AUTHOR

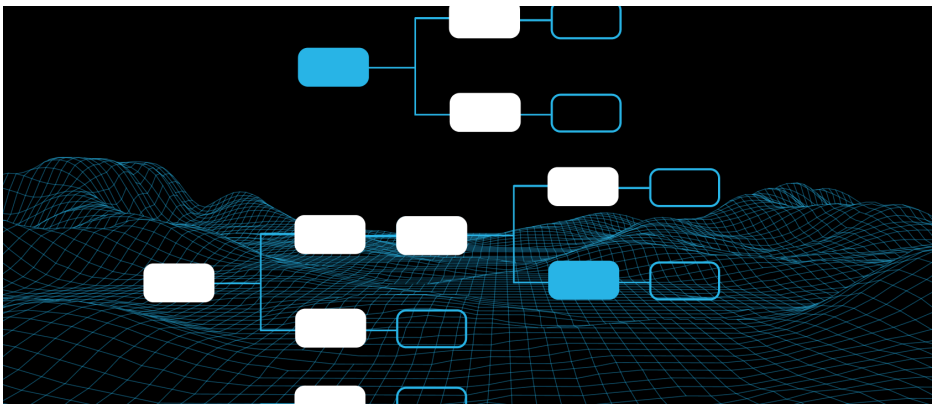
Cortex Search Team

AUG 08, 2024

Snowflake Cortex Search: High-Quality, Performant Search and Retrieval for Enterprise AI

SHARE

Gen AI



Search and retrieval systems have always been a critical backbone for knowledge management in enterprises. These systems cater to use cases ranging from site search, product catalog or feed search, to even internal search, enabling employees to quickly find the information they need. The recent infusion of large language models (LLMs) into enterprise tech stacks has brought renewed focus to search and retrieval solutions. LLMs are being used to sprinkle intelligence in various enterprise use cases — like customer service, financial research and sales chatbots — but they need a reliable partner in search to ensure that their intelligence is relevant, fresh and trustworthy. This is exemplified by the widespread adoption of the retrieval-augmented generation (RAG) stack to deploy contextualized LLMs in the enterprise. At the same time, while search and retrieval systems have traditionally relied on keyword signals, widespread research and adoption of representation models, otherwise known as embeddings and cross-encoders, have led to huge improvements in retrieval accuracy.

Cortex Search (in public preview), Snowflake's fully managed search service for documents and other unstructured data, is designed to be that reliable retrieval partner in an enterprise RAG stack. Cortex Search is built around the following guiding product principles:

Easy to use: fully managed infrastructure, so that customers are free of infrastructure and operations responsibilities. Cortex Search offers seamless, incremental ingestion with interactive, low-latency (sub-second) search results

State-of-the-art search quality: high-precision, high-recall ranked results out of the box, so that customers don't have to spend excessive time tuning search quality

Secure and governed: strong security and governance controls, so that customers can ensure that their RAG applications are secure and governed like the rest of their Snowflake data

In this post, we will take you backstage to see how we built Cortex Search to embody this vision to take on RAG workloads and beyond.

Guiding principles

Let's begin with the principles that guided us as we built Cortex Search. Vector search has greatly advanced the ability to determine how closely the content of a document matches with the meaning of the query. In fact, we have released world-class embedding models under Apache 2.0 licenses with our Arctic Embed **v1.0** and **v1.5** models. These are the world's most pragmatic text-embedding models for English-language search. However, our collective experience building production-worthy applications has taught us that "search" is more nuanced. There are a variety of query patterns to design around:

Sometimes users know exactly what they are looking for; they just don't remember the exact location or title. For instance, they could type: *"replication of dynamic tables in Snowflake."*

Keyword (lexical) searches, accompanied by traditional query rewrite and IR ranking techniques, do well in these cases.

Other times, queries are more exploratory – this is where vector search shines. For example, if the above query was "*data pipeline tools in Snowflake",* the user would receive semantically relevant results through vector search.

And many times, there is more to it than the query itself. An understanding of authority, recency, popularity, or other signals from beyond the strict contents of the document, help users ensure that desired results rank highly. For example, in the above queries, recently updated documentation on docs.snowflake.com may generally be preferable over old messages in a forum site.

Seeking to understand these nuances, we rigorously evaluated lexical, embedding and reranking search solutions against both external benchmarks as well as novel internal benchmarks developed to mirror customer workloads. We found that a multifaceted approach consistently outperforms other approaches. For example, as seen in Figure 1 below, on our internal benchmarks, we observed that NDCG@10 improves from 0.22 (Lexical) to 0.49 (Vector) to 0.53 (Hybrid) and 0.59 (Hybrid + reranker).

