Blaming embeddings is blaming the entire AI game

Who keeps underrating the power of cosine similarity and RAG missed the core of Generative AI as a whole.

8 min read · May 20, 2025

Fabio Matricardi

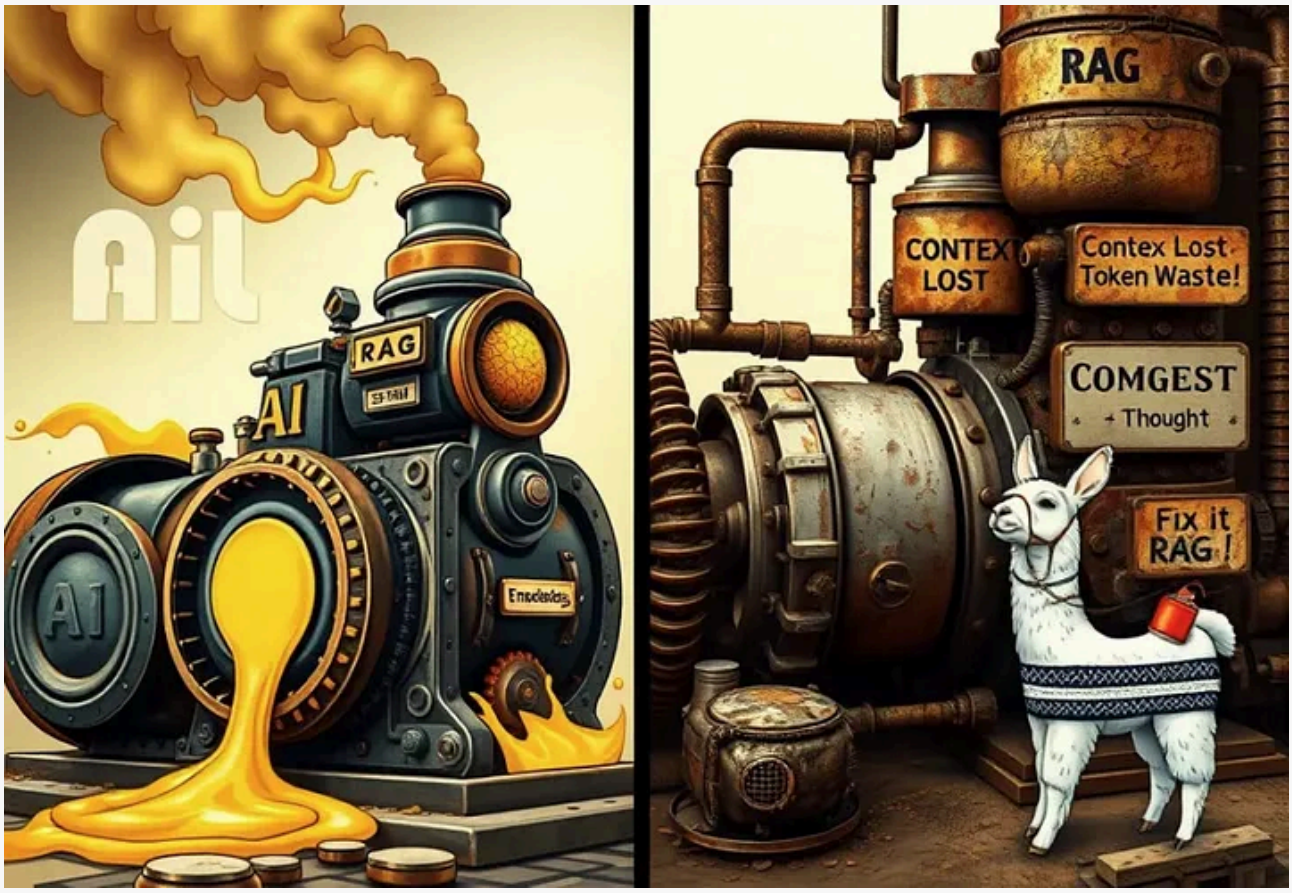


image by the author and Flux — using this method

Anyone dissing embeddings is missing the point of AI itself.

