Accelerating Large-Scale Inference with Anisotropic Vector Quantization

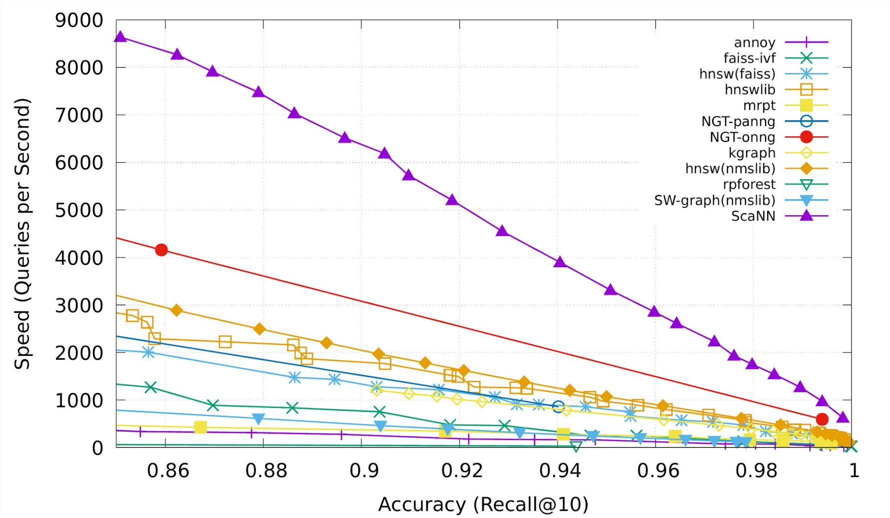
Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, Sanjiv Kumar

Google Research

Overview

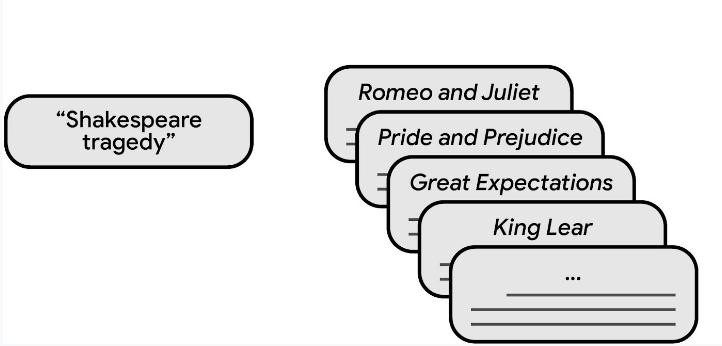
Vector quantization optimized for MIPS with a new loss

Open-source implementation (ScaNN) with leading performance on ann-benchmarks.com