Figure 1. Incorporation of hybrid search and reranking leads to major improvements over a single faceted search system.

As seen above, to get high precision and recall from a large corpus of documents, we need to carefully combine vector signals; traditional IR techniques based on query understanding/rewriting and keyword searches; and signals beyond the text in the document. This hybrid retrieval approach is the backbone of Cortex Search, enabling it to achieve high search quality out of the box. Given a user search query, our backend performs:

Vector search for retrieving semantically similar documents

Keyword search for retrieving lexically similar documents

Semantic reranking for ranking the most relevant documents in the result set

Promotion/demotion of results based on signals, such as popularity and recency. To highlight the importance of these signals, one of our early customers saw their hit rate@1 go from 0.79 to 0.83 with the addition of popularity, and up to 0.86 with the addition of recency and popularity.

Figure 2. Hybrid retrieval approach in Cortex Search

In any practical scenario, high search quality for a query must be accompanied with low latency and competitive cost. We achieve this by obsessing over the efficiency of our system at every level. Our guiding principle here is that by adjusting the “effort” (compute and I/O) spent on a search query based on the “hardness” or “difficulty” of the query, we are able to optimize performance and cost. This principle manifests in the design choices we made throughout our system:

Multilevel scoring: We combine "lightweight" and "heavyweight" scoring algorithms and reranking strategies to balance query latency with ranking precision.

Index encoding, vector compression: We encode our "keyword search" (term) and vector indices using space-efficient approaches, such as **Matryoshka Representation Learning** and **vector quantization**, that also lend themselves to efficient retrieval and an efficient first pass at scoring that can incorporate key document-level signals.

Resource separation of indexing and serving: For low-latency queries, indexing creates optimized data structures. This process is characteristically different in terms of resource footprints than what is required to serve search requests (e.g., memory required, short bursts of high CPU usage). Thus, we run indexing and serving on different resources so that they do not compete with query serving and impact query latencies.

Separation of storage and compute for serving, horizontal scaling: When it comes to our indices, we store them separately from our compute resources. This enables us to quickly scale up or down on load and be responsive to movement across nodes for maintenance or disaster recovery.

Managing memory hierarchy: We carefully orchestrate what is kept in RAM, what on local SSD, and what on cloud storage to balance cost with query latency.

An equally critical design consideration for Snowflake Cortex Search is **security**:

We are built from the ground up on first principles: encryption at rest, secure key management, authenticated and encrypted communication (This includes support for **tri-secret-secure** encryption of data and **Virtual Private Snowflakes**).

Cortex Search services are schema-level objects in Snowflake and can be easily managed and integrated with the familiar role-based access control (RBAC) policies in a Snowflake account. This gives fine-grained control over query and administrative access to search services.

All Cortex Search operations, including our embedding and reranking models, run fully within the Snowflake perimeter.

How Cortex Search works

Let's see these principles in action, starting with your data residing in a Snowflake table or a stage.

Indexing

When a Cortex Search Service pointing at this data is created, the first step is to build efficient indexed data structures. During **indexing**, the source data is scanned to build two indices:

1. Text from each document is vectorized using an embedding model hosted within Snowflake, and a **vector index** is constructed. This vector index enables us to perform an efficient lookup of documents semantically closest to a given user query.

2. The same text is tokenized and a **term index** is constructed (term → list of documents). The term index enables us to perform an efficient lookup of all the documents that contain the terms present in the query.

In addition, we efficiently encode and incorporate any available document-level ranking signals and create secondary indices from specified attributes to support metadata filtering.

Worth noting is how these indices are kept up-to-date. Because of our tight integration into Snowflake data sources (tables, views, etc.), we provide **incremental indexing with data freshness** without users needing to manage their own data pipeline. Under the hood, Cortex Search creates a **dynamic table** that incrementally ingests and embeds changes in the underlying data source. Currently,

indexing runs on the customer warehouse in the customer account in a single-tenant fashion.

Indexing throughput is a key metric for this phase, and **efficient embedding generation** is critical to that. We use a model from the **arctic-embed family** for embedding our source data. These models offer high search quality at lower model sizes. In addition, embeddings are generated using Snowflake's **Cortex Functions**, which means whether you have 100 documents or 100 million we can scale the embedding generation and meet your needs.

Serving

Once the indices are built, they are served using our own **purpose-built retrieval backend**. This backend supports vector and keyword retrieval, and provides hooks for a layered scoring approach. Furthermore, this serverless backend automatically scales with the size of the data and in the future, with the volume of search traffic. Low-latency local storage (SSDs) cache the encrypted content, portions of which are paged into memory and decrypted as needed for query processing.

