



# Smarter Search, Better Intelligence

When it comes to search, no one has a tougher job than a US government intelligence analyst. The pool of classified documents alone is growing at an estimated rate of 50 million documents per year. On top of that, analysts are expected to stay on top of relevant OSINT data streams from news and social media. If your area of responsibility is Ukraine, for example, the morning “traffic” of relevant information can grow to hundreds of classified documents plus even more OSINT.

And that’s just the new information that came in overnight, which you must somehow absorb before the day’s meetings begin. But how does all this information get to your desk in the first place?

It all begins with search. Analysts don’t have friendly gnomes working on their behalf, reading every document and deciding whether it’s worth their attention. To find relevant information, you need to search for it.

---

The documents waiting to be read every morning by intelligent analysts are the results of automatically running search queries. Below we'll compare the traditional "boolean" search method with newer methods made possible by machine learning, and finally the newest tricks that require advanced artificial intelligence (AI).

Each of the methods have advantages, disadvantages, and gotchas that are easy to misunderstand. Let's dive in!

### The old faithful method: Boolean search

Pros	Cons
Based on a simple and explainable algorithm	Very difficult to build good queries
Very mature technology	Even good queries are brittle

For decades, boolean search was the only option for analysts to find information. The name "boolean" comes from the granddaddy logician [George Boole](#). You build a query with search terms that include logical operators: AND, OR, NOT.

Boolean search uses a clever algorithm based on word overlap between documents and your query. A document that mentions words and phrases in your query is more likely to be relevant, so it gets a higher score. The concept is that simple. But crafting a good boolean query is anything but simple. Take a look at the query below. The intention is to find documents about Ukraine's diplomatic relations with NATO member states. Why must it include the names of all these countries, cities, and political leaders? Because a relevant document might describe a single NATO member country's relations with Ukraine without mentioning the word "NATO". And it might describe diplomatic relations without using the word "diplomatic".

```
( ( "Ukraine foreign relations"-5 OR "Ukraine diplomatic relations"-5 OR "Ukraine diplomacy"-5 OR "Ukraine international affairs"-5
OR "Ukraine foreign policy"-5 OR "Ukraine international diplomacy"-5 OR "Ukraine-Russia conflict"-5 OR
"Ukraine military cooperation"-5 OR "Ukraine security cooperation"-5 OR "Ukraine defense cooperation"-5 OR
"Ukraine military equipment"-5 OR "Ukraine arms trade"-5 OR "Ukraine sanctions"-5 OR "Ukraine peace negotiations"-5 OR
"Ukraine conflict resolution"-5 OR "Ukraine ceasefire agreement"-5 OR "Ukraine regional security"-5 OR "Ukraine alliance building"-5
OR "Ukraine strategic partnership"-5 OR "Ukrainian foreign relations"-5 OR "Ukrainian diplomatic relations"-5 OR
"Ukrainian diplomacy"-5 OR "Ukrainian international affairs"-5 OR "Ukrainian foreign policy"-5 OR
"Ukrainian international diplomacy"-5 OR "Ukrainian military cooperation"-5 OR "Ukrainian security cooperation"-5 OR
"Ukrainian defense cooperation"-5 OR "Ukrainian military equipment"-5 OR "Ukrainian arms trade"-5 OR "Ukrainian sanctions"-5 OR
"Ukrainian peace negotiations"-5 OR "Ukrainian conflict resolution"-5 OR "Ukrainian ceasefire agreement"-5 OR
"Ukrainian regional security"-5 OR "Ukrainian alliance building"-5 OR "Ukrainian strategic partnership"-5 ) AND (
North Atlantic Treaty Organization ) AND ( United States OR United Kingdom OR Ukraine OR England OR
"Britain" OR France OR Germany OR Canada OR Italy OR Spain OR Poland OR Netherlands OR
Belgium OR Greece OR Turkey OR Denmark OR Norway OR Portugal OR Hungary OR Czech Republic
OR Slovakia OR Romania OR Bulgaria OR Estonia OR Latvia OR Lithuania OR Slovenia OR Croatia
OR Albania OR Montenegro OR Macedonia OR Bosnia And Herzegovina OR Serbia OR Kosovo OR Cyprus
OR Washington, D.C. OR London, England OR Paris, Ile-De-France OR Berlin OR Ottawa, Ontario OR
Rome, Latium OR Madrid OR Warsaw, Mazovia OR Amsterdam, North Holland OR Brussels, Brussels Capital OR
Athens, Attica OR Ankara OR Copenhagen, Capital Region OR Oslo OR Lisbon OR Budapest OR
Prague, Praha OR Bratislava, Bratislavsky OR Bucharest, Bucuresti OR Sofia, Sofia-Capital OR Tallinn, Harjumaa OR
Riga OR Vilnius OR Ljubljana OR Zagreb, City Of Zagreb OR Tirana, Tirane OR Podgorica OR Skopje, Karpos
OR Sarajevo, Federation Of B&H OR Belgrade, Central Serbia OR Pristina OR Nicosia ) AND ( Vladimir Zelensky OR
Volodymyr Zelenskyy OR Volodymyr Zelenskiy OR Volodymyr Zelenskiy OR Ruslan Stefanchuk OR Ivan Bakanov OR
Dmytro Razumkov OR "Oleksandr Danyliuk" OR Ihor Zhovkva OR Oleksiy Arestovych OR "advisor" OR "adviser" )
AND NOT ( title:(live updates) OR title:(ukraine latest) )
```

