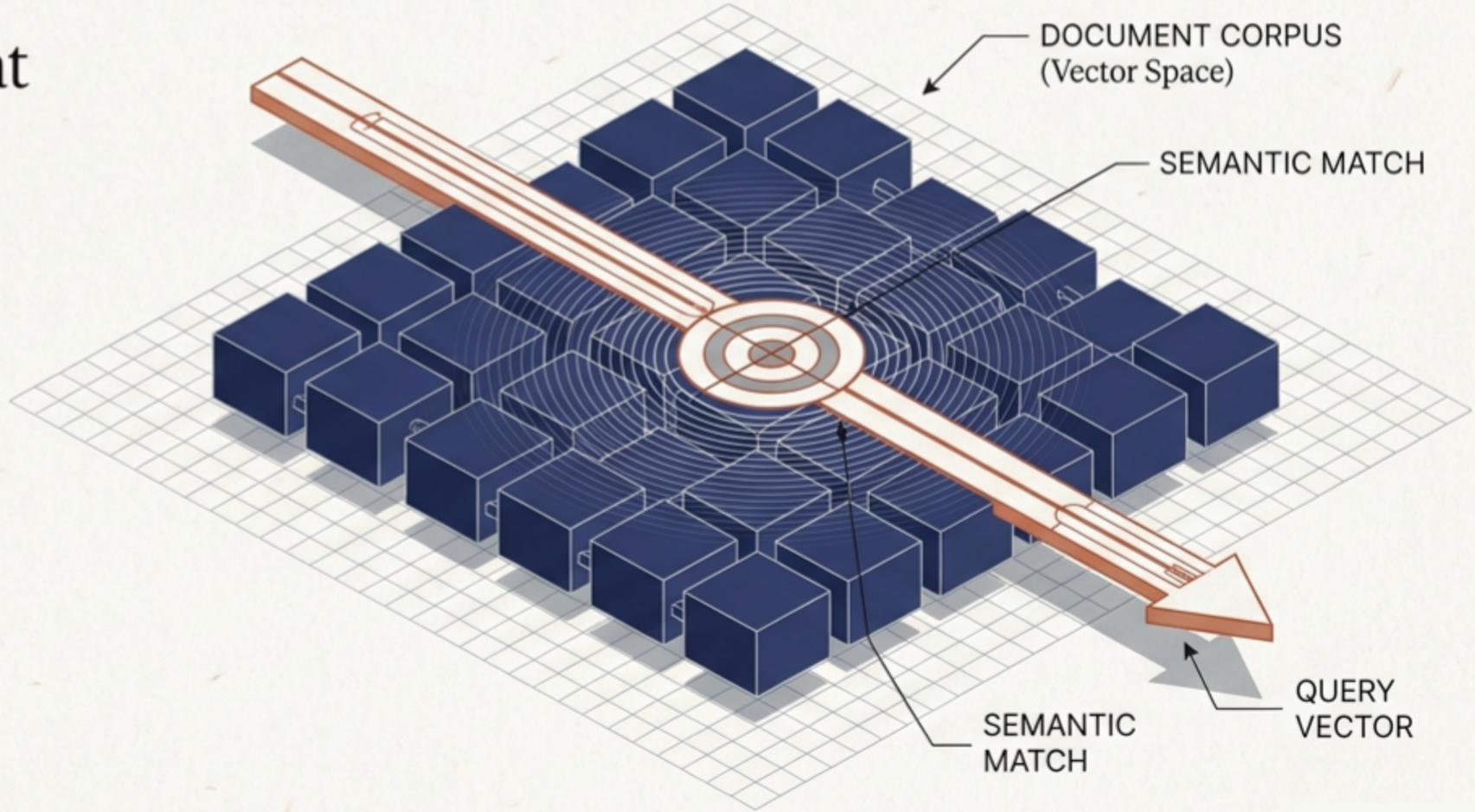
Dense Retrieval & Semantic Search

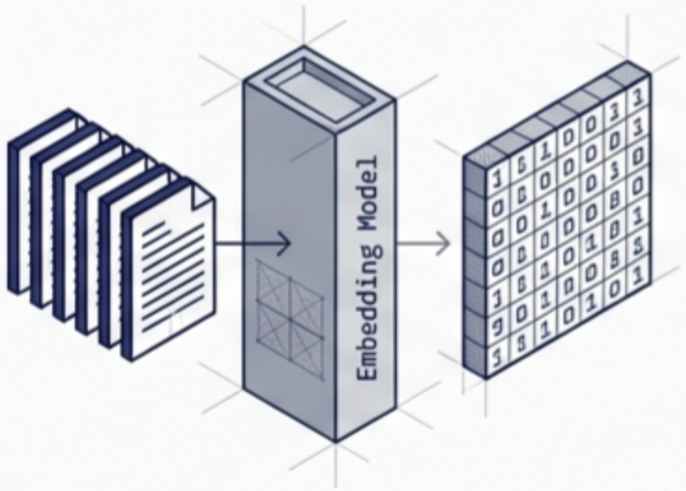
Building the engine that understands concepts, not just keywords.



The Three Steps of Dense Retrieval

