GPU を用いた高次元ベクトルの最近傍探索の高速化と自動チューニング方式の検討

Auto-tuning for nearest neighbor search of high-dimensional vectors using GPUs

村上 明男† 若谷 彰良‡
Akio Murakami Akiyoshi Wakatani

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目的とした高次元ベクトルの大規模な最近傍探索の手法として，直積量子化を用いた近似最近傍探索がある[1]．本稿では，直積量子化に基づく近似最近傍探索においてリファレンスデータの再配置を行い，GPGPU(General Purpose computing on GPU) により探索時間を短縮する手法を提案し，その評価を行う．また，そのための自動チューニング方式を検討する．

2. 直積量子化に基づく近似最近傍探索

近似最近傍探索の手順を以下に示す．

1. リファレンスデータから粗量子化コードブックを生成し，インデックス構造を作成する
2. 粗量子化コードブックを用いて探索ベクトルを量子化し，インデックスとの残差を求める
3. 2.で求めた残差とリファレンスデータとの距離計算により近似最近傍を探索する

一方，2.で求めた残差と直積量子化のコードブックとの距離を計算する．計算結果をルックアップテーブル（以下，LUT）というデータ構造に格納し，3.で行う距離計算をLUT の参照に置換する方法もある．本研究では，LUT を使用する場合と使用しない場合での差異も検証する．

3. スレッド別最近傍候補数の検討

本研究では，スレッド別に最近傍候補(=m)を抽出してから近似最近傍 K 個を決定する．スレッドブロック内のスレッド数を P とすると，保守的な手法である m=K では，P×K 個の最近傍候補を抽出してから K 個の近似最近傍を決定するが，このリダクション計算において多大な時間を要する．そこで，m の値を，精度を保つのに必要な最小の数になるように調整することで，リダクション計算に要する時間を削減する．

4. 実験結果と考察

近似最近傍探索の GPGPU 化プログラムを実装し，実行時間の計測と評価を行った．リファレンスデータ(=N)を 100,000 個，探索ベクトルを 100 個，部分ベクトルの次元数（=D）を 16，近似最近傍の数を 256 個，粗量子化および直積量子化のコードブックサイズを 256 とし，(P, m)=(1,256), (2,150), (4,83), (8,51), (16,30), (32,18), (64,13), (128,9), (256,7)で実行した．なお，1 個の探索ベクトルを 1 個のスレッドブロックに割り当てた．GPU は NVIDIA GeForce GTX590（1024 コア），CPU は Intel Core i7 875K, メモリは 4 GB を用いる．OS は Yellow Dog Enterprise Linux for CUDA を，CUDA のバージョンは 4.0 を用いる．実行結果を図 1 に示す．

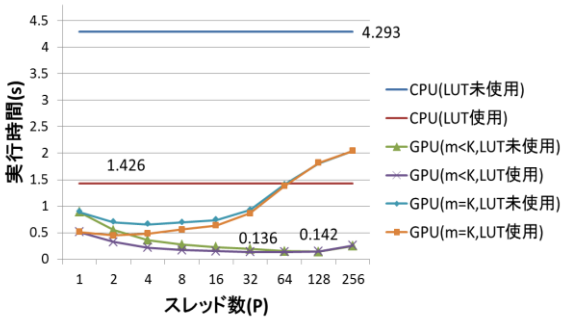

図 1: 実行時間の比較

GPU および LUT を使用し(P, m)=(32, 18)として実行した場合で最も実行時間が短い(0.136[s])という結果を得た．また，LUT を使用する場合，使用しない場合よりも実行時間が最短となるスレッド数が小さい．他のケースでの実行結果より，最適となる m の値，P の値，LUT の使用の有無は変わる．よって，これらを自動的に調整する（自動チューニング）必要がある．

5. 予測式と自動チューニング

探索対象ベクトル数を W，直積量子化のコードブックサイズを C_pとするとき，LUT を使用しない場合の実行時間 T_1，使用する場合の実行時間T_2は，以下のように表せる．

$$T_1 = \alpha_1 \cdot N \cdot D \cdot m / P + \beta \cdot m \cdot P \cdot K$$

$$T_2 = \alpha_2 \cdot N \cdot D \cdot m / P + \beta \cdot m \cdot P \cdot K + \alpha_1 \cdot W \cdot D \cdot C_P$$

なお，ここで，α_1，α_2，βは比例パラメータである．自動チューニングのためにこれらを求める必要があるので，数回の予備実行と最小二乗法によって求める．予備実行の回数および内容は，今後の実験により決定していく．

6. 終わりに

直積量子化を用いた近似最近傍探索の GPGPU 化において，スレッド別最近傍候補数およびスレッド数を最適化することで高速化を実現した．今後，実行時間の予測式および予備実行の方式を決定し，最適なスレッド数と LUT の使用の有無を決定する自動チューニングの実現と他のシステムでの検証を行っていく．

参考文献

[1] Herve Jegou and Matthijs Douze, “Product quantization for nearest neighbor search,” Cordelia Schmid IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, issue 1, pp. 117-128, 2011.

†甲南大学大学院自然科学研究科
‡甲南大学知能情報学部