When a user query arrives, the first phase is that of query understanding. For example, we rewrite the query to expand the relevant document set ("OR" the query terms with their synonyms, perform stemming); we embed the query into a vector, etc. Retrieval then proceeds with a multistage scoring approach: We retrieve N, use in-memory signals to compute lightweight scores and take the top 1/Kth of these results. We then rerank these, using additional signals from storage, to compute heavyweight scores and take the top 1/Kth of these results. These results are then reranked using a **cross-encoder**, which considers the relevance of a document given a query, before returning the final results to the user.

Integrating Snowflake Cortex Search with your enterprise data

Behind the scenes, Cortex Search manages all the components (models and infrastructure) needed to facilitate this hybrid retrieval, thereby greatly simplifying what an end user must do. **One can unleash the power of search on enterprise data with a**

single SQL statement. This removes the burden of creating and managing multiple processes for ingestion, embedding and serving, ultimately freeing up time to focus on developing your application. Once the service is created, it's easy to query it from your application via REST or Python APIs. This includes both applications hosted in Snowflake (e.g., Streamlit in Snowflake) or applications hosted in an external environment. It's equally easy to observe the state of the system using tools you already know (e.g., show or describe).

Figure 3. The index definition and query APIs for Cortex Search

With Cortex Search, Snowflake customers get an easy-to-use, high-quality, and efficient search and retrieval system to power their enterprise AI workloads.

Looking forward

The public preview of Cortex Search marks only the beginning of Snowflake's journey in the AI-powered search ecosystem. We plan to continue to develop in the open, iterating and continuously delivering new capabilities to our customers. Namely, as we march towards general availability, we are honing in on:

- Increasing scaling capabilities
- Better customer-facing observability tooling
- Retrieval customization hooks for customers with advanced search use-cases
- Multilingual support

Of equal importance in our roadmap is the development of our ecosystem around search, including the data ingestion and app

development layers. To this end, we are working on easily ingesting data in different file formats (e.g., pdfs) and extracting signals that could prove significant in improving search quality. Similarly, we'll make it even easier to orchestrate RAG apps, with a higher-level "talk to your data" API.

We're on a mission to enable companies to securely unlock the full potential of their data with generative AI. The strength of our approach is our ability to innovate in every part of the stack, from models to infrastructure, while keeping the experience simple for end-users. As we continue to tune search quality and infrastructure in tandem, we will explore new directions and uncover optimizations in this space. If you're interested in joining us on this journey, please see open roles in our **careers page**.

Want to learn more?

Snowflake Cortex Search is in public preview in these **Snowflake regions** and rolling out soon to an extended set of global regions. Check out the following resources to get started:

Snowflake Cortex Search Tutorials

Demo: Employee Conversational Knowledge Base With Cortex Search

Snowflake Cortex Search Documentation

Check out these blogs for a more detailed look at our infra and modeling work:

Arctic-embed family of models

Arctic-embed v1.5

Inside embedding models: **data prep**, **training recipes**

Scalar quantization

RELATED CONTENT

JUL 23, 2024

Achieve Low-Latency and High-Throughput Inference with Meta's Llama 3.1 405B using Snowflake's Optimized AI Stack

Meta's Llama 3.1 405B represents a groundbreaking milestone for open-weight large language models (LLMs), pushing the boundaries of what's possible in AI by bringing frontier model capabilities to everyone. However,...

Full Details

JUL 30, 2024

Adaptive Network Optimizations for Faster Query Performance

At Snowflake, we strive to deliver "automatic performance" to all our customers. This performance is...

See how

JUN 17, 2024

Snowflake Arctic Cookbook Series: Instruction-Tuning Arctic

On April 24, we released Snowflake Arctic with a key goal in mind: to be...

More to follow

PLATFORM

Cloud Data Platform

Pricing

Marketplace

Security & Trust

SOLUTIONS

Snowflake for Financial Services

Snowflake for Advertising, Media, & Entertainment

Snowflake for Retail & CPG

Healthcare & Life Sciences Data Cloud

Snowflake for Marketing Analytics

RESOURCES

Resource Library

Webinars

Documentation

Community

Procurement

Legal

EXPLORE

News

Blog

Trending

Guides

Developers

ABOUT

About Snowflake

Investor Relations

Leadership & Board

Snowflake Ventures

Careers

Contact

Sign up for Snowflake Communications

diana.shaw@sn United States

By submitting this form, I understand Snowflake will process my personal information in accordance with their Privacy Notice. Additionally, I consent to my information being shared with Event Partners in accordance with Snowflake's Event Privacy Notice. I understand I may withdraw my consent or update my preferences here at any time.

SUBSCRIBE NOW