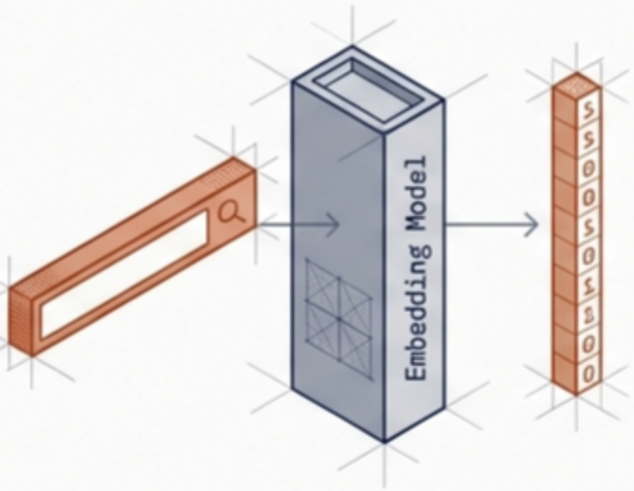
The intelligence lives entirely in the embedding model.

1. Embed Corpus



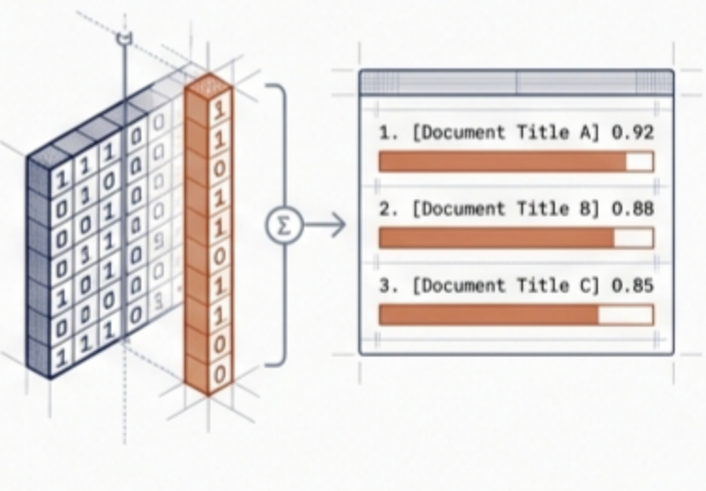
Embed every document once into a matrix.

2. Embed Query



Embed the query dynamically.

