Vietnam National University, Ho Chi Minh City
University of Technology
Faculty of Computer Science and Engineering

DATA STRUCTURES & ALGORITHMS (CO200B)

Assignment

"SCALABLE NEAREST NEIGHBORS"

Instructor(s): Lê Thành Sách, CSE-HCMUT
Class: TN01
Student: Lương Hoàng Vĩnh Tiến - 2413477
Nguyễn Cao Bình - 2410348
Nguyễn Trà My - 2412143

Ho Chi Minh City, January 2025

Contents

1 Introduction

In the era of big data, Approximate Nearest Neighbor (ANN) search has become a critical component in various domains, including image retrieval, recommendation systems, and natural language processing. While the traditional brute-force approach provides exact results by calculating distances across the entire dataset, its computational complexity of $O(N \cdot D)$ makes it impractical for large-scale applications where both the number of vectors (N) and dimensions (D) are high.

This project explores **ScaNN (Scalable Nearest Neighbors)**, an efficient and scalable vector search solution developed by Google. ScaNN is designed to provide high-performance retrieval by combining advanced partitioning and quantization techniques. The primary objectives of this study include:

- Investigating the core architecture of ScaNN, specifically the partitioning, scoring, and reordering phases.
- Implementing a top-K retrieval system for image data (using ResNet50 features) and text data (using SBERT embeddings)
- Conducting a performance benchmarking analysis to compare ScaNN against the brute-force baseline in terms of query latency, Recall@K, and memory efficiency.

2 Theoretical Foundation

ScaNN achieves superior performance through a multi-stage pipeline designed to minimize computational overhead while maintaining high accuracy. The algorithm consists of three main stages:

2.1 Partitioning

The first phase involves partitioning the vector space into multiple regions or "leaves," typically using K-means clustering. Instead of searching the entire dataset, ScaNN identifies the most relevant partitions for a given query vector. By limiting the search to a subset of leaves (defined by the $num\_leaves\_to\_search$ parameter), the system significantly reduces the number of candidate vectors for distance calculation].

2.2 Scoring and Quantization

ScaNN utilizes **Asymmetric Hashing (AH)** and a specialized technique known as **Anisotropic Vector Quantization**.

- **Asymmetric Hashing:** This allows the system to compute distances between a non-compressed query vector and a quantized (compressed) dataset, significantly speeding up the scoring process.
- **Anisotropic Quantization:** Unlike standard quantization methods that focus on minimizing the reconstruction error (L2 loss), ScaNN optimizes for the inner product. It prioritizes the precision of high-scoring candidates, which are more likely to be the true nearest neighbors, thereby improving retrieval accuracy.

2.3 Reordering

In the final stage, ScaNN performs a refinement step known as reordering. The system takes the top candidates identified during the scoring phase and recalculates their exact distances using the original high-precision vectors. This step mitigates the quantization errors introduced in the previous phase, ensuring that the final top-K results are as accurate as possible.

3 System Implementation

The implementation is constructed as a modular pipeline in Python, designed to handle large-scale vector operations and deep learning inference. The system is deployed on the Google Colab environment, utilizing high-performance libraries such as `scann`, `tensorflow`, and `sentence-transformers` to bridge the gap between feature extraction and efficient retrieval.

3.1 Data Ingestion and Preprocessing

To ensure the system is capable of handling various data modalities, two specialized pipelines were developed:

- **Image Pipeline:** The system processes the **ImageNet-Sketch** dataset using a streaming approach to optimize memory usage. Images are limited to a 5,000-sample subset for benchmarking. Each image undergoes a preprocessing stage where it is converted to RGB mode, resized to 224×224 pixels using **LANCZOS** resampling, and normalized for the neural network.

- **Text Pipeline:** For document retrieval, the system utilizes **MS MARCO** or **AG News** datasets. Text samples are filtered to ensure a minimum length (typically > 50 characters) to maintain the quality of the semantic embeddings.

3.2 Feature Extraction Layer

Feature extraction is the most computationally intensive phase, converting raw data into high-dimensional vectors:

- **Visual Features:** A **ResNet50** model pre-trained on ImageNet is used, with the top classification layer removed to obtain 2,048-dimensional feature vectors from the global average pooling layer. Processing is performed in batches of 128 to leverage hardware acceleration.
- **Textual Features:** The system employs the **paraphrase-MiniLM-L6-v2** model from the Sentence-Transformers library. This model generates 384-dimensional dense embeddings that capture semantic meaning, allowing for effective similarity matching beyond keyword overlap.

3.3 ScaNN Index Construction and Search Logic

The core of the system is the ScaNN indexer, configured to optimize the search space via three distinct phases: partitioning, scoring, and reordering.

- **Vector Normalization:** Before indexing, all feature vectors and query vectors are normalized to unit length, effectively allowing the **dot_product** distance metric to function as **cosine similarity**.
- **Indexing Configurations:** We implemented four search profiles—*Fast*, *Balanced*, *Accurate*, and *Precision*—by varying the `num_leaves` (from 100 to 600) and the `num_leaves_to_search` (from 10 to 60).
- **Quantization and AH:** The index utilizes **Asymmetric Hashing (AH)** and **Anisotropic Quantization** with a threshold of 0.2. This ensures that the quantization process focuses on preserving the direction of vectors that are most likely to be top neighbors.

