightarrow Q(\boldsymbol{x}) = [q_1(\boldsymbol{x}), q_2(\boldsymbol{x}), \ldots, q_m(\boldsymbol{x})] \qquad (2)$$

直積量子化は高次元のベクトルを$m \log_2 k$ビットに圧縮できる．ベクトル量子化よりはメモリを消費してしまうが，それでも十分にメモリコストを抑えることができる．また，SDC や ADC によって高速に距離計算ができる上に，ベクトル量子化よりも高い精度で探索を行うことができる．しかし，それでも大規模なデータを扱う場合はメモリ消費量や計算時間が増えるため，さらなる高速化が必要である．

## 4 提案手法

本節では，GPU によって大規模データにおける高速な探索を行うための手法を提案する．先行研究ではクエリごとに各部分空間の距離計算を行っていたが，本研究ではクエリをまとめて扱い，部分空間ごとに距離計算を行う．また，図 1 に示すように，事前にデータに対してソートやランレングス圧縮(RLE)を行い，GPU を用いて並列に処理を行う．

### 4.1 前処理

直積量子化によって圧縮したデータをソートする．これにより上位の部分空間ほど同じ量子化値が連続するため，部分空間ごとに計算を行う場合，比較的小さいメモリ空間を扱うことになる．

ソートしたデータを部分空間ごとにランレングス圧縮によってさらに圧縮する．これにより，データが占めるメモリ消費量を減らすだけでなく，GPU による並列処理が容易になる．

### 4.2 GPU による並列処理

ランレングス圧縮適用の部分空間とそうでない部分空間に分けて考えるが，両者ともルックアップテーブル(クエリとコードワード間の距離をまとめた表)を参照し，クエリとデータベースの部分空間ごとの距離を近似的に求める．すべての部分空間の距離が求まったらそれらを足し合わせ，その値が最小となる k 個のデータを出力する．

## 5 まとめ

本稿では GPU を用いた直積量子化による大規模データの探索の高速化手法を提案した．事前にデータをソートしておくことでメモリアクセス効率を上げ，ランレングス圧縮を行うことでメモリ消費量を減らした．今後は評価実験を通して本手法の有用性を評価する予定である．

## 参考文献

[1] H. Jégou, M. Douze, C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 33(1):117-128, January 2011.

[2] P. Wieschollek, O. Wang, A. Sorkine-Hornung, and H. P. A. Lensch. Ecient large-scale approximate nearest neighbor search on the GPU. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 2027-2035, June 2016.

[3] J. Johnson, M. Douze and H. Jégou, Billion-scale similarity search with GPUs, *IEEE Trans. Big Data*, Jun. 2019.



Copyright ©2021 Information Processing Society of Japan.
All Rights Reserved.