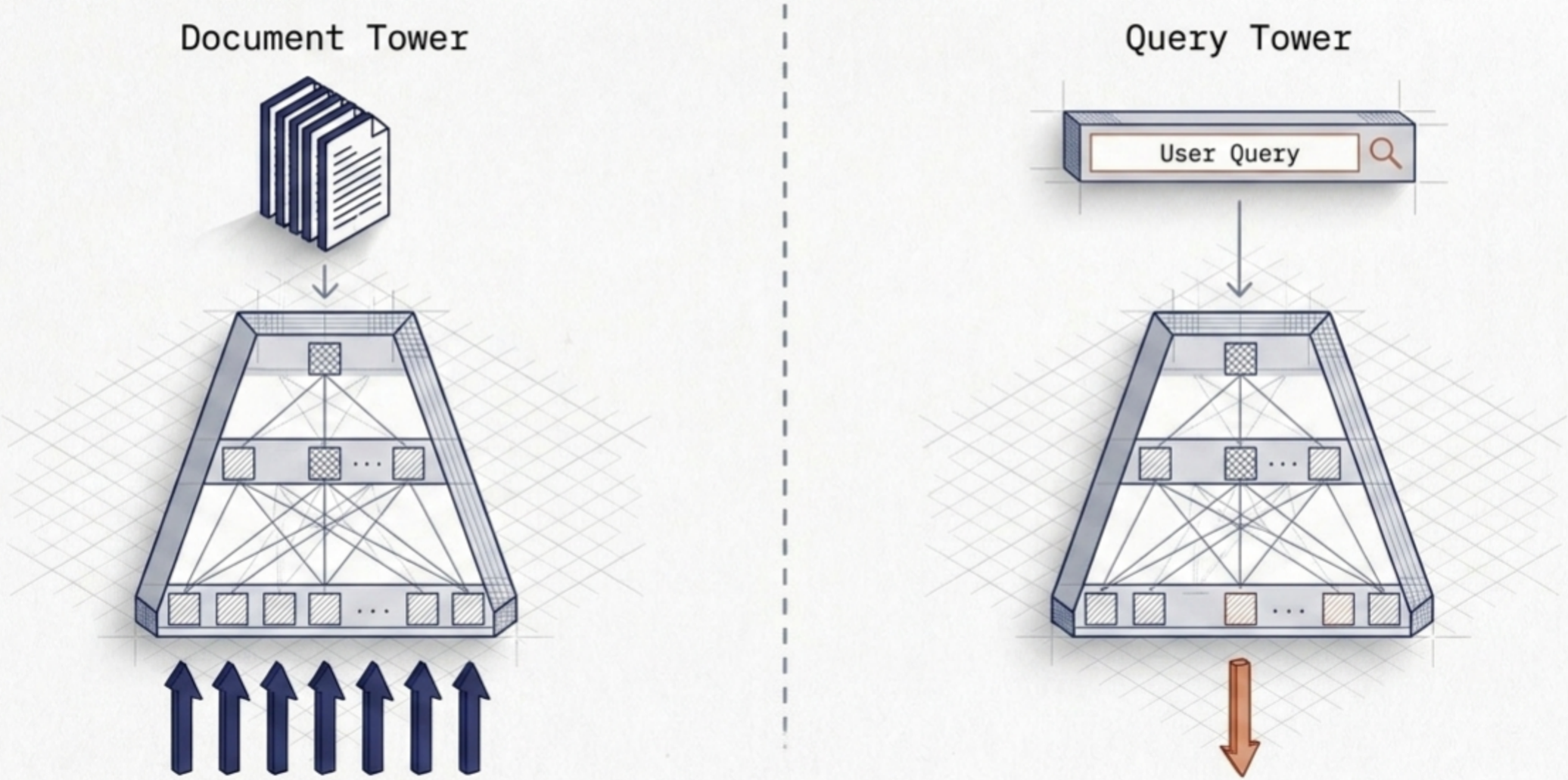
3. Score & Return



Score documents by cosine similarity and return the top-k nearest.

