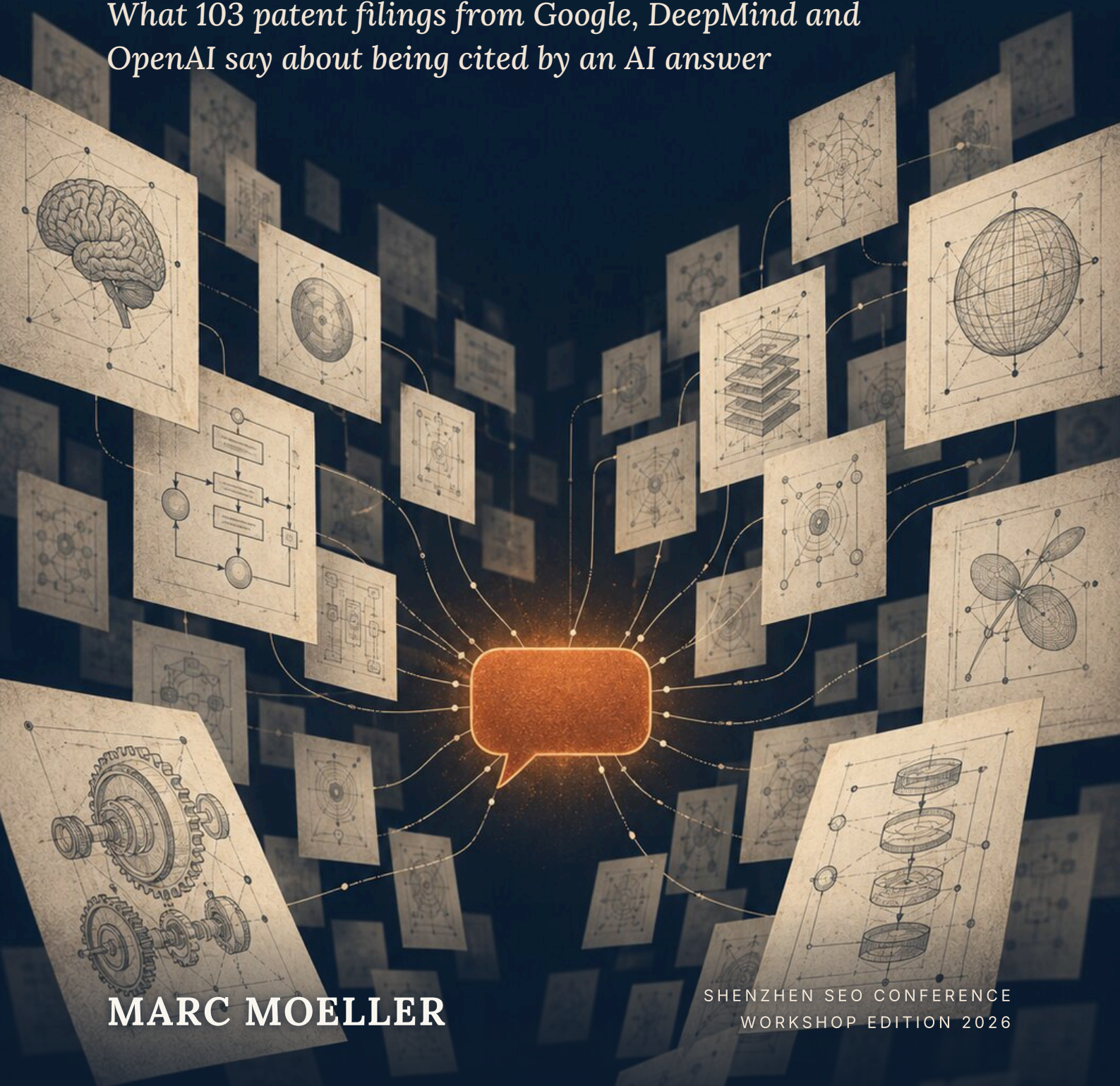


GENERATIVE ENGINE OPTIMIZATION

The GEO Patent Book

What 103 patent filings from Google, DeepMind and OpenAI say about being cited by an AI answer



MARC MOELLER

SHENZHEN SEO CONFERENCE
WORKSHOP EDITION 2026

ABOUT THE AUTHOR

Marc Moeller



Marc Moeller is a Perth-based SEO consultant, author and course creator, and the founder and SEO Director of Ecomexperts.

He wrote this book for the GEO workshop he taught at the Shenzhen SEO Conference 2026 and gave it to every attendee. This public edition is the same text, now free for anyone.

He is also the author of *SEO Algorithm Cracked* (2026).

marcmoeller.au

Free, open-source tools from the same work

Climby AI SEO

Search Console data, rank tracking and a GEO tracker for AI citations. Licence AGPL-3.0.

github.com/Marc-Moeller/Climby-AI-SEO

AI Visibility Tracker

Self-hosted tracker for brand mentions and citations across AI answer engines. Licence AGPL-3.0.

github.com/Open-Source-SEO-Tools/ai-visibility-tracker

All 103 patents as a pack, plus the workshop materials

workshop.seo1.us

The GEO Patent Book, public edition, October 2026. First given to attendees of the Shenzhen SEO Conference 2026 GEO workshop. Text licensed under Creative Commons Attribution 4.0 (CC BY 4.0): share and adapt with credit to Marc Moeller. A patent is a filing, not a shipped product; this book describes what the filings say, never what a search engine does.