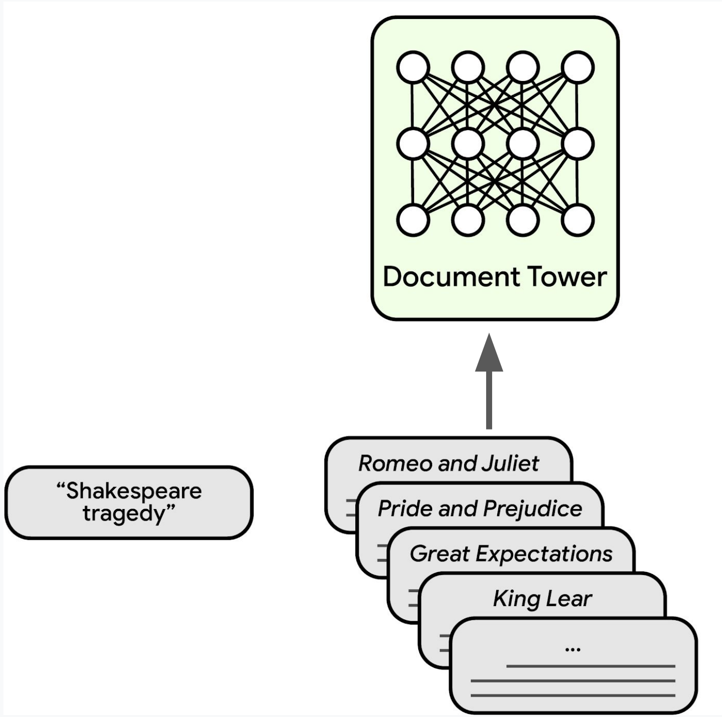
Google Research

Application: recommender systems



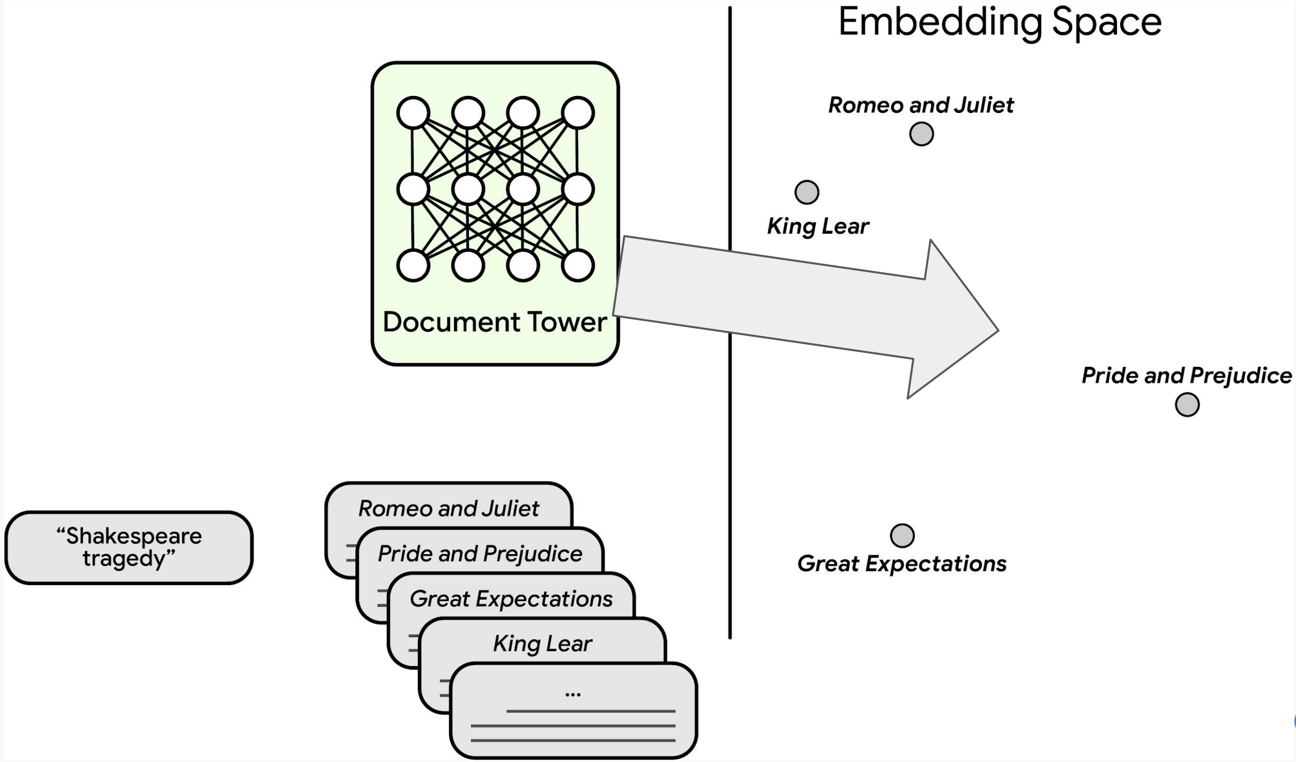
Google Research

Application: recommender systems



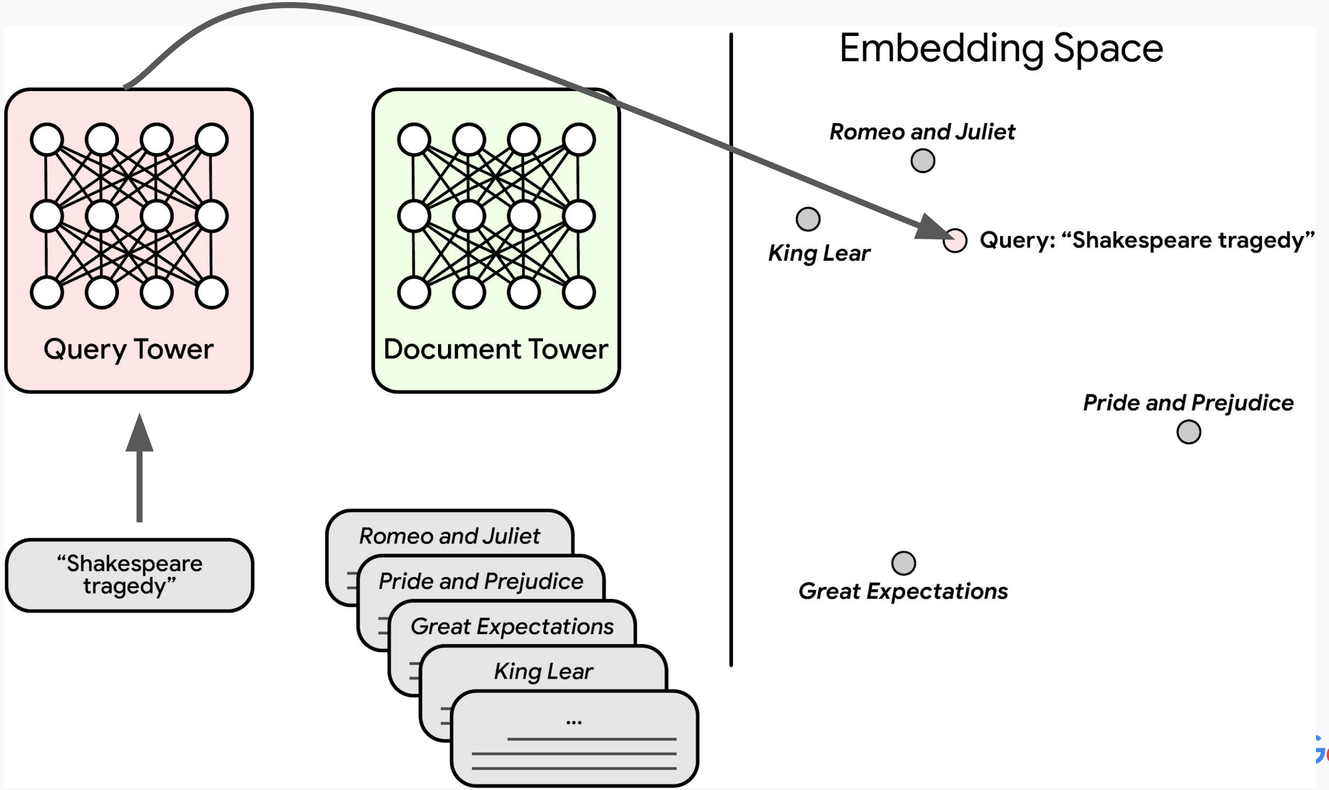
Google Research

Application: recommender systems



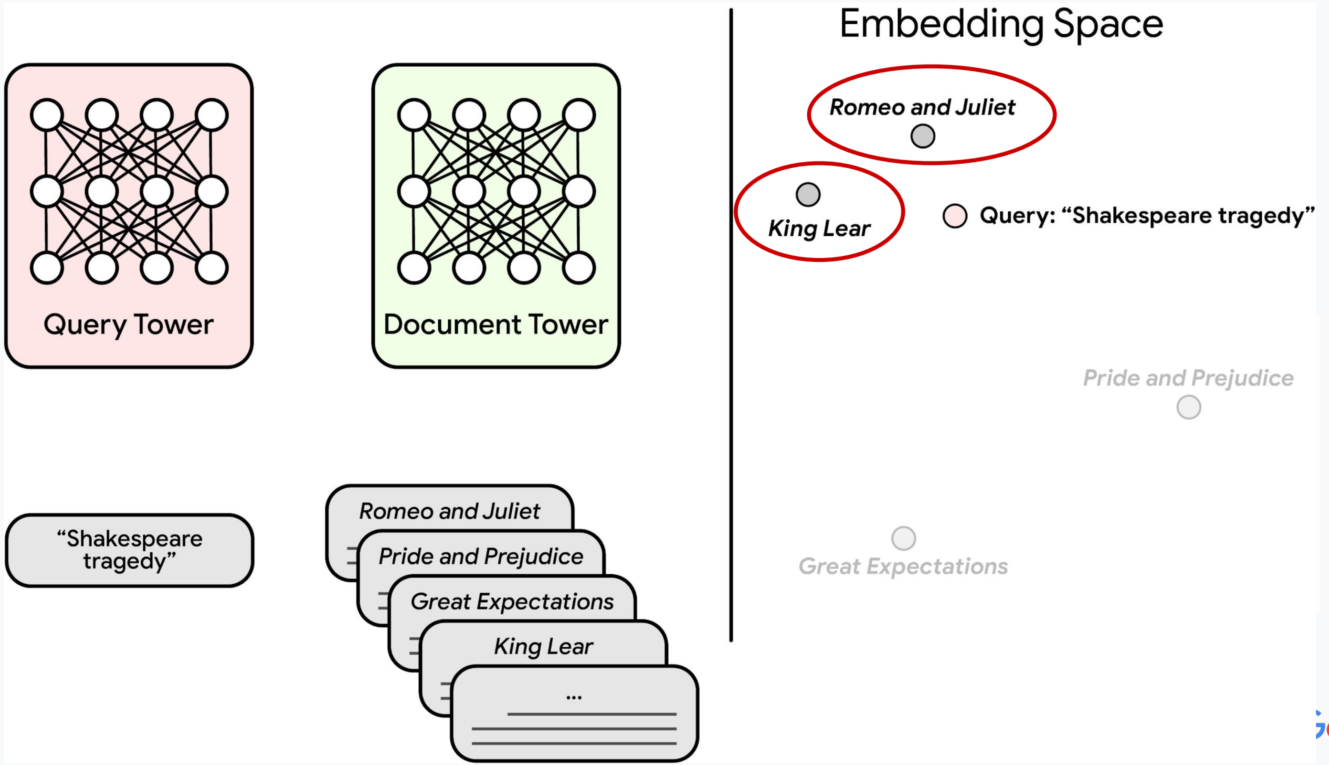
Google Research

Application: recommender systems



Google Research

Application: recommender systems



MIPS: partitioning and quantization

Partitioning:

- Split database into disjoint subsets
- Search only the most promising subsets

MIPS: partitioning and quantization

Partitioning:

- Split database into disjoint subsets
- Search only the most promising subsets

Quantization:

- Reduce the number of bits used to describe data points.
- Leads to smaller index size and faster inner product calculations.

MIPS: partitioning and quantization

Partitioning:

- Split database into disjoint subsets
- Search only the most promising subsets

Quantization:

- Reduce the number of bits used to describe data points.
- Leads to smaller index size and faster inner product calculations.

Quantization overview: codebooks

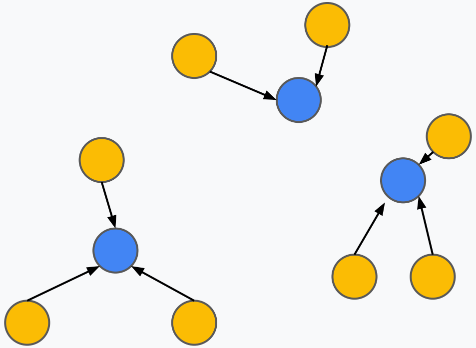
Given a set of vectors $x_1, x_2, \ldots, x_n$, we want to create a quantized dataset $\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n$.

Quantize to an element of the codebook, C_θ

Example codebook: vector quantization

Parameters are a set of centers $c_1, c_2, \ldots, c_k$.

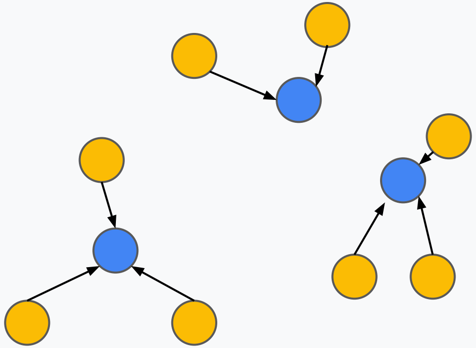
Codebook C_θ is the set of all centers: $\{c_1, c_2, \ldots, c_k\}$.



