Accelerating Billion-Scale ANNS On Modern Hardware

Jayjeet Chakraborty, Heiner Litz

With LLMs becoming ubiquitous, using LLMs to answer questions on your own data has become the most common application in the era of Gen AI. Fine-tuning / Re-training being costly, RAG has become the most common method. The “Retrieval” part of RAG consists of retrieving tokens similar to a query from a vector database using Approximate Nearest Neighbor algorithms. Since, vector search algorithms run inside the same GPU clusters as model inference / training, vector search engines should be able to leverage the GPUs embarrassingly parallel capabilities to speed up ANN searches. Vector searches should be accelerated as much as possible as they sit on the critical path in RAG applications.

When the size of the embeddings dataset fits inside the GPU memory, GPUs have been found to beat high-end CPUs for ANN searches. But, when datasets contain billions of vectors, and do not fit inside GPU memory anymore, GPUs use their Unified memory (UM) feature, where the CPU’s memory is used to extend the available memory to a CUDA kernel. Although UM allows running larger than memory workloads on the GPU, since it uses page faults to move data between the CPU and the GPU, the performance degrades exponentially. Even in modern architectures such as Grace Hopper that use System-managed memory (instead of Unified memory), the slow down is still quite significant.

To better understand and tackle this problem, we are working on characterizing the performance of vector search algorithms such as HNSW and IVF-Flat on modern CPUs (Intel Xeon Golds, Grace), GPUs (H100), and CPU-GPU systems such as Grace Hopper (GH200), exploring how distance calculations can be modeled to leverage the processing units efficiently, and looking for ways to accelerate the performance using intelligent prefetching and memory management techniques.