WORKSHOP EDITION

The GEO Patent Book

What 103 patent filings say about being cited by an AI
answer

103 filings from Google, DeepMind and OpenAI, every one with a priority date of 2020 or later, read in full. A patent is a filing, not a shipped product, so this book says what the filings describe and never what a search engine does.

CONTENTS

1. How a page earns an answer citation

2. Search may use a different question

How earlier questions shape the next search
When a question becomes several searches
How conversation context changes the search
Which extra words enter the search
Which pages join the answer material
When an existing page can replace generation
Why clear titles help explain the subject
What to change on the site

3. Getting fetched at all

What does a search system fetch?
Does search retrieve pages or passages?
Can a public page enter the search?
How much content survives selection?
Does search read the latest page?
What to change on the site

4. Ordering the candidates

When is the order decided?
What determines a page's score?
Where do training examples come from?
How do shared concepts change passage order?
How does comparing passages decide their order?
Can a later sentence improve rank?
Can one query produce several ordered blocks?
What happens when a domain inflates a signal?
What to change on the site

5. Which sources fit into the answer

Which documents reach the model?
Does search position control entry?
What happens when the input overflows?
Which part of your page enters?
Does source order change credit?
Do private document rules apply to websites?
What else takes up space?
What to change on the site

6. How the sentence gets written

Which words can become a quote?
When does a summary show a citation?
What shapes the answer used for training?
Can an answer without source support win?
Which draft passes the quality checks?
What can beat the first draft?
What to change on the site

7. The checks that run after the answer exists

What can reach the reader?
Can paraphrases earn citations?
What happens to unchecked claims?
What wording can checkers verify?
When do checks stop?
What to change on the site

8. Which source gets the link

What earns a citation?
Which pages can supply a quote?
What does the reader see?
Which pages can still earn citations?
Can a later search find sources?
Does page ranking choose citations?
What must a page provide?
What to change on the site

9. How answers change beyond search

Whose history shapes the answer?
Which pages qualify as evidence?
When can stored answers change?
Which pages teach the model?
Does grouping results change their rankings?
What page structure helps extract facts?
Can publishers influence these training processes?
What to change on the site

10. Five things the filings do not support

Is matching the typed query enough?
Does retrieval compare one whole page?
Will a paraphrase earn a citation?

Do reported results establish live product behaviour?

Can later scoring rescue a missed page?

11. Your plan for better search visibility

Make your pages available

Make each passage answer a question

Make your evidence easy to cite

Check whether the changes worked

What 103 patent filings say about being cited by an AI answer.

Google, DeepMind, GDM Holding and OpenAI filed these patents. Every one carries a priority date of 2020 or later, and every one was read in full rather than by its title. A patent is a claimed method, not a shipped product, so this book says what a filing describes and never what a search engine does. Each subsection ends with the publication numbers behind it, so you can check any claim against the source.

The filings disagree on plenty: what gets stored, what gets retrieved, where the numeric cutoffs sit, and whether a citation is an embedding match or an exact quote. They agree on the stages a page has to survive before a reader sees it cited. That agreement is what this book is built on.

How a page earns an answer citation

A page must survive rewritten searches, retrieval, selection, drafting and citation checks before users see a sourced sentence. This guide draws on 103 patent filings from Google, DeepMind, GDM Holding and OpenAI, read in full. All have priority dates of 2020 or later; no judgments were based on titles alone. A filing claims a method; it does not establish a shipped Search, Gemini, Drive, or ChatGPT product. The briefs differ on stored and retrieved material, numeric cutoffs, and citation requirements. Some require embedding matches; others require constrained quotes. They agree on the stages a page must survive.

Write for the searches a system might issue, because the user's typed words are only the starting point. Three session-history filings describe merging earlier queries into one combined search. Two long-prompt applications describe facet subqueries that divide a lengthy request. Another chat application replaces pronouns with searches naming an entity and its attribute. A language-model application appends extra terms to original tokens. Two context-retaining chat applications generate synthetic queries to fetch additional documents. A page matching only the user's wording can therefore miss every search the system issues.

Check what material a system searches, because some pipelines never search websites. Three applications describe index lookup through neighbor search or a sequence model. Neighbor search finds nearby encoded representations; a sequence model processes sequences to produce its outputs, here identifiers. Three further filings use different retrieval units: pages, passages, sentences, token sets, or cloud files. Other applications search Drive-style files, a tagged custom corpus, or a precomputed review database. Those three pipelines never search the open web, so crawlability matters only for website corpora.

Your page must survive the first selection, because the model sees only some of the fetched candidates. A two-pass application and a granted filing describe an initial cut, followed by scoring only that selected set. Five Google filings retain the top N already-ranked documents for the generation prompt. Examples set N to 2 or 3. Four GDM filings take a predetermined number p of the highest-ranked HTML documents. The two cited grants never split documents below that cutoff into segments, preventing those documents from contributing snippets. One cited Google grant and one cited GDM grant select snippets through query term overlap or embedding distance. Embedding distance measures separation between numerical representations of text; the prompt receives selected snippets rather than whole pages. Another cited Google grant summarizes material before packing it if the context window still overflows.