Example codebook: vector quantization

Parameters are a set of centers $c_1, c_2, \ldots, c_k$.

Codebook C_θ is the set of all centers: $\{c_1, c_2, \ldots, c_k\}$.



Product quantization:

- splits the space into multiple subspaces
- uses a vector quantization codebook for each subspace.

Google Research

Quantization basics: assignment

To assign a datapoint to a codeword, we select the codeword that minimizes a loss function.

$$\tilde{x}_i = \arg\min_{\tilde{x} \in C_\theta} \mathcal{L}(x_i, \tilde{x})$$

Google Research

Traditional loss function choice

Classic approach: reconstruction error.

$$\mathcal{L}(x, \tilde{x}) = ||x - \tilde{x}||^2$$

Traditional loss function choice

Classic approach: reconstruction error.

$$\mathcal{L}(x, \tilde{x}) = ||x - \tilde{x}||^2$$

By Cauchy-Schwartz:

$$(\langle q, x \rangle - \langle q, \tilde{x} \rangle)^2 \leq ||q||^2 ||x - \tilde{x}||^2$$

Some inner product errors are worse than others

Consider a query q and database points $x_1, \dots, x_n$

Rank points by inner product

Some inner product errors are worse than others

Consider a query q and database points $x_1, \ldots, x_n$
Rank points by inner product

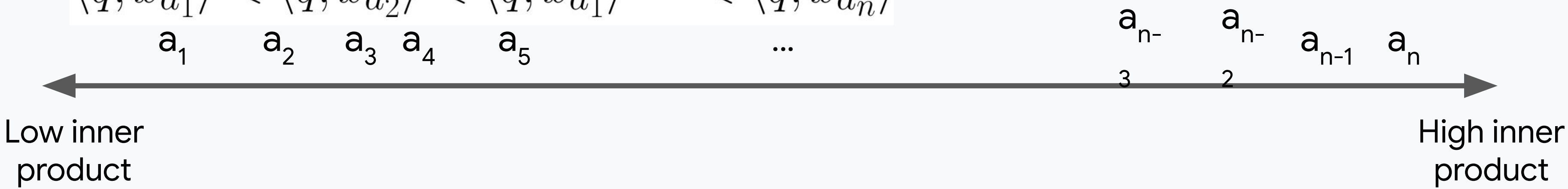
$$\langle q, x_{a_1} \rangle < \langle q, x_{a_2} \rangle < \langle q, x_{a_1} \rangle \cdots < \langle q, x_{a_n} \rangle$$

Some inner product errors are worse than others

Consider a query q and database points $x_1, \ldots, x_n$

Rank points by inner product

$$\langle q, x_{a_1} \rangle < \langle q, x_{a_2} \rangle < \langle q, x_{a_1} \rangle \cdots < \langle q, x_{a_n} \rangle$$