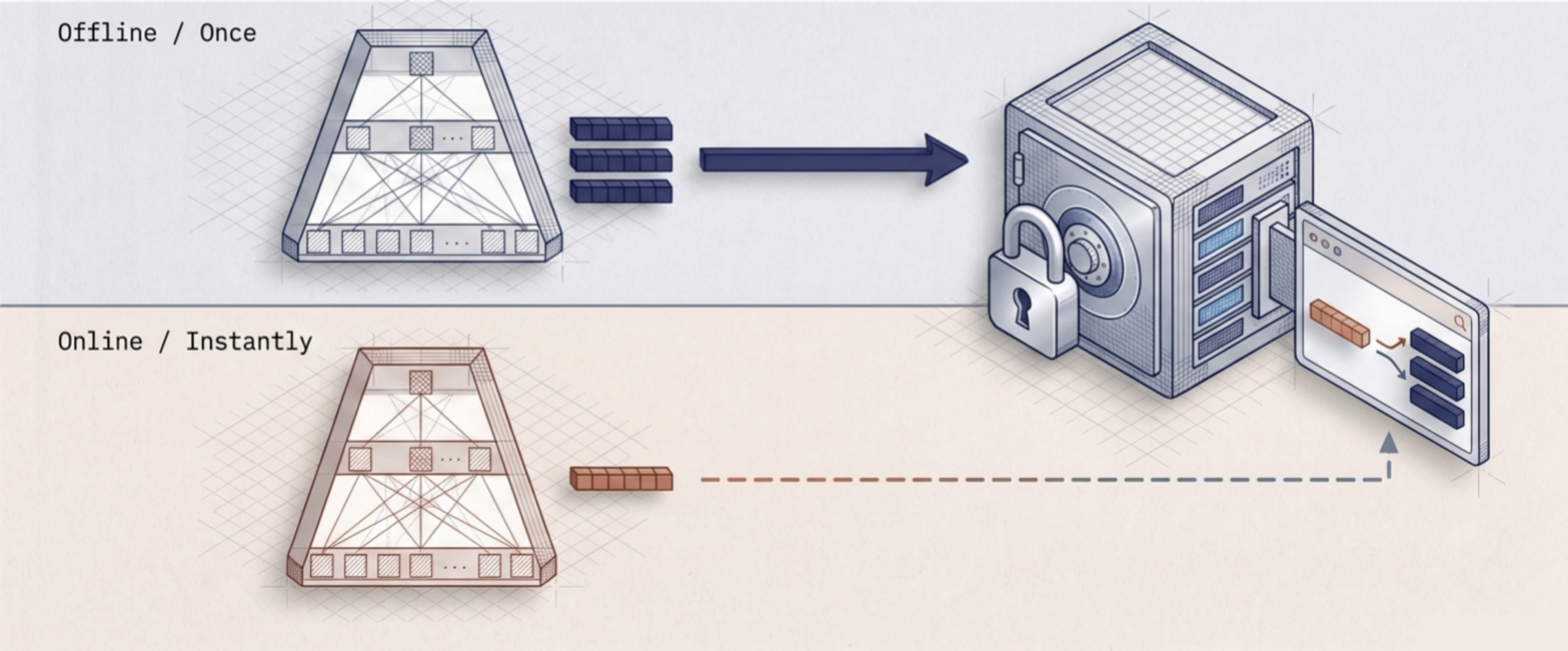
The Bi-Encoder Architecture

Embed queries and documents independently through parallel neural networks.