Make passages self-contained and support statements clearly, because drafting and citation checks can each determine what receives attribution. The writing step generates text under syntax rules, templates, or quote constraints. A selection step scores drafts; Canadian, European and Australian filings restrict output to exact substrings. These outputs can only copy an exact substring of a fragment already included in the prompt. Google's summary filings write first, then attach citations when an encoder finds a summary portion close to document content. A second pipeline inspects drafts; two string-match filings normalize text windows. They compare alphanumeric copies, hashes, or edit distance. Edit distance counts changes needed to align text, and these string-match filings produce no match for paraphrases. Two support-scoring applications can flag a claim, rewrite it, or leave it unannotated. Query-time links and quotes

require membership in the retrieved candidate set, except in one Google summary filing. The cited Google summary grant and international application describe finding a citable document by searching with the summary portion.

Four findings recur most often and show where to focus work on your content:

- Systems rewrite, expand, split, or replace machine queries, so matching the user's exact wording is not the retrieval key.
- The first fetch gates access: later rankers, prompt packers and citation checkers never see pages that miss it.
- Make passages or sentences self-contained: systems embed, pack, quote and check these units, rather than whole pages.
- Citations depend on post-draft closeness through embedding distance or exact copying; only retrieved or later verification-search documents qualify.

```
[ typed query ] --rewrite--> [ machine query ]
      |
      neighbor search
      v
[ candidates ] --re-rank--> [ top-N ]
      |
      pack snippets
      v
[ prompt ] --decode--> [ draft ]
      |
      verify
      v
[ cite or withhold ]
```

Filings: US11762848B2, US12242472B2, US20210406260A1, US20250117381A1, WO2025080561A1, US20250173519A1, WO2024229375A1, US20240289407A1, WO2024182453A1, US20250363101A1, US20250165469A1, WO2023154385A1, US20250217373A1, WO2024215344A1, US11003865B1, US20250103826A1, US20240362093A1, US20250252121A1, WO2024226614A2, US11989649B2, US11769017B1, US12118325B2, US11886828B1, US12008473B2, US12333436B2, CA3245108A1, EP4466630A1, AU2023236937B2, WO2024145537A1, US11900068B1, US20260064780A1, US12596889B2, US20250156456A1, WO2024073087A1, WO2023175089A1

Search may use a different question

The filings describe searching rewritten queries, which can differ substantially from the words the user originally typed. They use session history, add model-generated terms, combine earlier requirements, split questions by facet, or create synthetic queries. A page matching only the user's words can therefore miss every search that the described system actually issues.

How earlier questions shape the next search

The family describes searching a combined query that includes earlier questions, rather than the latest message alone.

Three filings, two granted and one published application, describe the same family, with a priority date of 2020-06-25. A **semantic similarity model** checks whether current and earlier queries follow the same inquiry. Alternatively, the system compares **embedding distance** (distance between numerical representations of meaning) against a threshold to decide whether the queries belong together. The system checks previous queries from newest to oldest, favouring queries that occur close in sequence or time. More intervening queries lower that likelihood, while phrases such as “Actually, I meant” signal an extension of the previous query.

When queries belong together, their **parameters** accumulate into a growing search. The first granted filing illustrates this accumulation with an example combining the terms “Bluetooth”, “true wireless”, and “earbuds”. Mandatory parameters can control which results qualify; weighted parameters help the ranking engine. The session reuses this growing query until an interruption; striking through tokens or removing results can add exclusions. All three filings also describe inferring exclusion topics unique to a removed result, but none gives a numeric similarity threshold. The second grant and the application claim silent combining or a mode that maintains state across queries.

Filings: US11762848B2, US12242472B2, US20210406260A1

When a question becomes several searches

The filings require checks on query length, rarity, result quality, and server load before splitting a question.

Two filings from Google LLC share the priority date 2023-10-10 and describe the same sequence for splitting queries. A selective subquery engine considers token count, whole-query result quality, server load, and submission frequency. Rare queries favour splitting, helpful whole-query results bypass it, and failed checks leave the original query unchanged.

When checks pass, a **generative model** proposes subqueries. Its prompt supplies a few examples or instructions such as “break the following into simple pertinent questions”. A cheaper **encoder** then filters candidate subqueries for relatedness and diversity. It checks relatedness through embedding distance from the original input, and diversity through distance from already-selected subqueries. Search receives only surviving subqueries, each retrieving its top result, top N results, or results meeting a quality threshold. Neither filing specifies numeric cutoffs for these checks.

The international filing describes one response mode using a **large language model (LLM)** to summarise results. Another mode displays each selected subquery with its corresponding results; this filing also describes omitting extraneous clauses from searches. The US application states that whole-query results “only address each of the facets at a superficial level”.

Filings: US20250117381A1, WO2025080561A1

How conversation context changes the search

The filing describes matching a rewrite containing conversation context, then a group of queries sharing the same intent.

A generative language model reads the input and multiple turns of session history to create a self-contained query. It replaces context-dependent wording, such as pronouns, with proper-noun referents. An embedding captures the rewrite or its intent and routes it into a learned **intent graph** (a network grouping queries by purpose). Groups within that graph identify stored results and tagged content, while known queries can reuse **cached results**. Tagged items have conditions for appearing uniformly across every query mapped to the same group of queries.

The example rewrites “how much does it cost?” as “Brand X Model 17 pro max price”, making the subject explicit. **Graph nodes** can carry attributes such as “consumer facing”, “late-stage funnel”, or “commercial facing”. Connections between topics can support suggestions from price to warranty research. Intent models may group queries whose clicks have similar distributions, but this published application gives neither thresholds nor ranking weights. It allows searching rewritten or grouped queries, reusing cached results, or selecting tagged items, leaving the retrieval path open.