Some inner product errors are worse than others

Consider a query q and database points $x_1, \dots, x_n$

Rank points by inner product

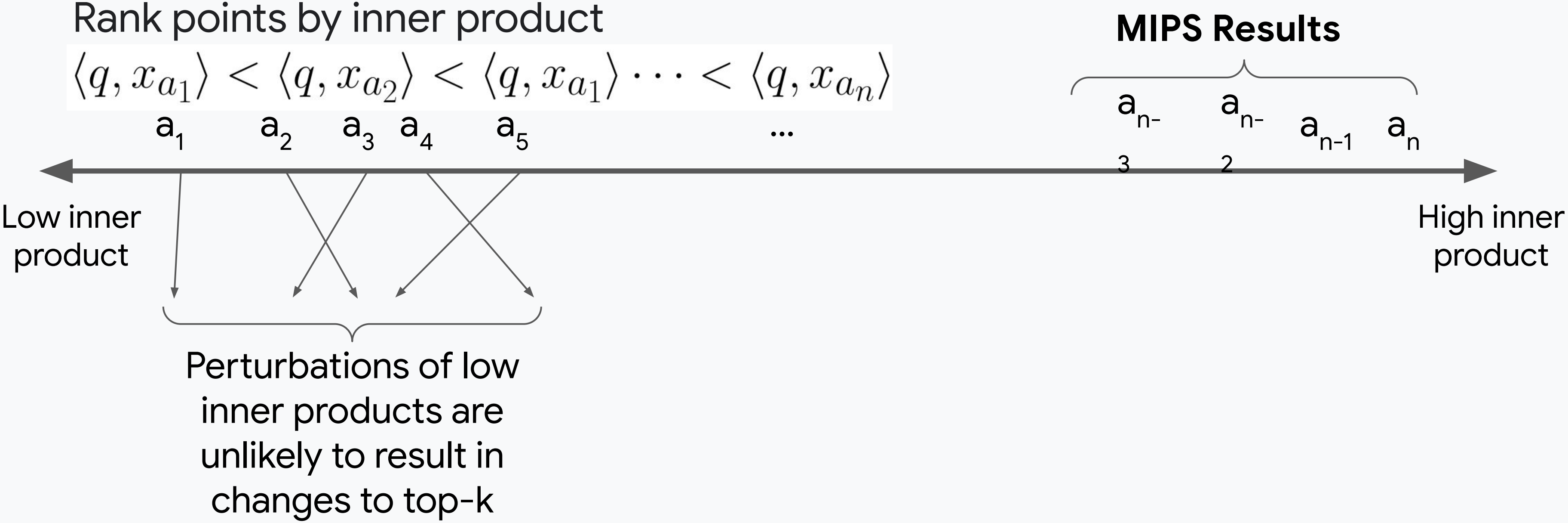
$$\langle q, x_{a_1} \rangle < \langle q, x_{a_2} \rangle < \langle q, x_{a_1} \rangle \cdots < \langle q, x_{a_n} \rangle$$

Some inner product errors are worse than others

Consider a query q and database points $x_1, \ldots, x_n$

Rank points by inner product

$$\langle q, x_{a_1} \rangle < \langle q, x_{a_2} \rangle < \langle q, x_{a_1} \rangle \cdots < \langle q, x_{a_n} \rangle$$

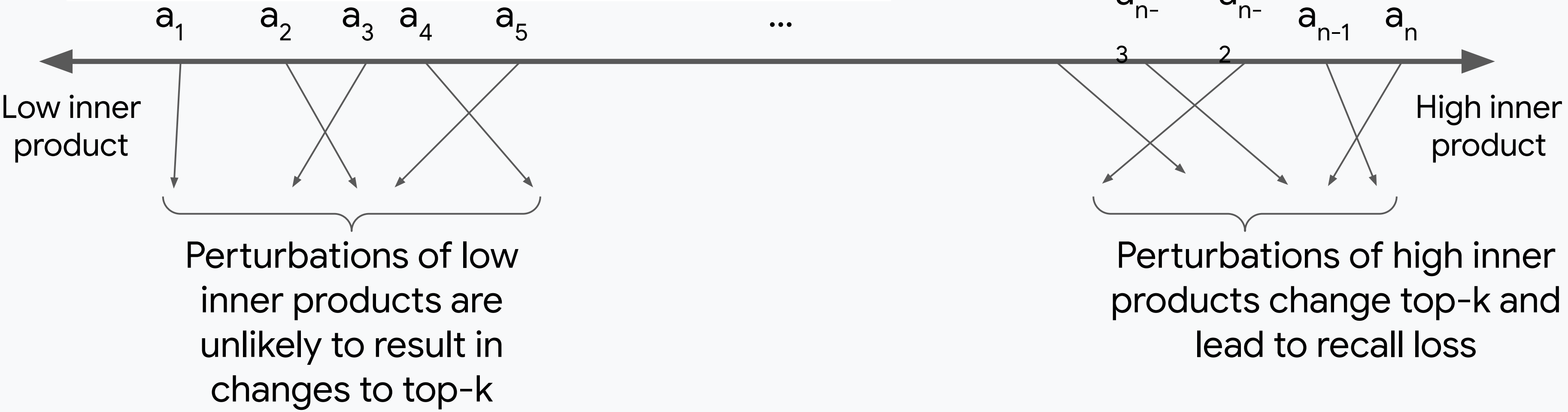


Some inner product errors are worse than others

Consider a query q and database points $x_1, \dots, x_n$

Rank points by inner product

$$\langle q, x_{a_1} \rangle < \langle q, x_{a_2} \rangle < \langle q, x_{a_1} \rangle \cdots < \langle q, x_{a_n} \rangle$$