Good boolean queries are handed down from analyst to analyst like family jewels. And it remains the core search method today. Despite the complexity, it does have advantages.

Boolean search is very precise. Consider an analyst tracking a known threat actor that has a unique alias. Or intelligence about black market sales of a weapon with a unique codename. It is hard to beat boolean search for finding documents about those entities.

Another advantage is that boolean search is explainable. It is not a black box neural network. Its straightforward rule-based word-counting algorithm makes intuitive sense to a non-expert. When a non-relevant document appears at the top of your search results, it is usually obvious how it got there. For example, when searching for documents about Apple the company, it's not mysterious when you get a document about the fruit industry that mentions the word "apple" many times. The non-mysterious solution is to add "NOT fruit" to your query to exclude those false-positive documents. That iterative process is how pages-long boolean queries get crafted.

But the major drawback, besides the labor of building queries, is that it is brittle. A change in the spelling of a word can make a relevant document invisible to your search query. And some concepts simply do not have a reasonably small set of search terms. Imagine looking for weapons and having to build an exhaustive list of weapon names into your query.

What boolean search is missing is semantics. The algorithm has no understanding of the meaning or intention of your query.

## The magical method: Semantic search

Pros	Cons
Queries are natural and easy	Fast evolving technology that is more complex and expensive to implement than boolean
Queries are robust to data drift	Far less understandable and explainable than boolean

Back to our Ukraine analyst working on a report about diplomatic relations with NATO member states. Rather than crafting a page-long boolean query to search for docs containing certain words, semantic search makes it possible to just describe what you seek: "Ukraine's diplomatic relations with NATO member-states".

## Here's how semantic search works, in a nutshell.

First you take your corpus of documents and break them into smaller text chunks, typically about the size of a paragraph. Then you pass each chunk through a language model, much smaller than the large language models that power systems like ChatGPT, but essentially the same. The output is a list of several hundred numbers, usually called a vector or an embedding. Each embedding is a mathematical representation of the semantic meaning of the text chunk. (This is the magical part of semantic search that is hard to explain and also hard to understand. Language models are indeed pretty magical.)

With your documents indexed into a vector database, you can now do semantic search. When a user writes a query — "Ukraine's diplomatic relations with NATO member-states," for example — you treat it as just another chunk of text. It goes through the language model's neural network and out pops an embedding.

The final step is much less magical. To find document chunks likely to be relevant for the query, we use reliable old nearest neighbor search algorithms. Text chunks that are indeed about Ukraine's diplomatic relations will tend to be very close in the embedding space to a query about the same.

We have seen first-hand how semantic search changes the daily reality for intelligence analysts as they describe it as pure joy, like gaining the ability to fly. What used to take hours of effort to craft a single boolean query can now be accomplished as quickly as you can write a sentence.

However, semantic search has its limitations. Its performance is only as good as the training of the language model generating the embeddings. For example, if our Ukraine analyst were searching for documents about new types of weaponized drones being used in the conflict, semantic search may struggle. Language models are trained on massive samples of text — most of it scraped from the internet — that was available up to the date of training. So there will always be entity names, jargon terms, and especially unique codes that come into use after model's training. The embedding space is smart, but not smart enough to guess that some arbitrary string of letters and numbers is the name of a drone used in Ukraine, unless the text chunk provides enough context clues.

Another gotcha is that most first-time users of semantic search expect it to be more powerful than it actually is. These great expectations are thanks to Google Search and, increasingly, to ChatGPT. More on that below. It can be a frustrating experience to ask a semantic search system to “find me docs published today about cybersecurity in my area of responsibility” and get back nothing relevant. Using semantic search requires some good intuitions about how it works, what it can and can’t do.