Filings: US20250173519A1

Which extra words enter the search

The filing describes searching an expanded query that combines the user’s original words with a language model’s output.

During a query, the system prompts a language model to produce extra terms for the search it issues. Output can be a keyword list, a passage, or reasoning plus an answer. The system combines these terms with the original query and repeats the original words to increase their weight. The description uses five

repetitions as an example, while different prompt choices change the terms added during expansion.

Prompt choices include **chain of thought**, **zero-shot**, and **few-shot**. Keyword lists and passages are also prompt choices, and independent claims differ on which prompt they require.

The system can optionally use **pseudo-relevance feedback**, inserting first-pass hits as Context. Those hits let retrieved documents shape expansion and can come from Vector Space Model, BM25, or language-model retrieval. Expansion aims to find documents sharing no words with the original query, extending searches beyond exact wording. The description prefers keyword-list prompts because passage prompts introduce irrelevant terms; the claimed output is expanded-query results, rather than a generated answer.

Filings: WO2024229375A1

Which pages join the answer material

Pages answering the original query and machine-written synthetic queries enter the document set described in these filings.

Two filings, both with priority 2023-02-28, describe maintaining **user state** (stored context carried across conversation turns). That state draws on prior queries and extracts from earlier result pages and documents throughout the conversation. Calendar or correspondence, position, synthetic-query results, and actions after a query also enrich the available user state. A first generative model creates synthetic queries from the query and context before a **classifier** routes subsequent work to zero, one, or several downstream models. Named classes include SRP summarization, next-step query, clarification, and do not interfere.

Documents qualify through query-dependent, query-independent, and user-dependent measures. Query-dependent measures include ranking position, selection rate, locality, and language; query-independent measures include popularity, freshness, and trustworthiness. Trustworthiness considers the author, domain, and inbound links. Further checks control whether documents from related, recent, or implied queries join the set of qualifying documents. These checks consider related-query correlation, recent-query overlap, and low quality or diversity among the original search results. An answer portion receives a source link only when its embedding distance from document content satisfies a threshold. Neither filing gives numeric thresholds for these decisions, so the described conditions do not establish numerical targets.

Filings: US20240289407A1, WO2024182453A1

When an existing page can replace generation

The filing describes switching off generation for part of a query when retrieved content provides a strong match.

The system receives and parses a query, then decides how to handle each portion through retrieval, generation, or both. Explicit phrases such as “generate a”, “show me”, and “make a for me” can trigger generation and bypass retrieval. Words associated with a generative model can also signal generation without an explicit generate verb, as in “play relaxing jazz”. System load can favour retrieval, while user preferences can shift routing through an **affinity score** (a measure of user preference fit).

The system selects retrieved items using query-item similarity, measured through Euclidean distance, cosine similarity, or dot product. The description gives 95% as an example threshold at which a strong retrieved match can bypass generation. Its qualifiers, “such as” and “e.g.”, make this threshold an example rather than a claimed constant for implementation. Retrieved content’s presence can switch off generative models for that portion, while ranking considers quality, consistency, and affinity from user preference. Quality covers query-result similarity, clicks on similar past queries, and keyword number and quality; consistency means routine selection. An optional embodiment allows the routing to proceed sequentially or in parallel.

Filings: WO2025165363A1

Why clear titles help explain the subject

Titles and URLs should clarify the subject because this filing uses them for training context, rather than live ranking features.

The filing describes an offline training recipe using retrieved documents. A **teacher model** (a model supervising another model) labels queries by reading retrieved documents’ URLs and titles joined together. Two **distillation** stages (training that transfers behaviour between models) pass that behaviour to a smaller model for later use. At **inference**, the smaller model sees only bare query text, without those documents. For the example “ua 1234”, retrieved documents help the teacher identify a United Airlines flight as the subject. Technical terminology is an explicit query category, and only pages retrieved for that query contribute to its inferred labels. The filing leaves the classifier’s use at query time unspecified; its background only says predicted intent can change search-engine behaviour.

Filings: US20240232637A9

What to change on the site

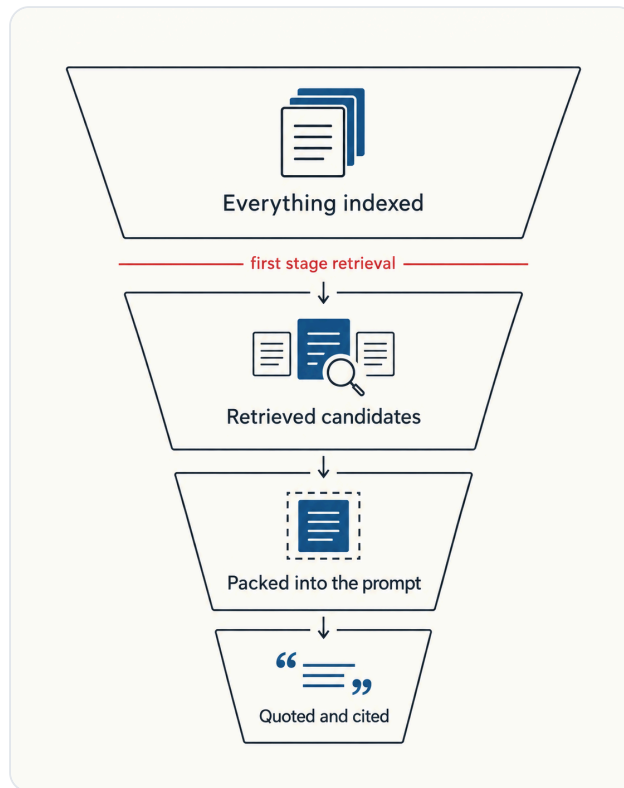
Make each page answer a clear question while naming the details that rewritten and expanded searches can contain.