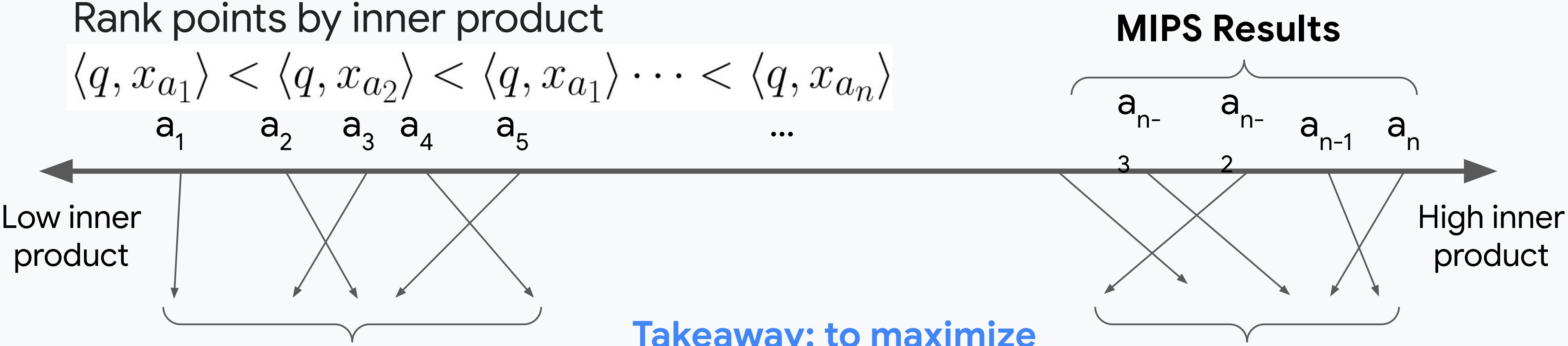


Some inner product errors are worse than others

Consider a query q and database points $x_1, \ldots, x_n$

Rank points by inner product

$$\langle q, x_{a_1} \rangle < \langle q, x_{a_2} \rangle < \langle q, x_{a_1} \rangle \cdots < \langle q, x_{a_n} \rangle$$



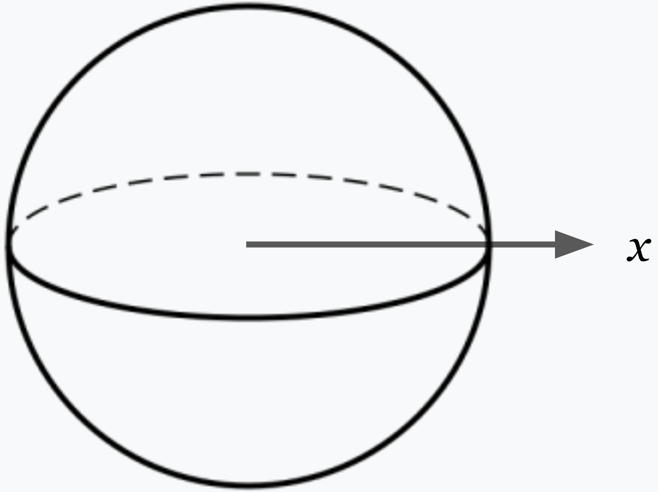
Perturbations of low inner products are unlikely to result in changes to top-k

Takeaway: to maximize recall, emphasize reducing quantization error for high inner products

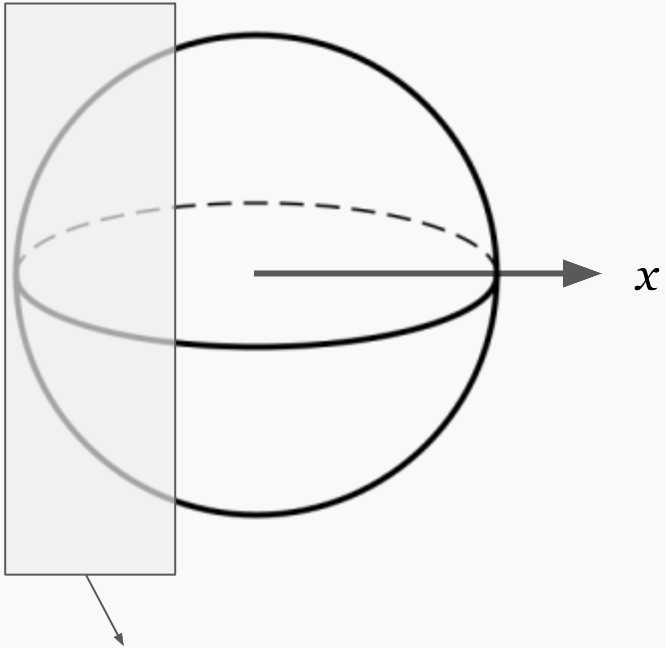
Perturbations of high inner products change top-k and lead to recall loss