Trashing embeddings & RAG is like saying your engine doesn't need oil

A curious narrative is gaining traction in some corners of the AI discussion: a downplaying of fundamental components like embeddings, cosine similarity, and even the well-established Retrieval Augmented Generation (RAG) architecture.

Some suggest that the in-context learning (ICL) capabilities of modern Large Language Models (LLMs) are so potent that these other elements are becoming secondary, even unimportant.

> *This perspective, however, is so damn wrong!*

It misunderstands how these technologies interlink and, at sadly, how Generative AI models like GPT perceive and process information at their very core.

To dismiss these elements is to miss the Foundation of the entire AI game.

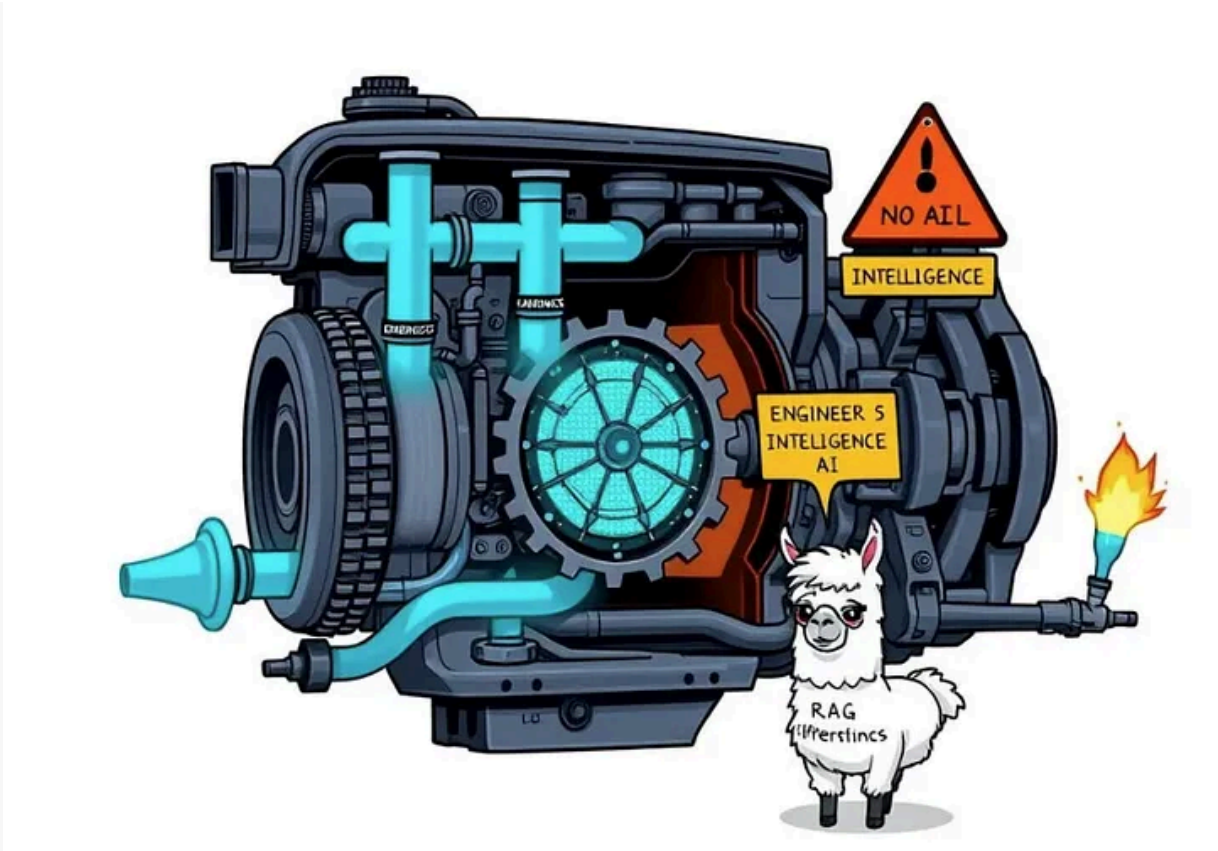


image by the author and Flux — using this method

Embeddings — the Lingua Franca of LLMs

At the heart of any Generative Pre-trained Transformer (GPT) model, and indeed most advanced neural networks dealing with language, lies the **embeddings layer**. This isn't only a peripheral feature; the embedding layer is the initial gateway through which all textual input must pass to be understood by the machine.