1. Publish a dedicated page that fully answers each facet you sell or explain. Title it as a clear question about one problem. Two query-splitting filings search LLM-generated “simple pertinent questions” rather than the user’s cluttered, multi-part wording. They state that whole-query results cover each facet only superficially. (well supported)

2. Name entities, attributes, and complete specification combinations explicitly in page copy and titles. The conversation-rewriting application replaces pronouns with entity-plus-attribute queries. The three filings in the query-combining family accumulate mandatory parameters such as “Bluetooth”, “true wireless”, and “earbuds”. The query-expansion filing adds model-generated entities and synonyms to the query sent to the index. (well supported)
3. Place key terms in titles, in bold, high on pages, and near related query terms. The three query-combining filings extract suggested parameters from resources using frequency, location, and proximity to current parameters. They also use font, size, and alignment, explicitly naming title position and bolding. (well supported)
4. Keep each page focused on one entity or problem. The three query-combining filings describe inferring topics unique to a removed result as exclusions for later combined queries. (well supported)
5. Target rewritten, expanded, synthetic, and grouped queries alongside the user’s original wording. The international and US filings on user state add synthetic-query documents to the set used for summaries. The intent-graph application ties results to groups of equivalent queries. The expansion filing explicitly seeks documents sharing no words with the original query. (well supported)
6. Write self-contained passages ready for snippets, with wording close in embedding space to the claims you want cited. The US and international user-state filings require answer-to-document embedding distance to satisfy a threshold before adding source links. One response path in the query-splitting filings summarises retrieved results, so only surviving content reaches the user. (well supported)
7. Give titles and URL paths plain, human-readable subject names, especially for part numbers, model codes, and cryptic technical queries. Retrieved titles and URLs supply teacher-model annotations for inferring intent labels in the distillation filing. (single filing)
8. Host a canonical original with strong keywords and appealing search listings. Place it in the public collection retrieval models compare against. The filing combining retrieval and generation can bypass generation for a portion when retrieved content exists and similarity is strong. Its ranking uses keyword quality, clicks on similar past queries, and routine selection. (single filing)

Filings: US20250117381A1, WO2025080561A1, US20250173519A1, US11762848B2, US12242472B2, US20210406260A1, WO2024229375A1, WO2024182453A1, US20240289407A1, US20240232637A9, WO2025165363A1

Getting fetched at all



Retrieval is the hard gate. Nothing downstream can rescue a page that misses it.

What does a search system fetch?

The fetched unit depends on what each filing encodes or binds, so structure content for that unit.

One filing describes a **fixed dimensional encoding** (FDE), a fixed-size numerical representation per item, for first-stage retrieval. Retrieval uses **k-nearest-neighbors**, the k closest matches, to select items using those precomputed numerical representations. Each item contains a set of embeddings P stored as an FDE. This approximates **Chamfer/MaxSim**, a score based on closest matches, rather than cosine similarity of a pooled document embedding. **Cosine similarity** compares vector directions; pooling combines information into one representation. The original embedding set loads only for optional Chamfer re-ranking of the top- k subset.

Other filings store different objects. **Dual encoders** encode queries and content separately. One application compares phrase, sentence, passage, or document embeddings with a query encoding created when the request arrives. Two granted filings build one document encoding from sentence blocks with limited capacity. They use cosine similarity, and one pooling option combines information using only the first contextual block. Another granted filing ranks a document vector through **Maximum Inner Product Search** (MIPS), which finds the highest inner products. An inner product multiplies corresponding values and sums them. One application scores adapted query and corpus embeddings by

cosine; another converts a **multivariate distribution** to a dot product for approximate nearest neighbor search. A dot product is an inner product here; approximate search finds close matches without guaranteeing the closest.

Two **Differentiable Search Index** (DSI) filings replace the vector table with a model whose parameters form the index. Their sequence-to-sequence model emits resource identifiers without an inverted index, embedding table, or MIPS search. The international application describes direct indexing, which binds a resource's leading span to its docid. Set indexing removes stopwords and duplicate tokens. Inverted indexing maps chunks from anywhere in the resource to its docid.

These filings describe different stored objects, not shipped products. First-stage fetch searches neighbors among precomputed vectors or uses a model to emit identifiers.

Filings: US20250363101A1, WO2024215344A1, US11694034B2, US12210837B2, US11003865B1, US20240378224A1, US20240354557A1, US20250165469A1, WO2023154385A1

Does search retrieve pages or passages?

Search can retrieve pages or passages, and the filings define no single unit that every system must use.

The contextualized token retriever embeds documents beforehand as context-sensitive token vectors. A document becomes a candidate when any one token vector enters a query token's most similar set. The predefined set size k caps document token vectors reaching scoring; examples give 10, 100, and 1000. This also caps candidates at up to n times k . The relevance score averages the maximum matches for each query token. Unmatched query tokens receive a substitute value, with examples of zero, 0.2, or 0.5. The document set can contain individual text segments.