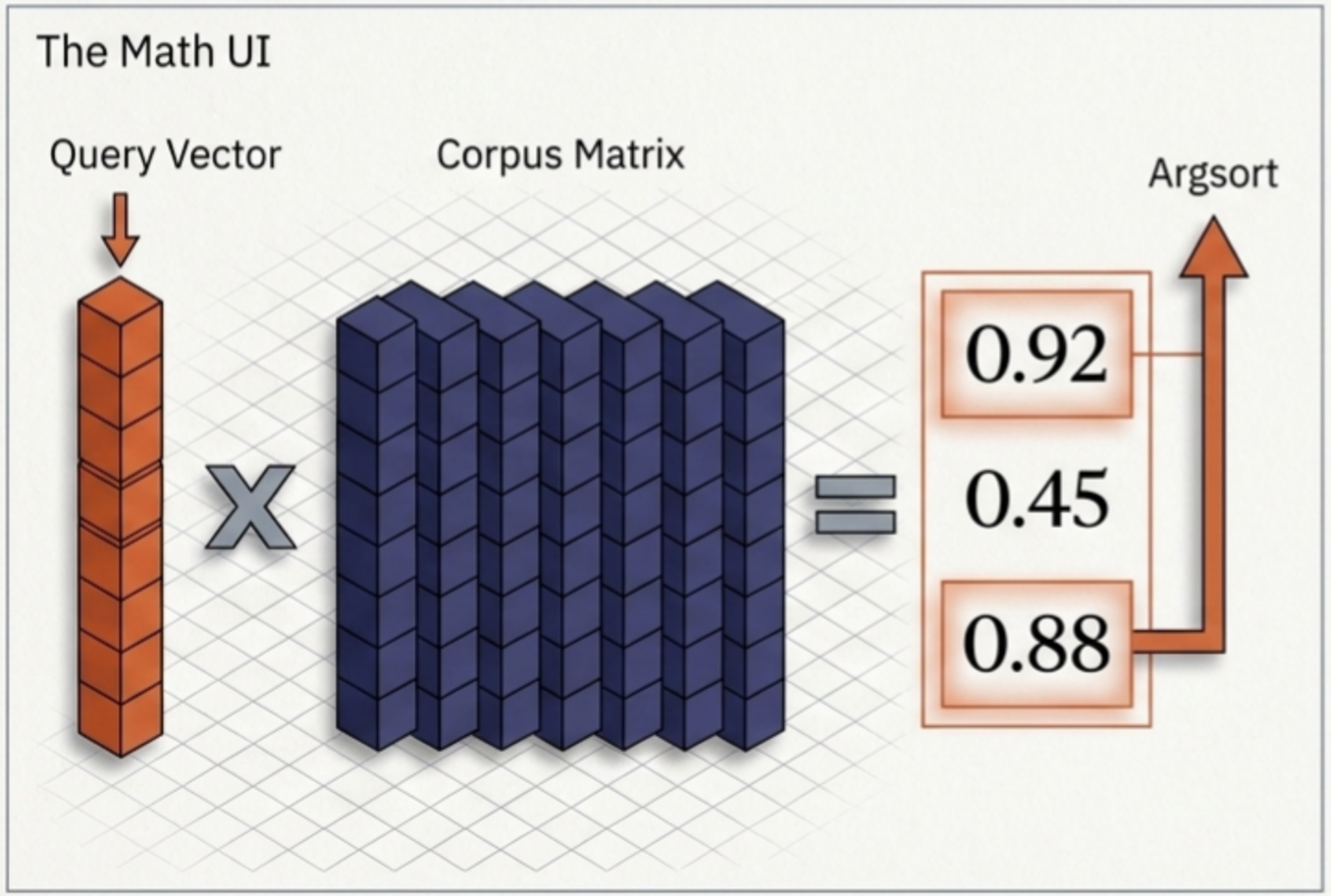
The Precomputation Advantage

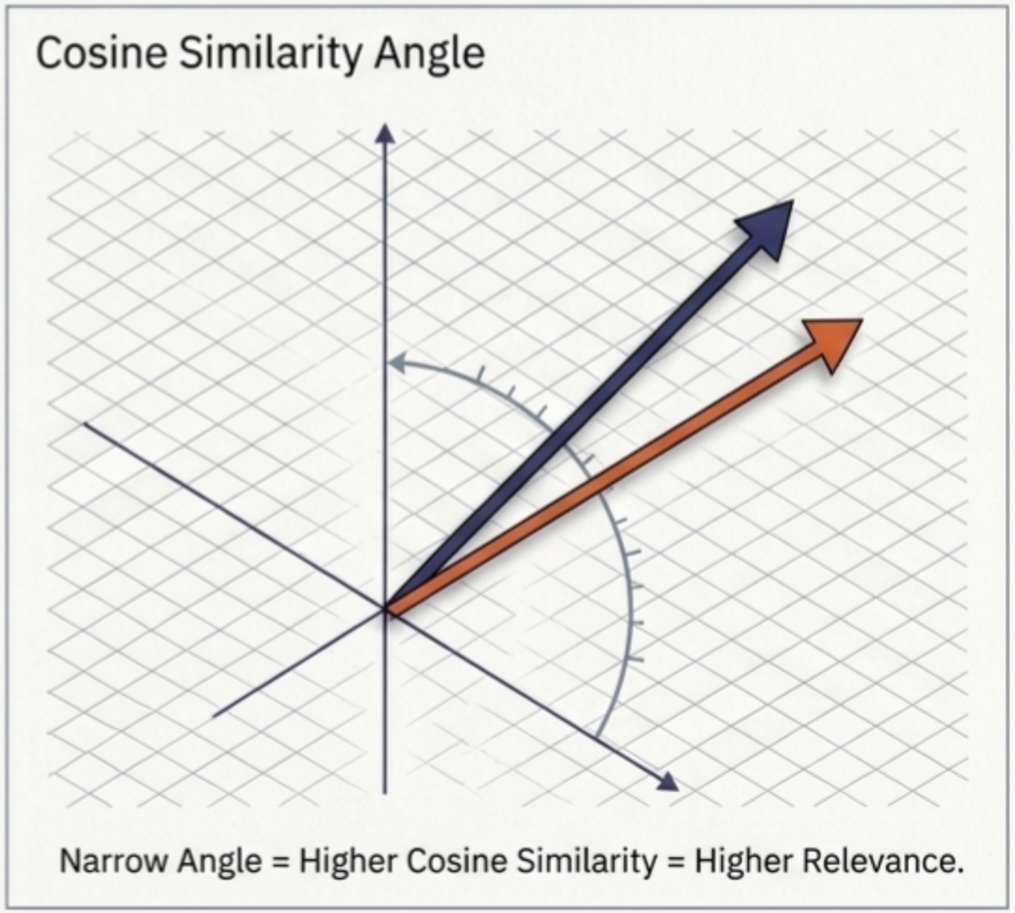
Because document vectors do not depend on the query, they can be computed once, saved, and reused. This is essential for scale.



