OAK

Cedric Sirianni, Mithi Jethwa, Lachlan Kermode

ACM Reference Format:

Cedric Sirianni, Mithi Jethwa, Lachlan Kermode. 2024. OAK. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ABSTRACT

Vector databases are growing in popularity as they become widely used in similarity search and RAG systems as part of ML workloads. At the same time, applications increasingly leverage mixed-modality data, requiring support for search over *vector data* such as images and text, and *structured data* such as metadata and keywords, simultaneously. Recent work in ACORN helps improve the feasibility of this *hybrid search* by providing a performant and predicate-agnostic index built on Hierarchical Navigable Small Worlds (HNSW), a state-of-the-art graph based index for approximate nearest neighbor search (ANNS). However, ACORN does not take into consideration predicate access patterns, leaving room for improved performance under certain modal workloads. To address this, we present OAK, a system for creating predicate subgraphs. To evaluate OAK, we compare OAK to ACORN on ….. We show that OAK achieves improved performance … Our code is available at: https://github.com/breezykermo/oak.

INTRODUCTION

In recent years, embedding models have advanced significantly, enabling the use of vector embeddings in a variety of applications. For example, retrieval-augmented generation (RAG) helps improve the accuracy and relevance of LLM outputs by including additional vector embeddings in user prompts. And, recommendation algorithms at companies like Netflix and TikTok use vector embeddings to represent user profiles and platform content. In these applications, approximate nearest neighbor search (ANNS) is the method used for performant semantic similarity search. Because machine learning inherently clusters/groups data, ANNS is a way to coalesce inherently unstructured data for specific operations, making it an essential primitive in big data operations.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

However, efficient ANNS is a challenging problem when considering multi-million or billion scale datasets. Both NeurIPS'21 [[**https://big-ann-benchmarks.com/neurips21.html?**]]and NeurIPS'23 [[**https://big-ann-benchmarks.com/neurips23.html?**]] hosted a competition for billion-scale indexing data structures and search algorithms, showcasing a wide range of solutions that improved search accuracy and efficiency. One popular approach is Hierarchical Navigable Small Worlds (HNSW) [[**https://arxiv.org/abs/1603.09320?**]], a hierarchical, tree-link structure where each node of the tree represents a set of vectors.

ANNS becomes increasingly complex when introducing predicate filtering, i.e., performing on ANNS on a subset of data that matches a given predicate. For example, customers on an e-commerce site may want to search for t-shirts similar to a reference image, while filtering on price. To support such functionality, applications must implement *hybrid search*, i.e., similarity search queries containing a one or more structured predicates. Implementations must be **performant**, retrieving results with high throughput/low latency and also **accurate**, retrieving results that are sufficiently similar to the provided query.

Several strategies exist to address these challenges with varying degrees of success. *Pre-filtering* first finds all vectors that satisfy a given predicate and then performs a similatiy search on the remaining vectors. This approach performs poorly when using medium to high selectivity predicates on large datasets. *Post-filtering* first performs a similarity search on the dataset, then filters results that do not match the given predicate. Since vectors with the greatest similarity may not satisfy the predicate, this approach sometimes requires searching repeatedly with increasingly large search spaces (top-1k, top-10k, etc.), incurring large amounts of overhead. Specialized indices such as Filtered-DiskANN [cite:] use predicates during index construction to eliminate the need for pre- or post-filtering. However, these indices restrict predicate set cardinalities to about 1,000 and only support equality predicates.

ACORN [cite:] addresses this limitation by proposing a *predicate-agnostic* index which supports unbounded and arbitrary predicates. The results are impressive but still fall short of an *oracle partition index* (Figure 1).

Given some search predicate p and dataset X, an oracle partition index is an HNSW index on X_p.

In this paper, we present **OAK** (Opportunistic ANNS K-Subgraphs), a system that combines predicate-agnostic indices with oracle partition indices to support arbitary/unbounded predicates and high performance for high-frequency query predicates.

RESEARCH PROBLEM/MOTIVATION

BACKGROUND AND RELATED WORK

MAIN DESIGN

The central premise of OAK is to route queries with high-frequency predicates to an *opportunistic index* constructed using the same predicate. When OAK receives a query q with predicate p, sending to an opportunistic index is (1) potentially more performant (if the base index is larger than the opportunistic index) but (2) potentially less accurate (if the opportunistic index does not contain all vectors that match p). We factor this performance-accuracy tradeoff into our query routing strategy.

Given some base index of vectors B and some opportunistic index of vectors X, we send q X in two cases:

(1) $\$B_p$ X
(2) $|B||X|$ >= Jaccard()

IMPLEMENTATION

OAK is built in approximately 700 lines of Rust.

Bindings

Predicate Filtering

Query Routing

EVALUATION

DEEP1B [1] and SIFT1B [2] are datasets commonly used to test performance and accuracy for VectorDBs. Similarly, the big ANN benchmarks repository [3, 4] provides various datasets calibrated to four different classes of load: filtered (including metadata), out-of distribution (queries are significantly different in distribution than the database), sparse (vectors have a majority of zero values), and streaming (load includes insertion and deletion operations).

FUTURE WORK

LOGISTICS

BIBLIOGRAPHY

[1] Artem Babenko and Victor Lempitsky. 2016. Efficient Indexing of Billion-Scale Datasets of Deep Descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2055–2063.

[2] Hervé Jégou, Romain Tavenard, Matthijs Douze, and Laurent Amsaleg. 2011. Searching in one billion vectors: Re-rank with source coding. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2011. 861–864. https://doi.org/10.1109/ICASSP.2011.5946540

[3] Harsha Vardhan Simhadri. 2024. Harsha-simhadri/big-ann-benchmarks.

[4] Harsha Vardhan Simhadri, Martin Aumüller, Amir Ingber, Matthijs Douze, George Williams, Magdalen Dobson Manohar, Dmitry Baranchuk, Edo Liberty, Frank Liu, Ben Landrum, Mazin Karjikar, Laxman Dhulipala, Meng Chen, Yue Chen, Rui Ma, Kai Zhang, Yuzheng Cai, Jiayang Shi, Yizhuo Chen, Weiguo Zheng, Zihao Wan, Jie Yin, and Ben Huang. 2024. Results of the Big ANN: NeurIPS'23 competition. https://doi.org/10.48550/arXiv.2409.17424