Back to the Embeddings...

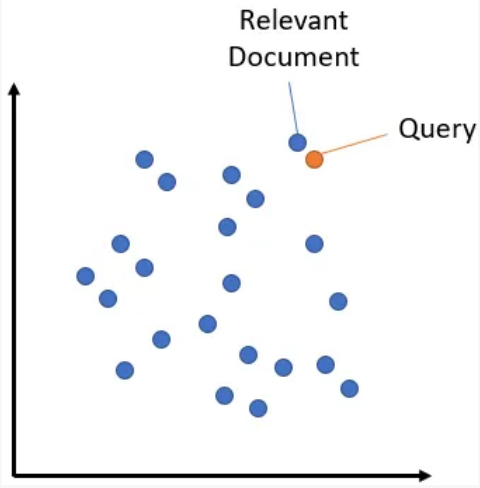
When you provide text to a GPT model, it first undergoes tokenization, breaking the input into words or sub-word units. These tokens, initially just discrete identifiers, are then transformed by the embedding layer into dense numerical vectors. These vectors, or "embeddings," achieve several critical functions:

1. **Numerical representation:** They convert symbolic language into a mathematical format that neural networks can process.
2. **Capturing semantic meaning:** This is where the *kind of magic* begins. During their extensive pre-training, these models learn to **assign similar vector representations to tokens that share similar meanings or appear in similar contexts.** "King" and "Queen" will have vectors that are closer in this high-dimensional space than, say, "King" and "Carburetor." This semantic encoding is vital for the model to grasp nuance, context, and relationships.
3. **Dimensionality reduction:** Compared to older methods like one-hot encoding (which would create astronomically large and sparse vectors), embeddings offer a much lower-dimensional yet richer representation, making computations feasible and learning more effective.

The output of this embedding layer, often combined with positional encodings (**to give the model a sense of word order**), is the actual input that flows through the subsequent complex transformer blocks.

So, to downplay the significance of embeddings is to ignore the very mechanism that allows an LLM to begin understanding language in the first place.

This same powerful concept of representing meaning as vectors is then extended for use in RAG systems. Here, dedicated embedding models (often themselves sophisticated neural networks) convert chunks of external documents and user queries into these meaningful vector spaces.



Cosine Similarity — finding meaning in the Matrix

Once we have these text embeddings, we need a way to find which documents are relevant to a user's query (also represented as a vector).

In a RAG context where we have a vast database of document chunks represented as vectors, embeddings are the game changer. This is where **cosine similarity** comes into play.

Cosine similarity is a metric that measures the cosine of the angle between two vectors. A value closer to 1 indicates that the vectors are pointing in roughly the same direction, implying high semantic similarity between the corresponding text snippets. A value near 0 suggests little similarity, and -1 would indicate opposite meanings.

Is cosine similarity a perfect, infallible measure of relevance in every conceivable scenario? Perhaps not.

Nuance can sometimes be missed, and more advanced retrieval techniques, including hybrid search or re-ranking layers, are continually being developed.

But again, regardless of the technique, I still believe that the main problem lies somewhere else!!!