Calculating Top-K Relevance

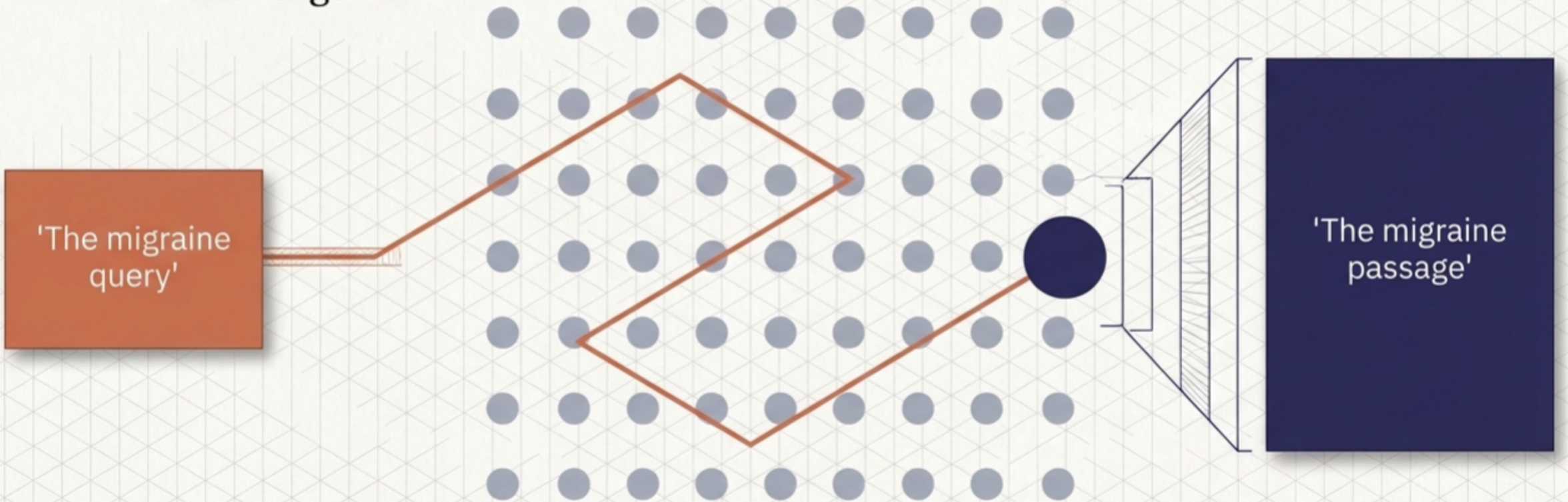
1. Build a normalized matrix.
2. Calculate Dot Product.
3. Argsort for Top-K.





Semantic Matching in Action

Dense retrieval maps concepts, allowing the system to bridge the gap between user intent and target text.



The Search Showdown: Dense vs. BM25

Neither is strictly better. The winner depends on the query on the query profile.

Dimension	Dense Retrieval	Traditional BM25
Matching Type	Concept & Paraphrase	Exact Keyword
Core Strength	Connects meaning	Precision on rare terms
Weakness	Struggles with highly specific jargon	Fails on synonyms
Bottom Row (The Decider)	‘a large ocean animal’ –> blue whale	Exact rare token (product code XYZ)