The multivariate-distribution application also allows a text segment to count as a content item. Three dual-encoder filings encode a phrase, sentence, passage, or document. A granted filing describes **retrieval-augmented pre-training**. It treats a document as a whole page, selected sections such as title, body, or tables, or one paragraph or sentence. Three custom-corpus filings use one precomputed embedding per document. Four cloud-document filings retrieve a sentence, paragraph, or section.

Filings: US20250217373A1, US20240354557A1, WO2024215344A1, WO2024064249A1, US20260064743A1, US11003865B1, US20240362093A1, WO2024226082A1, EP4662565A1, US20250103826A1, US20250103867A1, WO2025065013A1, US20250328578A1

Can a public page enter the search?

Making a URL public does not, by itself, put that page into the cloud-file searches these filings describe.

Four related filings retrieve from a cloud-based content management platform: Drive-style documents, spreadsheets, emails, and transcripts, rather than ranked web pages. The first and fourth filings use permission metadata to remove files the user cannot open before generation. Last-modified and last-

opened timestamps help rank or exclude files. The fourth and third filings support this time filtering; one description uses last week as an example range. At indexing, the first filing embeds **synthetic questions** (generated questions about the content) for each portion and matches those embeddings to the user prompt.

A separate trio conditions a **large language model** (LLM) on a private or third-party custom corpus. An external application's **API** provides access. Corpus tags and application metadata control access, and selection stays within that corpus.

Two sequence-generation retrieval filings use a reference set that can change at inference and describe no open-web crawl. Another application retrieves tools by matching queries to documentation plus synthetic queries, rather than pages. A browser application searches a vector store of pages the user has already loaded. Two entity-summarization filings retrieve from a precomputed public-review database instead of the live page; unselected sources never enter.

A 2020 filing provides the exception resembling open-web retrieval. Its knowledge corpus “comprises one or more websites remote from the processing system”, and complete HTML pages count as documents. Crawlability is necessary only for pipelines that name websites as their corpus.

Filings: US20250103826A1, US20250103867A1, WO2025065013A1, US20250328578A1, US20240362093A1, WO2024226082A1, EP4662565A1, EP4388458A1, US12536439B2, US20250265285A1, US20250217428A1, US20250252121A1, WO2025170840A1, US11003865B1

How much content survives selection?

The filings describe small surviving sets, sometimes empty, but name no production k for a public-web generative answer.

Three related custom-corpus filings retain a predefined N, described as an integer from 3 to 20, such as 10. Selection follows an unspecified distance threshold using L1, L2, correlation, or cosine similarity. L1 sums absolute coordinate differences; L2 measures straight-line distance; correlation measures how values vary together.

A granted filing describes REALM-style retrieval, keeping a top-k set ranked by inner product. It uses **softmax**, which converts scores into weights. It appends a **null document** whose success marks retrieval as unhelpful. The FDE application keeps a top-k subset for Chamfer re-ranking and can drop items below an unspecified threshold. The token-retrieval application can produce up to n times k' candidates. A relevance threshold, rank, logistic function, or softmax selects the returns.

Two PROMPTAGATOR filings describe query-generated retriever training and use the retriever's top 200 as the inference distribution. This distribution supplies candidates during model execution to a listwise reranker. A cross-example softmax filing calibrates distances; an absolute threshold can withhold a nearest neighbor and request clarification. The first two custom-corpus filings also use a response confidence score that can withhold the answer.

Another application selects portions for the generative prompt after conventional ranking, while balancing diversity. It skips generation if too few pass a portion relevance threshold. It compares embedding similarity with a diversity threshold to accept or reject candidates against already-selected portions. Selecting one portion can exclude other portions from the same resource or domain. Claims 5 and 16 limit portions to less than 500 characters.

One **self-reflection** filing ranks chunks of approximately 300 words or tokens and keeps the k highest for generation. The description gives five for multi-step reasoning and one for a single piece of data. The model skips remaining context if it predicts that this subset can satisfy the query. Two sequence-generation filings retrieve k nearest reference segments per output chunk, with examples of 2, 5, or 10. They embed only the initial sub-segment as the key.

These filings converge on top-k or top-N after a distance threshold. Diversity or self-reflection can further cut that set, sometimes leaving it empty.

Filings: US20240362093A1, WO2024226082A1, EP4662565A1, US11003865B1, US20250363101A1, US20250217373A1, WO2024064249A1, US20260064743A1, US20230111978A1, US20260072965A1, WO2026020082A1, EP4388458A1, US12536439B2

Does search read the latest page?

Stored summaries replace live pages in two related filings, so freshness depends on updating the material search retrieves.

Both retrieve precomputed public-review summaries from an external entity-summarization database. A recommendation-class label or keywords such as “can you recommend” / “what is the best” control access. The system embeds the query, optionally with chat history, and compares it with stored summarized-description embeddings. It uses cosine similarity or **Euclidean distance** with a distance threshold controlling selection. The US application additionally stores feature-level summaries as separately retrievable entries. Both filings describe detected updates to published reviews triggering real-time regeneration of database entries.

Three custom-corpus filings recompute document embeddings when content is added or updated, or through periodic checks, without retraining the LLM. A dual-encoder application lets indexes grow through bulk inference on newly discovered or updated documents without retraining the encoders. A cloud application filters by last-modified recency. The DSI pair differs because its corpus lives in model parameters. Most of these filings therefore depend on index maintenance to keep fetched content fresh; the DSI pair depends on parameter updates.

