WunDeeDB.jl: An easy to use, zero config, WAL, SQLite backend vector database

Alexander V. Mantzaris [1]

1 Department of Statistics and Data Science, University of Central Florida (UCF), USA

DOI: 10.21105/joss.08033

Software

- Review
- Repository
- Archive

Editor: Mehmet Hakan Satman

Reviewers:

- @tlaceby
- @MichaelHopwood

Submitted: 22 March 2025
Published: 17 June 2025

Summary

WunDeeDB.jl is a package written in Julia (Bezanson et al., 2017) that provides a disk-backed system for storing, searching, and managing embedding vectors at scale, influenced by disk-oriented graph-based ANN techniques (Jayaram Subramanya et al., 2019; Pan et al., 2023; Singh et al., 2021) and the broader insights from hierarchical small-world graphs (Malkov & Yashunin, 2018; Wang et al., 2021). By maintaining embeddings in an SQLite database and optionally using graph-based indices, WunDeeDB.jl minimizes in-memory overhead while supporting efficient similarity searches on commodity hardware. Its design also facilitates integration with common vector-database or ML pipelines that rely on embedding retrieval. The widely known DiskANN algorithm, has an open source code base (Simhadri et al., 2023) and implements many of the core principles that underlines DiskANN development directions.

In contrast to fully in-memory approaches, WunDeeDB.jl leverages disk-based storage and user-configurable adjacency (e.g., HNSW, LM-DiskANN, or fallback linear search), allowing large-scale data to be handled without saturating RAM. It supports incremental insertions and deletions, ensuring the index remains up-to-date as datasets evolve. By combining these disk-native strategies with tunable BFS expansions and adjacency pruning, WunDeeDB.jl enables robust nearest neighbor searches for high-dimensional embeddings.

Features include:

- **SQLite-backed embeddings** with automatic consistency checks on dimension and data type.
- **Optional ANN indexing** (HNSW, LM-DiskANN) or linear fallback, selectable at DB initialization.
- **Incremental insert/delete** operations that update disk-based adjacency structures to keep pace with dataset changes.
- **Configurable BFS expansions** (e.g., `EF_SEARCH`, `EF_CONSTRUCTION`) to balance search speed vs. recall.
- **Choice of Precision** the user can choose any of the standard base precisions to store the embeddings as.

With these capabilities, **WunDeeDB.jl** offers a practical and scalable solution for disk-based embedding management, building on research showing the viability of disk-native approaches for large ANN indexes (Jayaram Subramanya et al., 2019; Pan et al., 2023; Singh et al., 2021). It is designed for minimal configuration, providing Write-Ahead Logging (WAL) and transactions.

Statement of Need

Approximate Nearest Neighbor (ANN) search is a key element in recommendation systems, large-scale retrieval, and embedding-based machine learning (Wang et al., 2021). Traditional in-memory approaches often face significant memory demands and slower scaling when dealing with more points than can fit in core memory. By persisting adjacency structures on disk rather than in RAM, WunDeeDB.jl tackles these bottlenecks—building on disk-based ANN research (Jayaram Subramanya et al., 2019; Pan et al., 2023; Singh et al., 2021) and provides:

1. **Reduced Memory Overhead**: Only a fraction of data resides in memory, making it feasible to handle larger datasets on commodity hardware.

2. **Dynamic Updates**: Graph-based indexing supports insertions and deletions, allowing adaptation to evolving or streaming data.

3. **High Recall**: Adjusting BFS expansions and adjacency parameters yields near state-of-the-art accuracy in neighbor retrieval.

4. **Scalable & Simple Architecture**: Built using Julia's performance ecosystem, WunDeeDB.jl integrates disk operations with numeric libraries, and is straightforward to install.

This approach benefits practitioners needing large-scale nearest neighbor indices without requiring specialized clusters or massive RAM. Within the Julia package ecosystem such an implementation is not currently provided. The closest package is introduced in Tellez & Ruiz (2022), which provides essential ANN algorithms but they reside in memory and not disk, do not provide data protection through transactions or journaling such as WAL (Write Ahead Logging). For users that need these features along with ANN, this package provides them as well as being cross platform.

Acknowledgements

Thanks to the Julia community for their continued support of open-source scientific computing.

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Jayaram Subramanya, S., Devvrit, F., Simhadri, H. V., Krishnawamy, R., & Kadekodi, R. (2019). Diskann: Fast accurate billion-point nearest neighbor search on a single node. *Advances in Neural Information Processing Systems*, *32*. https://doi.org/10.1145/3543507.3583552

Malkov, Y. A., & Yashunin, D. A. (2018). Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(4), 824–836. https://doi.org/10.1109/TPAMI.2018.2889473

Pan, Y., Sun, J., & Yu, H. (2023). Lm-diskann: Low memory footprint in disk-native dynamic graph-based ann indexing. *2023 IEEE International Conference on Big Data (BigData)*, 5987–5996. https://doi.org/10.1109/BigData59044.2023.10386517

Simhadri, H. V., Krishnaswamy, R., Srinivasa, G., Subramanya, S. J., Antonijevic, A., Pryce, D., Kaczynski, D., Williams, S., Gollapudi, S., Sivashankar, V., Karia, N., Singh, A., Jaiswal, S., Mahapatro, N., Adams, P., Tower, B., & Patel, Y. (2023). *DiskANN: Graph-structured indices for scalable, fast, fresh and filtered approximate nearest neighbor search* (Version 0.6.1). https://github.com/Microsoft/DiskANN

Singh, A., Subramanya, S. J., Krishnaswamy, R., & Simhadri, H. V. (2021). Freshdiskann: A fast and accurate graph-based ann index for streaming similarity search. *arXiv Preprint arXiv:2105.09613*. https://doi.org/10.48550/arXiv.2105.09613

Tellez, E. S., & Ruiz, G. (2022). Similaritysearch.jl: Autotuned nearest neighbor indexes for Julia. *Journal of Open Source Software*, *7*(75), 4442. https://doi.org/10.21105/joss.04442

Wang, M., Xu, X., Yue, Q., & Wang, Y. (2021). A comprehensive survey and experimental comparison of graph-based approximate nearest neighbor search. *arXiv Preprint arXiv:2101.12631*. https://doi.org/10.14778/3476249.3476255