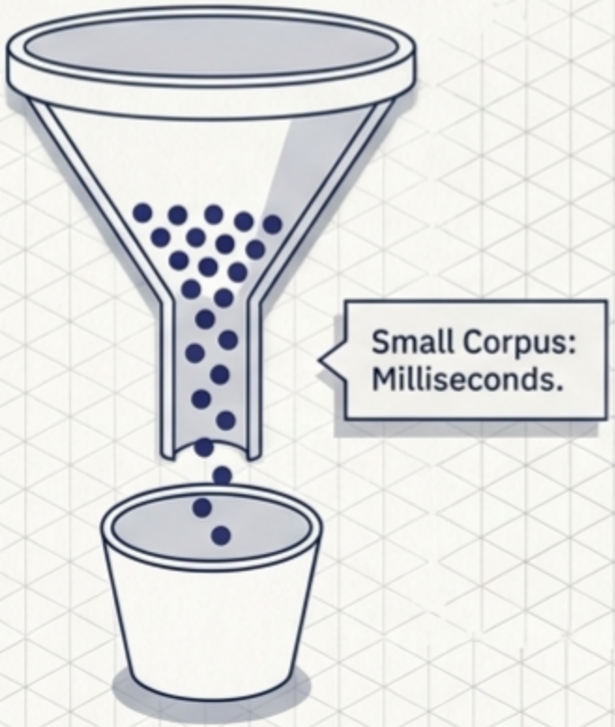
WINNER

WINNER

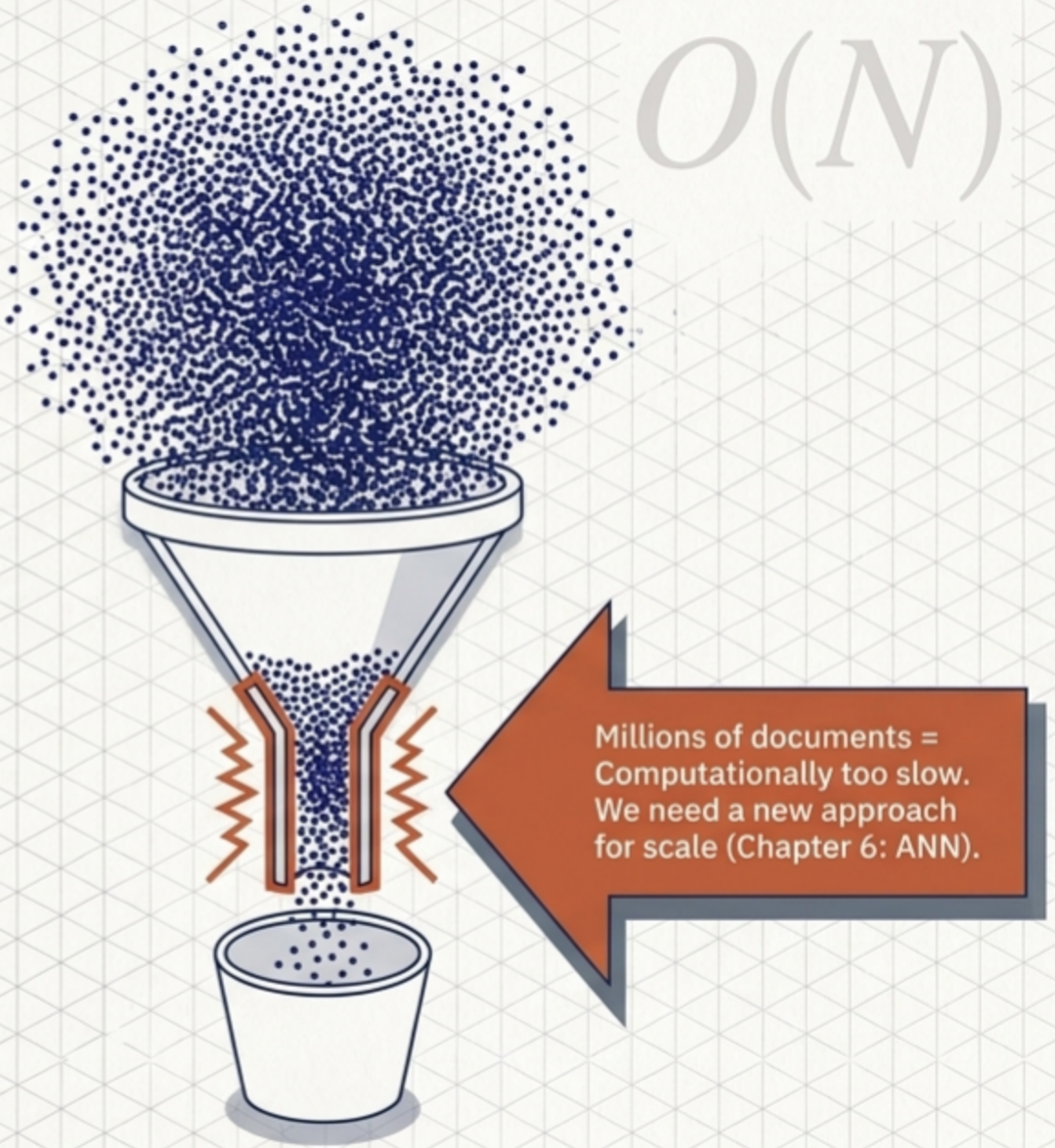
Scoring the query against every single document is an O(N) operation.