Expert analysts typically jump back and forth between boolean and semantic search, depending on what they seek. But what if you didn’t have to?

### The cake and eat it too method: Hybrid search

Pros	Cons
Makes a “single search bar” experience possible	So new it’s still an active research area
Can find jargon and entities as well as text that matches the semantic meaning of the query	Makes the final search ranking even harder to understand and explain

When intelligence analysts go home from work, they search for information on the internet just like the rest of us. Using a system like Google Search is deceptively simple. It’s just a single search bar. No matter how you word your query, highly relevant information is usually at the top of the search results.

There is a lot going on under the hood to make this possible. One huge advantage that Google has is its vast population of users. Chances are, a great many people have made the same query before you. That allows Google to keep track of click-throughs and adjust its relevancy scoring. Human users are doing a lot of the work required to make the search experience great.

But there is some real magic that makes a single search bar experience possible on proprietary data. By treating the user’s query as both a boolean and a semantic query and running both, the results can be combined. This hybrid search approach can combine the best of both worlds. Here’s a technical walk through of one version of hybrid search by our partners at Microsoft.

Most of the disadvantage of hybrid search is the flip side of its advantages: You take control away from the user. Intelligence analysts are comfortable with complexity, and they tend to prefer control and explainability over convenience.

However, there are deeper technical limitations. Translating the natural language phrase of a semantic query into a boolean equivalent is not trivial. Not every word should be given equal weight, and some should be excluded. So builders of hybrid search systems are starting to use large language models to handle this translation layer. It is an active area of research, and still rare in off-the-shelf search systems.

### Adding an LLM on top: Retrieval Augmented Generation (RAG)

Pros	Cons
If the query is a question, you get not just docs but a direct answer	Anything involving an LLM significantly increases latency and cost
Time to insight improves massively	Opens to door to hallucination errors

Even the best search system in the world can only do one thing: Find relevant documents. You still have to read those documents to find the information you seek. For intelligence analysts, this is the ultimate bottleneck.

A typical human can read and comprehend text at about 300 words per minute. Many intelligence analysts can speed-read at double that rate. But with hundreds of documents to read and only so many hours in the day — and brain power depleting steadily — choosing which documents to read becomes as important as searching for them in the first place.

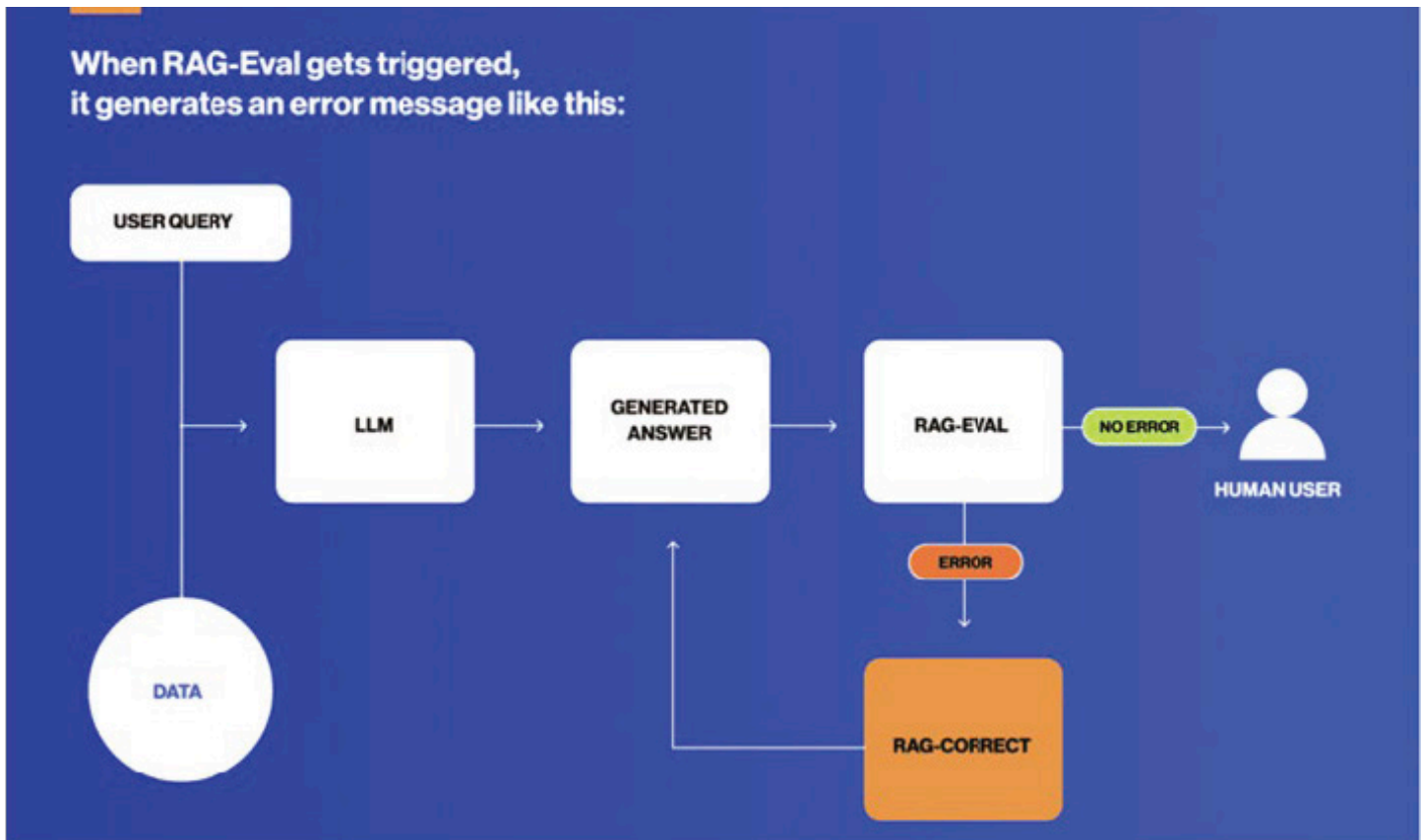
This is where large language models (LLMs) shine. An LLM can read and comprehend an entire document in seconds. By looking over your shoulder as you use a search system, an LLM can read both your query and the dozens of retrieved search results and generate an answer or summary on the spot.