Google Research

Visualization of query distribution

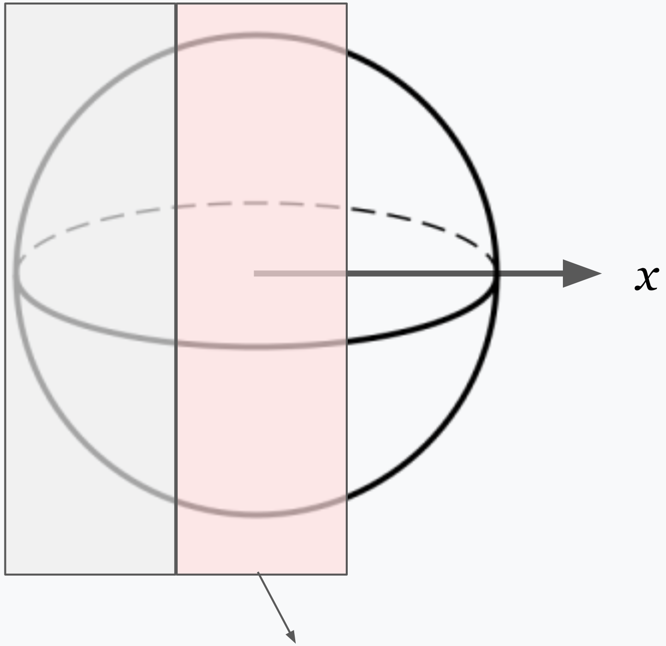


Visualization of query distribution



Quantization error: little impact on MIPS recall

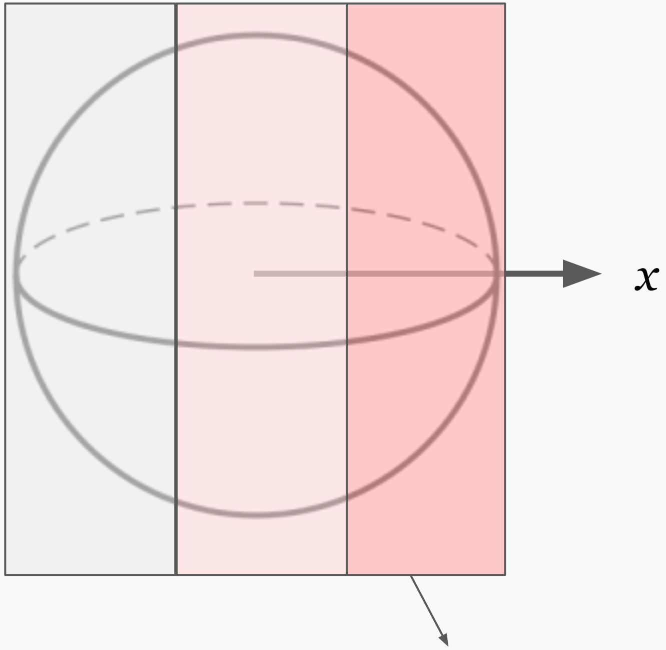
Visualization of query distribution



Quantization error: some impact on MIPS recall

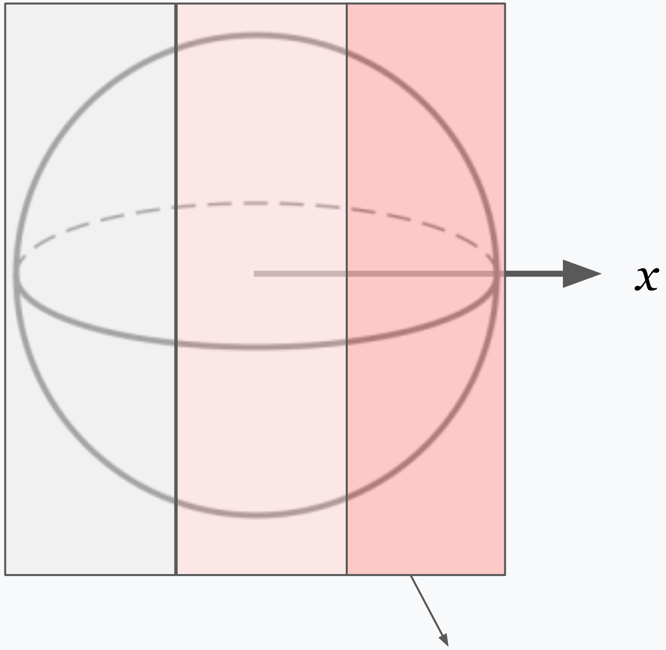
Google Research

Visualization of query distribution

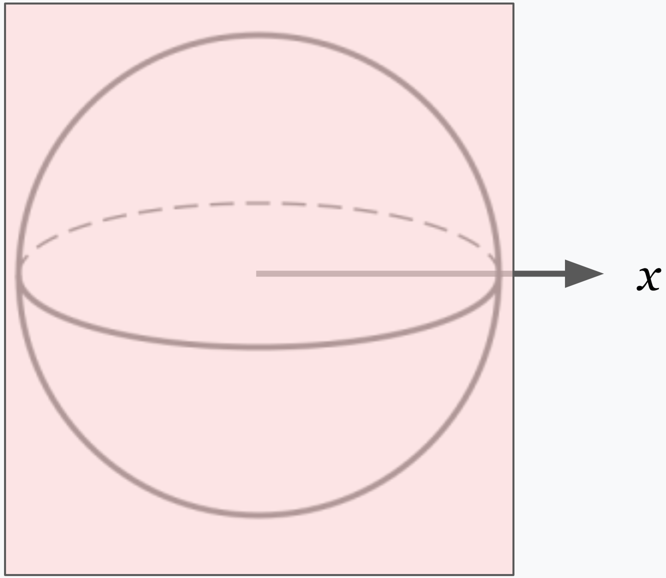


Quantization error: significant impact on MIPS recall

Visualization of query distribution



Quantization error: significant impact on MIPS recall



Reconstruction loss

Score-aware quantization loss

Traditional quantization loss: $\mathbb{E}_{q\sim\mathcal{Q}}[\langle q, x_i - \tilde{x}_i\rangle^2]$

Score-aware loss: $\mathbb{E}_{q\sim\mathcal{Q}}[w(\langle q, x_i\rangle)\langle q, x_i - \tilde{x}_i\rangle^2]$

$w : \mathbb{R} \mapsto \mathbb{R}^+$

By earlier intuition, w should put more weight on higher $\langle q, x_i\rangle$.

Example weight function: $w(t) = \mathbf{1}(t \geq T)$.

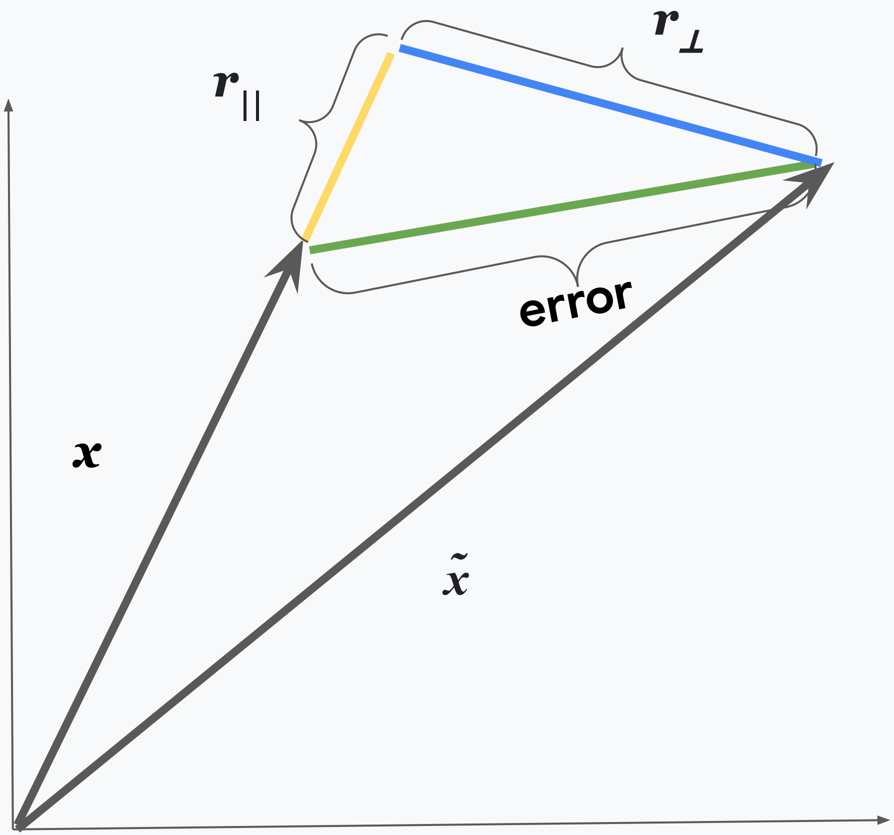
Evaluating and minimizing score-aware loss

$$\mathbb{E}_{q\sim\mathcal{Q}}[w(\langle q, x_i\rangle)\langle q, x_i - \tilde{x}_i\rangle^2]$$