AI is only as good as its diet

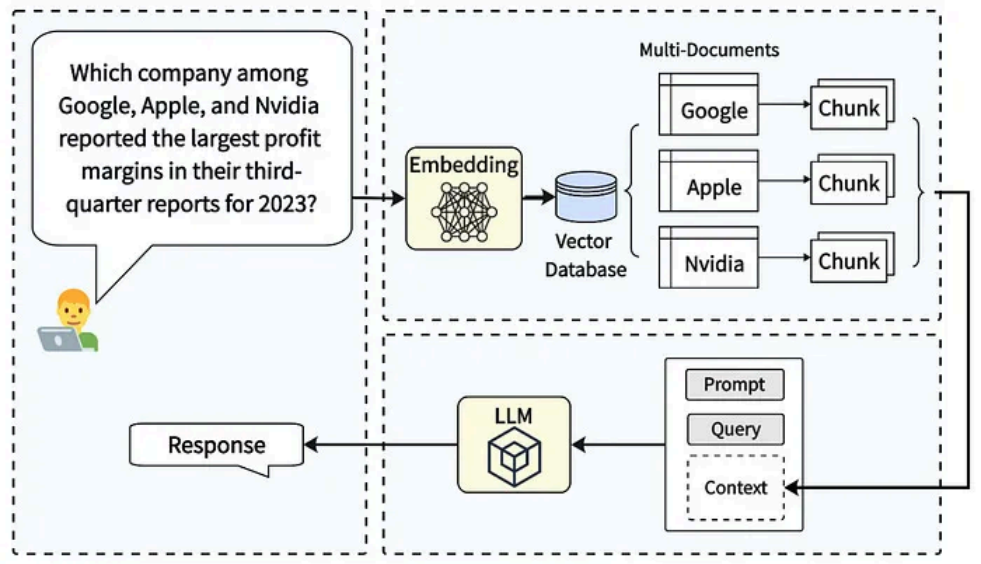
Retrieval Augmented Generation is still the best option, even with 10 million tokens context. Because Garbage In...

medium.com

However, to label it "not important" is a gross mistake.

For the task of efficiently sifting through potentially millions or even billions of vectorized document chunks to find those that are semantically aligned with a query, cosine similarity is a robust, efficient (no waste of tokens), and effective tool.

It's the workhorse that powers the "Retrieval" in RAG, allowing the system to quickly plug in potentially useful information.



Retrieval-Augmented Generation from [2]: Yixuan Tang, Yi Yang: MultiHop-RAG: Benchmarking Retrieval-Augmented Generation for Multi-Hop Queries (2021), arXiv — CC BY-SA 4.0

Retrieval Augmented Generation (RAG) — beyond internal knowledge

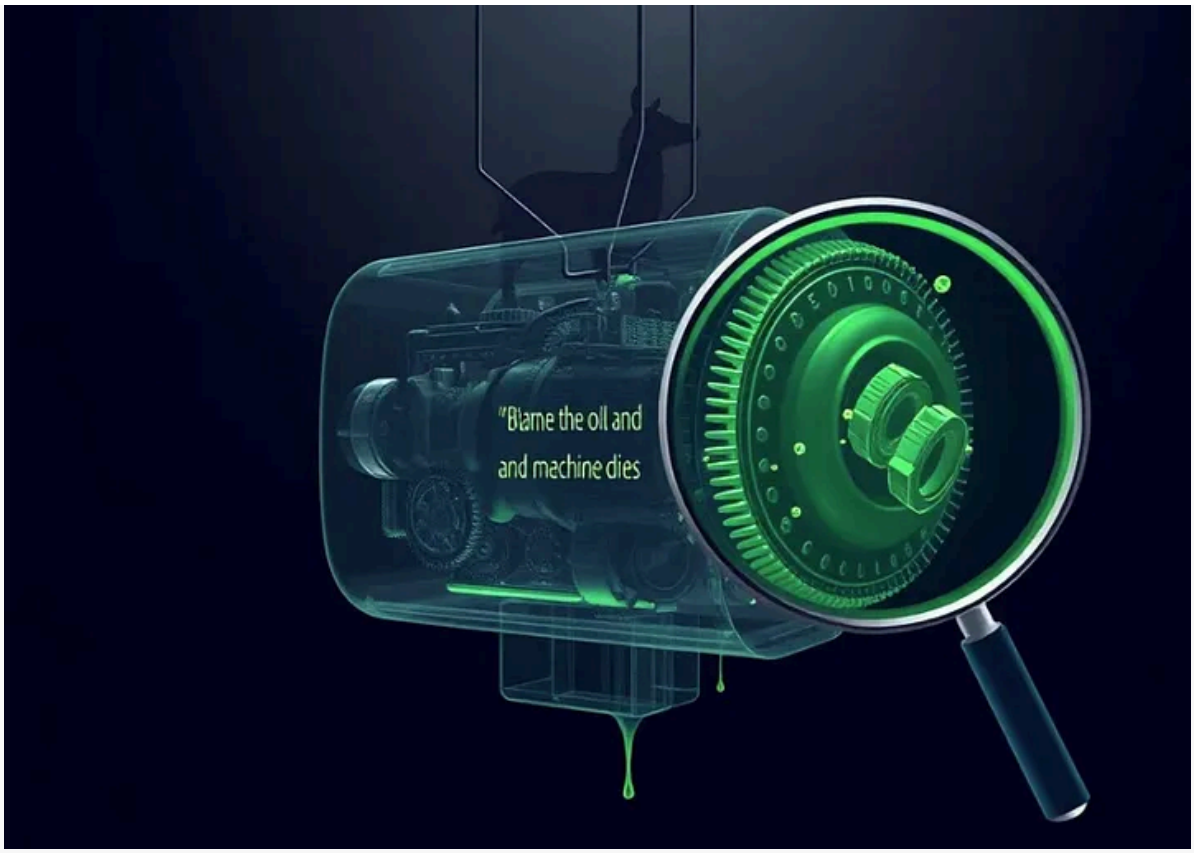
Retrieval Augmented Generation (RAG) is an architecture designed to make LLMs more factual, up-to-date, and relevant by connecting them to external knowledge sources.