This setup is called retrieval augmented generation (RAG). The search system can be boolean, semantic, or any system that retrieves relevant information for the task. In the context of search, the goal for the LM is to be your study buddy, reading ahead for you and providing a quick analysis.

RAG systems have become commonplace, but beware, buyer. For intelligence analysts in particular, there are two features that are absolutely required.

First, the text generated by a RAG system must cite its sources rigorously. Usually that means every single sentence has a citation to at least one of the source documents, and individual claims are all covered. This is a core part of the tradecraft. A system that just generates text based on vibes will not be trusted.

Second, it only takes a few mistakes to lose the trust of an intelligence analyst. RAG systems for intelligence use cases require an error-catching and error-correcting mechanism. We built such a system here at Primer called RAG-Verification.



With those caveats, RAG is an incredible search tool. It makes it possible to launch search queries quickly, reading the generated answer to understand the gist and then iterating on your query rapidly. What once required an analyst to scan through dozens of articles to find a relevant one, they now have the help of an LLM to summarize hundreds before settling on the few docs worth reading.

---

## Search as conversation: LLM-powered chat

Pros	Cons
The most natural interface for iterative search	Bringing such systems to air-gapped systems is a challenge
Opens the door to AI agents	Open source solutions lag far behind proprietary products

The next evolution of AI-powered search is already here. Our intelligence analyst customers tend to be savvy early adopters. They're all using ChatGPT and Claude in their private lives. They see the future and they can't wait to have this power at work.

It's coming. As you can imagine, there are extra hurdles to cross when the data lives in an air-gapped top secret system. But these systems are slowly arriving to analyst's day job.

In a certain sense, chat is just RAG with a different user interface. Under the hood, it still all begins with search. Relevant data must be retrieved for an analyst to have anything meaningful to chat about. And it is indeed RAG that powers that.

But chat systems do have a wonderful advantage: Conversation is a very natural mode of analytical exploration. It is the difference between getting help from a librarian to find relevant documents versus talking with a smart colleague at a whiteboard, working on your analysis together.

The key consideration for intelligence analysts as they start adopting this powerful search interface is this: How do you prevent complacency from allowing best practices and analytical tradecraft from dwindling away? Working directly with intelligence customers, we understand that challenge completely. These are tools that should make it easier to practice traditional tradecraft, not a set of shortcuts.

There are even greater capabilities, beyond search, that a chat interface can bring to the desk of our intelligence analyst. Empowering LLMs as tool-using "agents" is the future. Stay tuned for a blog post devoted to AI agents for intelligence analysis.

This concludes our overview of the new and ever-growing search tools that AI is bringing to intelligence analysis.

: Learn more: [www.primer.ai](http://www.primer.ai)