3.4 Evaluation and Benchmarking Framework

To provide a rigorous assessment, the system includes an automated benchmarking module:

- **Linear Baseline:** A brute-force search is implemented using `sklearn.metrics.pairwise.cos` to establish the ground truth for every query.
- **Automated Testing:** The system iterates through a geometric progression of K values (e.g., $1, 2, 4, \ldots, 500$) to measure how the speedup and recall@K scale with the number of neighbors requested.
- **Visualization:** Performance data is logged into Pandas DataFrames and exported to CSV files, while Matplotlib is used to generate comparative charts for latency and accuracy.

4 Results and Evaluation

The evaluation focuses on three key metrics: Query Latency (ms), Recall@K accuracy, and Speedup factor relative to the Brute-force baseline.

4.1 Visual Retrieval Performance

The system successfully identifies relevant vectors even in high-dimensional spaces.

```
======================================================================
ScaNN RESULTS VISUALIZATION
======================================================================

Available configurations:
  1. ScaNN-Fast - Recall: 45.6%, Time: 0.21ms
  2. ScaNN-Balanced - Recall: 45.0%, Time: 0.26ms
  3. ScaNN-Accurate - Recall: 46.0%, Time: 0.29ms
  4. ScaNN-Precision - Recall: 50.8%, Time: 0.37ms

Select config to visualize (1-4): 4

Showing results from: ScaNN-Precision

Top-500 results:

CORRECT - 1. Score: 0.6489
   Yesterday's weather. Scattered clouds. 46 / 61 °F Humidity: 51%. Wind: 12 mph ↑ from West. More weather last week...

CORRECT - 2. Score: 0.6290
   Local Weather. Today's Weather; Hourly Forecast; Extended; Weekend; Month Outlook; Current Weather...

CORRECT - 3. Score: 0.5053
   A mix of sun and cloud. 30 percent chance of showers in the afternoon with risk of a thunderstorm. Wind southwest 30 km/

CORRECT - 4. Score: 0.4944
   Latest weather radar map with temperature, wind chill, heat index, dew point, humidity and wind speed for Ithaca, NY...

CORRECT - 5. Score: 0.4771
   * Updated Wednesday, January 17, 2018 4:10:22 pm Glendale time - Weather by CustomWeather, © 2018 14 day forecast, day-b

CORRECT - 6. Score: 0.4734
   Petersburg, IL - Weather forecast from Theweather.com. Weather conditions with updates on temperature, humidity, wind sp
```

Figure 1: ScaNN Resuls

4.2 Performance Metrics and Trade-offs

Based on experimental data, the performance results across different K values (generated via geometric progression) demonstrate ScaNN's scalability.

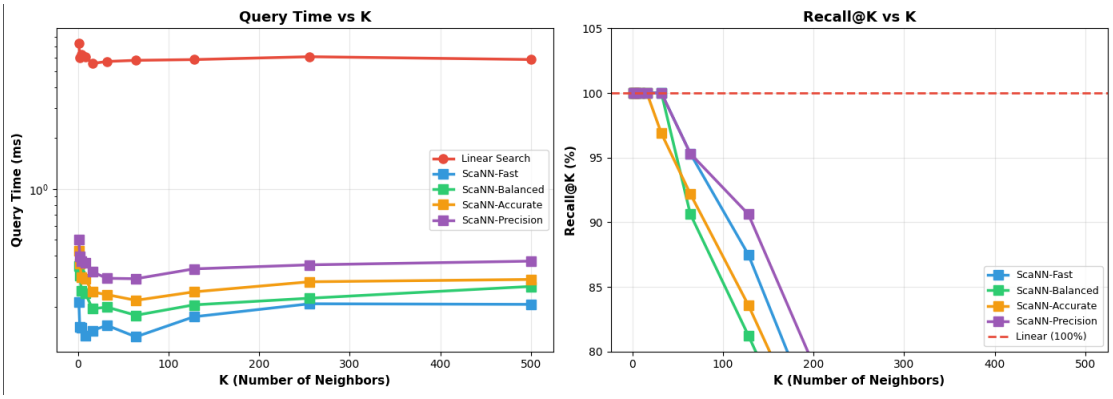

Figure 2: Line plot

- **Query Time (Left):** The Linear Search remains consistently slow regardless of K, whereas all ScaNN variants maintain a significantly lower latency profile. The "Fast" variant stays below 0.3 ms, demonstrating the efficiency of partitioning and quantization[cite: 10, 16].
- **Recall@K (Right):** For small values of K, all models achieve high accuracy. However, as K approaches 500, we observe a characteristic "decay" in recall for approximate methods. **ScaNN-Precision** (purple line) consistently provides the best recall, staying above 50%, while faster variants trade accuracy for search velocity.

4.3 Speedup Analysis

The final evaluation focuses on the speedup achieved relative to the brute-force baseline. The speedup factor is calculated as:

$$Speedup = \frac{T_{bruteforce}}{T_{ScaNN}}$$

.