Filings: US20250252121A1, WO2025170840A1, US20240362093A1, WO2024226082A1, EP4662565A1, WO2024215344A1, US20250328578A1, US20250165469A1, WO2023154385A1

What to change on the site

1. Write each retrievable claim as a self-contained sentence, paragraph, or short section. Several filings retrieve a portion, segment, or passage rather than a whole page. A unit that answers one question on its own gets embedded and cited. (well supported)
2. Put distinctive topical content at the start of every page and passage. Direct indexing binds leading tokens to a docid. First-block pooling can supply the whole-document encoding. Sequence-generation retrieval embeds only the initial sub-segment, and diversity portions have a limit of less than 500 characters. (well supported)
3. Keep each indexable unit on one topic. Split distinct aspects into passages whose embeddings will sit far apart. Single-vector encodings blur multi-topic pages. Diversity selection prefers embedding distance from the already-chosen seed. It can therefore skip a restated angle even when that angle is highly relevant. (well supported)
4. Keep named entities, rare terms, and exact phrases as distinct tokens, beyond a paraphrased gist. Chamfer/MaxSim scores each query embedding against its nearest document embedding. The FDE inner product approximates that local match. (single filing)
5. Pack the facts that answer a likely query into one span of approximately 300 words or tokens. The self-reflection filing's first generation attempt reads only k chunks of that size. If the model predicts that subset will suffice, it never processes the rest of the page. (single filing)
6. Publish into the corpus the pipeline actually searches, and keep that copy current. Custom-corpus, cloud-permission, review-source, and reference-set filings do not fetch the open web at query time. Embeddings or summaries regenerate when content is added or updated. Stale documents remain in the fetched set until that maintenance runs. (well supported)
7. State the question a page answers in titles and headings, using the language users type. Adapter cosine scoring, index-time synthetic-question embeddings, and document-to-query retriever training all match query-shaped text to corpus text. (well supported)

Filings: US20250217373A1, US20250103826A1, US11003865B1, WO2024215344A1, US20250165469A1, WO2023154385A1, US11694034B2, US12536439B2, US20260072965A1, US20240354557A1, US20240362093A1, US20250363101A1, WO2026020082A1, US20250252121A1, WO2025170840A1, US20250328578A1, US20240378224A1, WO2024064249A1, US20260064743A1

Ordering the candidates

These filings describe scoring between **retrieval** and presentation. The upstream **retriever** has already narrowed the candidates, but their order can change. Some filings describe more than one scoring pass before a model sees the list.

When is the order decided?

Order is decided after a retriever narrows the full collection to a set of candidates. The LITE filing describes selecting a predetermined subset, then **re-ranking** that subset. Its second pass never scores a page that misses the first cut.

The **pairwise neural ranker** filings make the same distinction between finding candidates and ordering them. A **candidate generation model** filters items by predicted relevance before a ranking network scores those candidates. The graph reranker family sits between a **dense passage retriever** (system finding text through numerical representations) and answer-generating reader models. **Extractive-summary re-ranking** reorders already-retrieved top resources for complex queries. **Pairwise ranking prompting** compares candidates after retrieval and before presentation.

Filings: WO2024226614A2, US11989649B2, US20210406680A1, US12511325B2, US20250335486A1, WO2025226655A1, US20250139105A1, US20250124067A1

What determines a page's score?

The LITE filing describes scoring a grid of similarities between **tokens**. At **index time**, a second **transformer** (neural model processing relationships between tokens) converts document tokens into stored **embeddings** (numerical representations of text units), D . Those tokens use Word-Piece or SentencePiece, methods for dividing text into units. At query time, a first transformer converts query tokens into Q , their numerical representations for scoring.

The query-document similarity grid, $S = Q^T D$, feeds a thin **MLP** (small multilayer neural network), producing one relevance value per document. Flattened LITE feeds $\text{vec}(S)$, the flattened grid, into the MLP, while Separable LITE collapses rows, then columns. **Average pooling** and **linear projection** can shrink stored D . The scorer therefore uses compressed embeddings, which may represent less than the full token sequence. It does not use one **pooled page vector**.

Another filing derives an **information retrieval score**, abbreviated IR, from publisher-configurable page characteristics. These characteristics include search-term count, location, font, size, and color, and the score later feeds an aggregate ranking function f . The two filings describe different scorers: LITE learns from token interactions, while the exploit-detection filing uses individual IR signal inputs. The exploit-detection filing does not name the signals under test.

Where do training examples come from?

The pairwise neural ranker filings derive **labels** from displayed order and user actions. Both belong to the same family, with priority 2020-06-26, and their training pair generator reads presentation logs. An item the user views, selects, or installs becomes a positive example for the training process. A skipped item becomes a negative example only when it appeared above that positive in the displayed order. A **position bias model** (model adjusting for display-position effects) gives less weight to positives near the top and more to positives lower down. Named inputs include time spent, usage beyond a threshold, and a quality measure for the item. Scoring at **inference** stays unchanged: the network scores candidates, which are sorted by score. The worked examples concern an application store, and the description extends them to search results. However, the claims provide no evidence for that extension.

A later filing trains **ranking losses** on pairs from two different training lists. A **correlation score** controls which pairs qualify, based on the relationship between their queries. Alternative controls include publisher identity, genre, user features, item features, or list position in the training data. This filing concerns training methods, and claims 1 to 20 stand canceled. It says nothing about selection or citation when serving results.

Two related filings train a **semantic matching model** using document activity logs rather than clicks. They concern cloud or private storage search, not open web search, and public-web publishers cannot control those logs.