Expand expectation:

$$\int_{-||x_i||}^{||x_i||} w(t)\underbrace{\mathbb{E}_q[\langle q, x_i - \tilde{x}_i\rangle^2 | \langle q, x_i\rangle = t]}_{\frac{t^2}{\|x\|^2}||r_{\parallel}(x,\tilde{x})||^2 + \frac{1-\frac{t^2}{\|x\|^2}}{d-1}||r_{\perp}(x,\tilde{x})||^2} d\mathrm{P}(\langle q, x_i\rangle \leq t)$$

Evaluating and minimizing score-aware loss



Google Research

Evaluating and minimizing score-aware loss

$$\int_{-||x_i||}^{||x_i||} w(t)\mathbb{E}_q[\langle q, x_i - \tilde{x}_i\rangle^2 | \langle q, x_i\rangle = t] d\mathrm{P}(\langle q, x_i\rangle \leq t)$$

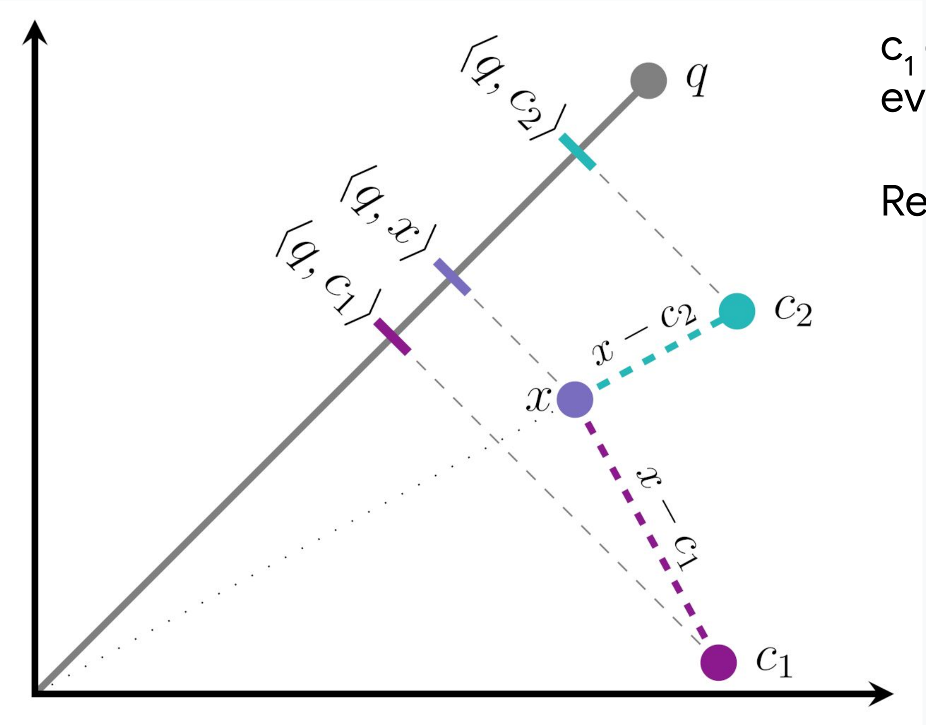
Integral evaluates to a weighted sum of $r_{||}$ and $r_{\perp}$:

$$h_{i,||}||r_{||}(x_i, \tilde{x}_i)||^2 + h_{i,\perp}||r_{\perp}(x_i, \tilde{x}_i)||^2$$

For w that weight higher inner products more, $h_{i,||} > h_{i,\perp}$

Google Research

Visualization of result



c_1 gives lower inner product error than c_2 even though $||x - c_1|| > ||x - c_2||$

Reason: $x - c_1$ is orthogonal, not parallel, to x

Applications to quantization

Given a family of codewords C, we now want to solve the following optimization problem.

$$\arg\min_{\theta} \sum_{x_i} \min_{\tilde{x}_i \in C_\theta} h_{i,\parallel} \|r_\parallel(x_i, \tilde{x}_i)\|^2 + h_{i,\perp} \|r_\perp(x_i, \tilde{x}_i)\|^2$$

We work out an approach for efficient approximate optimization in the large-scale setting for:

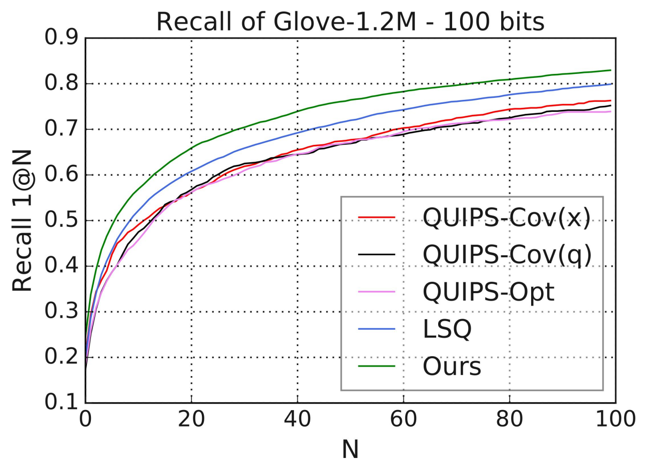
1. Vector Quantization
2. Product Quantization