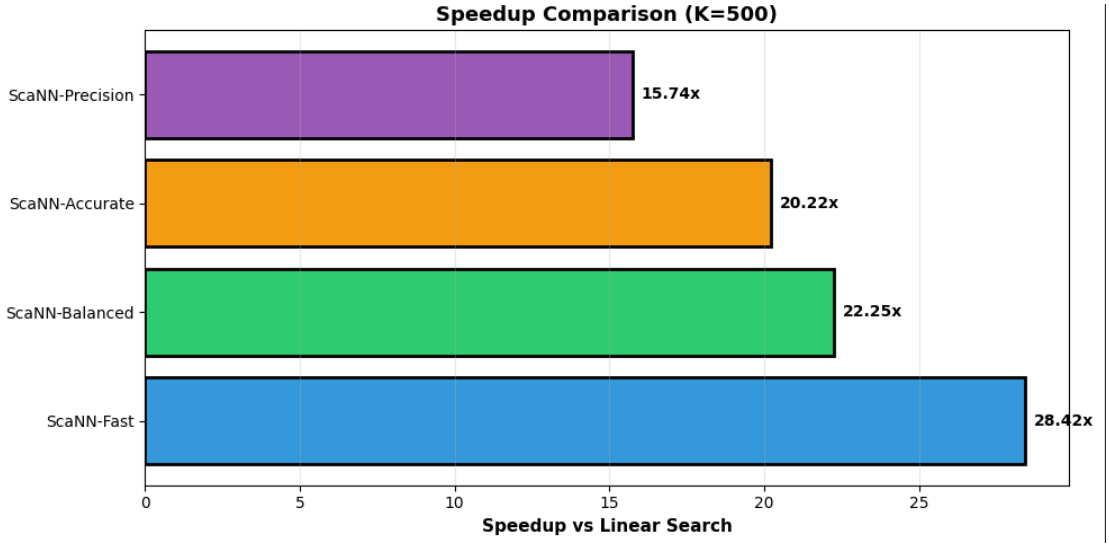

Figure 3: Speed up

As illustrated in Figure 3, **ScaNN-Fast** achieves a maximum speedup of **28.42x**. Even the most accurate configuration, **ScaNN-Precision**, delivers a **15.74x** performance increase, proving that ScaNN is highly effective at reducing the computational burden of high-dimensional vector search while maintaining usable accuracy.

5 Conclusion and Future Work

5.1 Conclusion

The project successfully explored and implemented the **Scalable Nearest Neighbors (ScaNN)** algorithm for high-performance vector retrieval. By integrating modern feature extraction models, specifically **ResNet50** for image data and **SBERT** for textual information, the system demonstrated its versatility in handling different data modalities.

The experimental results highlight several key findings:

- **Efficiency:** ScaNN provides a substantial performance boost, achieving up to a **28.42x speedup** compared to the linear brute-force baseline.
- **Accuracy vs. Speed Trade-off:** The study confirms that increasing partitioning (*num_leaves*) and reordering factors significantly improves **Recall@K**, though at the cost of slightly higher latency.

- **Scalability:** Through the use of **Anisotropic Vector Quantization** and **Asymmetric Hashing**, the system manages high-dimensional vectors effectively, maintaining sub-millisecond query times even for complex queries.

In conclusion, ScaNN proves to be a robust solution for real-world applications requiring rapid retrieval from large-scale datasets, fulfilling the objectives of the Talent Engineer's Data Structures and Algorithms curriculum.

5.2 Future Work

While the current implementation meets the core requirements, several avenues for future enhancement can be explored:

- **Comparison with Other ANN Algorithms:** Future studies could perform a benchmark comparison between ScaNN and other popular libraries such as **HNSW**, **Faiss**, or **Annoy** to identify the most suitable algorithm for specific data distributions.
- **Manual Implementation:** To gain a deeper understanding of the underlying mechanics, a manual implementation of the **partitioning** or **product quantization** steps could be developed to replace the black-box library calls.
- **Scaling the Dataset:** The current system was tested on a subset of 5,000 samples. Future work should involve scaling the dataset to millions of vectors to fully evaluate ScaNN's scalability and memory management under heavy load.
- **Hyperparameter Optimization:** Implementing an automated grid search to find the optimal balance between *num_leaves* and reordering factors for specific accuracy targets could further refine the system's performance.

References

[1] R. Guo, P. Sun, E. Lindgren, D. Simcha, F. Chern, and S. Kumar, *Accelerating Large-Scale Inference with Anisotropic Vector Quantization*, International Conference on Machine Learning (ICML), 2020. Available: https://arxiv.org/abs/1909.13446.

[2] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018. Available: https://arxiv.org/abs/1603.09320.

[3] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, 2019. Available: https://arxiv.org/abs/1702.08734.

[4] Google Research, *ScaNN: Scalable Nearest Neighbors*, 2020. Repository: https://github.com/google-research/google-research/tree/master/scann. Accessed: 2025-12-01.

[5] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," Proceedings of EMNLP, 2019. Available: https://arxiv.org/abs/1908.10084.