The process is straightforward yet powerful, and I will try to make it easier as possible:

1. A user poses a query.
2. The query is embedded, and this embedding is used (often via cosine similarity) to search a vector database of pre-embedded document chunks.
3. The most relevant chunks are retrieved.
4. These retrieved chunks, along with the original query, are fed as an augmented context to the LLM.
5. The LLM generates a response based on this combined information.

Those who suggest abandoning RAG overlook critical limitations inherent in LLMs if used in isolation.

- **Knowledge cutoff:** LLMs are trained on data up to a specific point in time. For instance, as of today, May 9, 2025, a model trained only until late 2023 would have no knowledge of events or information created since. RAG allows LLMs to access and utilize information far newer than their last training date.
- **Limited context window:** While context windows are expanding, they are still finite. You cannot cram an entire enterprise's internal documentation, a comprehensive medical encyclopedia, or the live internet into a single prompt. RAG allows the LLM to draw from vast external repositories on demand.
- **Accessing proprietary or specific data:** General-purpose LLMs don't know your company's confidential data, specific project details, or niche domain knowledge.
- And if you are using documents protected by copyright or NDA… RAG is the bridge to make this information accessible and usable by your local LLM.
- **Reducing hallucinations & improving verifiability:** when we ground the LLM's response in specific retrieved documents, RAG significantly reduces the tendency for models to "hallucinate" or generate **plausible but incorrect information**. Not to mention that it allows for citations, providing users with sources for the generated claims.



The Misconception: is In-Context Learning (ICL) truly enough?

In-Context Learning (ICL) is indeed a phenomenal capability. It allows LLMs to learn tasks and adapt their responses based on examples and instructions provided directly within the prompt at inference time, without needing to be retrained.

However, ICL is not a replacement for RAG; they are highly complementary:

- **ICL needs content to learn from:** RAG provides the *relevant, external content* that the LLM then uses its ICL abilities to process, synthesize, and incorporate into its answer. **Without RAG, ICL is limited to the information** explicitly typed into the prompt or the model's pre-existing (and potentially outdated or generic) knowledge.
- **Scale and Specificity:** ICL cannot overcome the context window limitation for vast knowledge bases. RAG provides the targeted snippets from these bases; ICL then helps the model understand *how* to use those snippets to answer the specific query.

. . .

Conclusions

Blaming embeddings is like blaming the alphabet for poorly written prose. **Embeddings are the fundamental building blocks for language understanding in modern AI**.

Cosine similarity, while not the only tool, is a powerful and practical method to find relevant and semantic accurate information created by these embeddings. And by the way, it is the way every recommendation system works!

RAG, in turn, makes use of both to feed LLMs with knowledge beyond their training data, making them significantly more potent and reliable.

To underrate these components is to misunderstand the elegant and deeply interconnected architecture that allows Generative AI to perform its almost magical feats.

> *They are not an optional extra: they are the core machinery that drives the current AI revolution.*