Constant-bitrate comparison

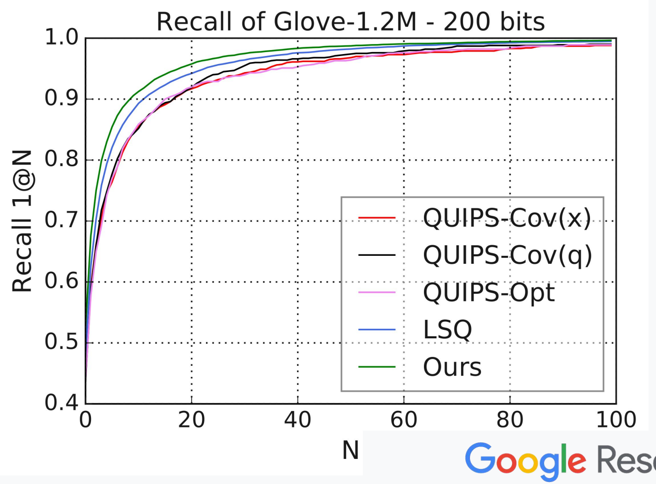
GloVe: 100 dimensions, 1183514 points

Cosine distance dataset; normalize dataset to unit-norm during training time

25 codebooks, 16 centers each



50 codebooks, 16 centers each



Google Research

Glove: QPS-recall experiment setup

Pruning via K-means tree → **Quantized Scoring** → **Exact re-scoring**

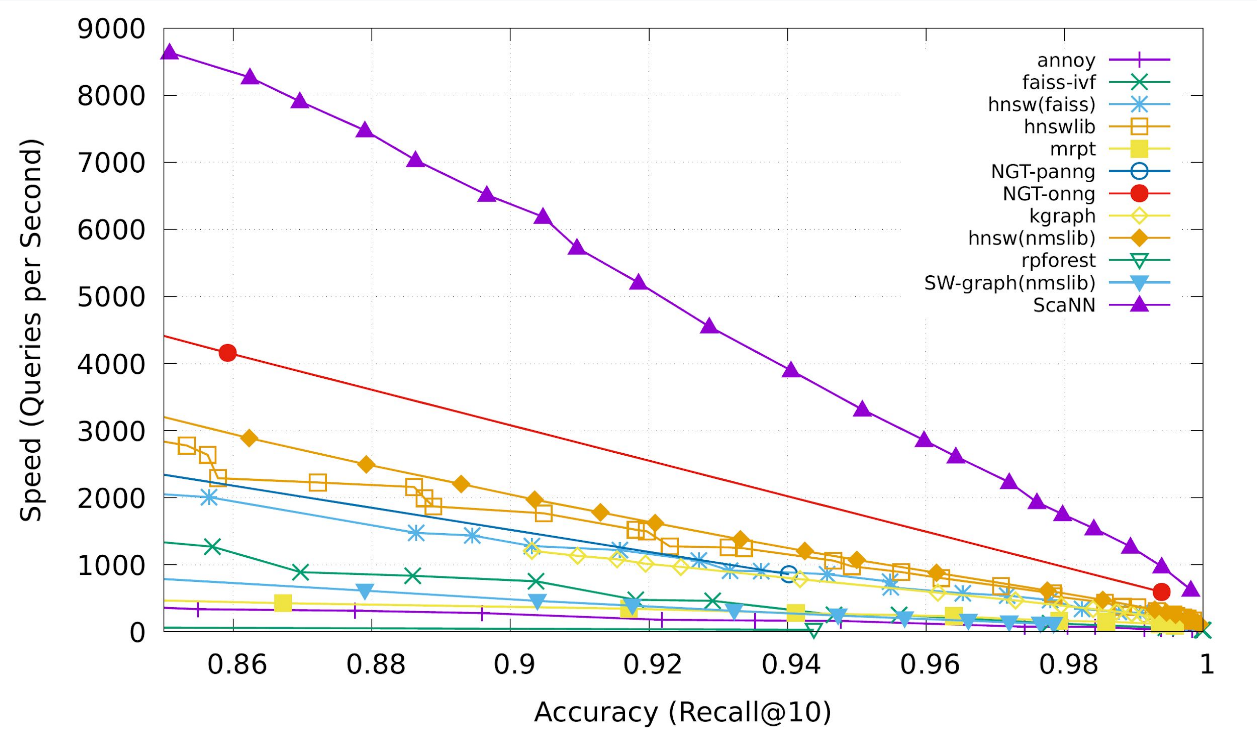
2000 centers; all but the closest a centers to the query are pruned

Compute approximate inner products via with quantized database (product quantization with anisotropic loss)

Top b inner products from AH are re-computed exactly; top 10 are returned as MIPS results

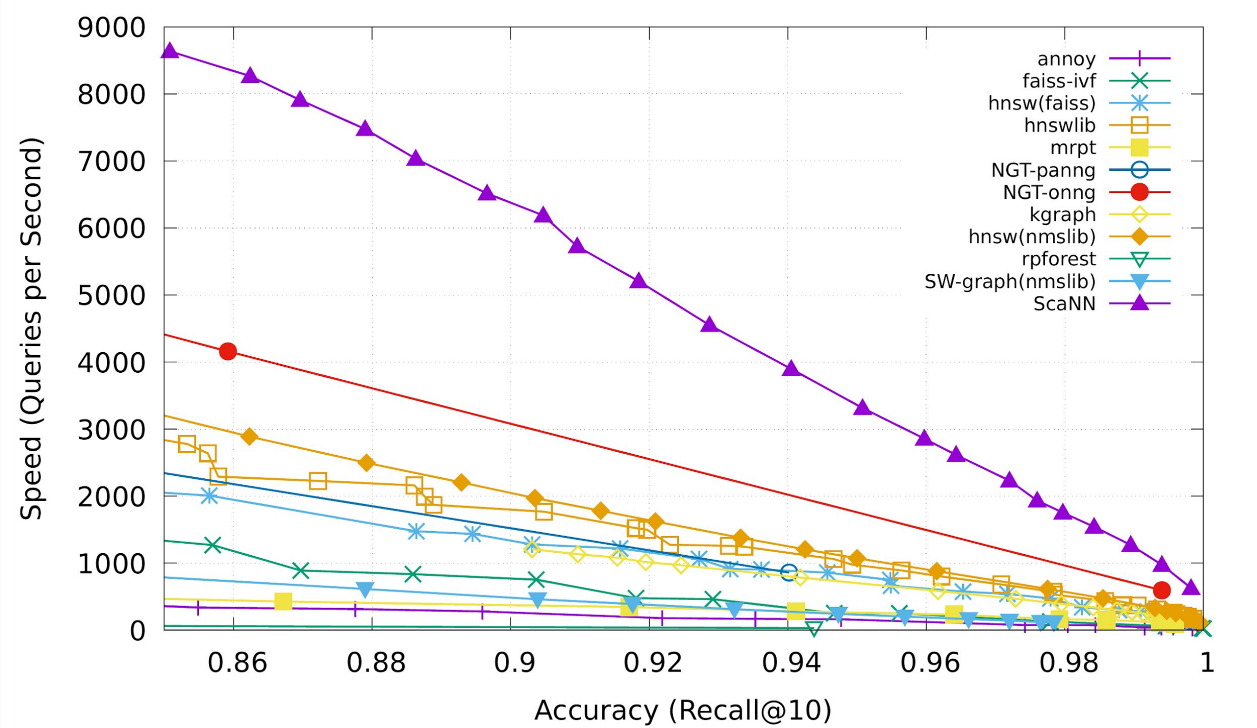
Higher a, b result in higher recall, lower QPS

Glove: QPS-recall pareto frontier



Google Research

Glove: QPS-recall pareto frontier



Google Research

Source code: https://github.com/google-research/google-research/tree/master/scann