$O(N)$

The O(N) Brute Force Funnel

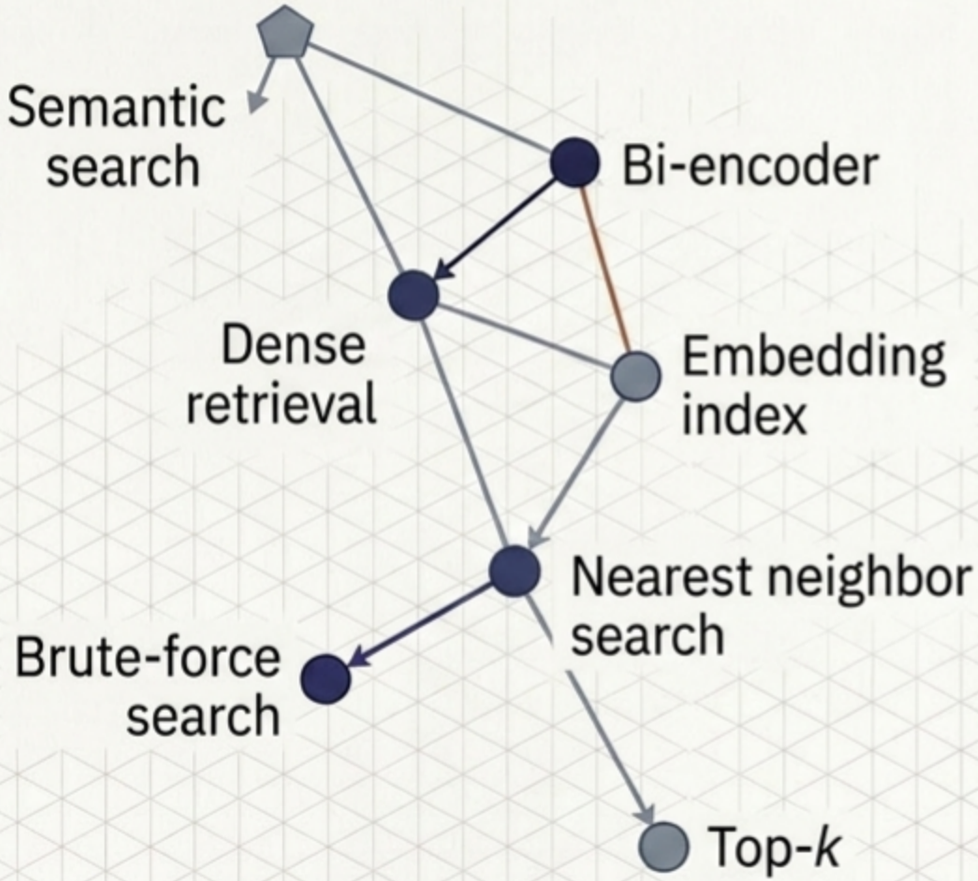


Small Corpus: Milliseconds.



Millions of documents = Computationally too slow. We need a new approach for scale (Chapter 6: ANN).

Key Terms



Review & Synthesis

Q: State the three steps of dense retrieval.

A: Embed corpus once; embed query; return nearest vectors.

Q: Why can a bi-encoder precompute document vectors, and why does it matter?

A: Embedded independently of the query = reusable = fast large-scale search.

Q: Give a query where BM25 beats dense retrieval.

A: Exact rare tokens (product codes, library names).