How do shared concepts change passage order?

The filings describe how passing information between linked passages in a **document graph** changes rank. The first filing is granted; the other two are applications in the same family, with priority 2024-04-26. Agreement across these three filings provides this set's strongest evidence for the mechanism they describe.

In the described **open-domain** setting, dense passage retrieval, abbreviated DPR, returns passages of a predetermined length. The system builds an **AMR** (abstract meaning representation linking concepts) graph for each question-passage pair, then builds a document graph. Passages become **nodes** in that document graph. Two passages get an **undirected edge** when their AMR graphs share concept nodes. The system then removes isolated nodes from the document graph.

This is the closest these briefs get to **de-duplication**, but it works against classic diversity. The system drops unique wording without conceptual overlap rather than promoting it to make results more varied. None of the supplied filings describe diversity or **near-duplicate suppression** when serving results.

Node features come from single-source shortest paths, the shortest routes starting at the question node. A **GNN** (graph neural network combining neighboring representations) lets a weakly linked positive inherit evidence from a neighbor. The reranking score is the **dot product** of **query encoding** y and the final node representation. Independent claim 1 in the applications requires neither AMR, GNNs, nor DPR. Natural Questions and TriviaQA experiments measure **ODQA** reranker performance, but do not establish a search ranking change. The text makes no connection to Google Search or AI Overviews.

Filings: US12511325B2, US20250335486A1, WO2025226655A1

How does comparing passages decide their order?

The filing combines head-to-head preferences into a list instead of using numeric comparisons. It prompts a **generative sequence processing model** with a query and two candidate texts. A second prompt swaps the texts to cancel position bias, and consistent answers produce a local winner. Conflicting or irrelevant outputs force a tie, with a clear preference earning one point and uncertainty half a point.

The filing specifies three ways to schedule comparisons:

- **All-pairs aggregation** compares every pair, with ties falling back to the initial ranking.
- A **sliding window** starts from that ranking and moves swaps upward over K passes.
- **Heapsort** uses the **LLM** preference as its **comparator**.

The description's sliding-window example uses $N=100$ and $K=10$, but those figures are not claim limitations. This mechanism re-ranks existing candidates rather than selecting citations, and differs from the two related neural training filings. Those filings describe **offline training** using pairs, rather than prompting to order candidates.

Filings: US20250124067A1, US11989649B2, US20210406680A1

Can a later sentence improve rank?

The filing describes summary relevance scores that can lift a resource above one with a higher initial score. The process runs only for queries classified as complex that a stored **entity attribute** cannot answer. A **portion identifier** selects relevant parts before a sentence scorer evaluates every sentence against the query. Here, a sentence includes table rows and list items, so those formats also enter the scoring process. A **concatenator** joins the highest-scoring sentences into an extractive summary, keeping their original document order. It inserts an ellipsis between sentences from different portions or far apart in the source resource. The system scores that summary against the query and uses the score to re-rank the top resources. It can also display the summary on the page as a **snippet**. The filing gives no numeric sentence or portion thresholds, and the claims leave score combination open. They do not specify how the summary score combines with other ranking signals.

Filings: US20250139105A1

Can one query produce several ordered blocks?

The filing orders content blocks from separate **subtask** searches by interaction score. When a query relates to a specific task, a **multi-task model** splits it into subtasks. Each subtask gets a separate search, and its content item appears above another subtask's item when its user-interaction score is higher. The score combines interactions across users, and one interaction can raise a score and reverse later ordering. Scrolling can insert a third content item from the subtask the user engaged with during that visit. That concentrates the live page on that subtask rather than adding variety to the displayed content. Queries without an associated task fall back to a large language model. Before a query arrives, the system maps product pages **crawled** from a client account to tasks and subtasks. The only disclosed ordering rule strictly compares two interaction scores: higher goes above lower in the displayed results. This published application establishes neither a patent grant nor a launched feature.

Filings: US20250209127A1

What happens when a domain inflates a signal?

The filing describes resetting an exploited signal's IR score to its expected value before determining presentation order. This happens after retrieval and individual signal scoring, but before the ranking function determines the presentation order. The system aggregates IR scores and fits a **linear model**, $y=a*x+b$, with a non-negative coefficient. It compares a domain-level **residual** (difference from the model's prediction) against a threshold to identify an exploit. It excludes **navigational queries** from residual aggregation before checking for the exploit. When the residual indicates an exploit, the system assigns the exploited signal its expected value for ranking. The domain loses the outlier boost, but cannot rank below that expected value either, while other signals still feed function f . The claims give no numeric residual threshold and leave the signals under test unnamed in the filing. This clips one signal back to trend rather than banning or deleting the domain from results. It provides no rule for citations in generated answers.

Filings: US20260023790A1

What to change on the site

1. Make each passage a self-contained answer that names entities and their relations in plain sentences. Use the topic's canonical names so related passages share concepts. Graph rerankers parse question-passage pairs into concept nodes and drop isolated nodes. They pass score along edges that exist only where AMR graphs share nodes. (well supported)
2. Put matching language into the tokenized body as distinct terms, extending beyond a topical summary. LITE ranks a grid of token-level similarities between query embeddings and stored document embeddings. (single filing)

3. Win the interaction when your listing appears, including in lower positions. Pairwise training takes positives from engaged items and negatives from skipped items above them. Its position bias model gives extra weight to positives from lower positions. (well supported)
4. Answer complex questions with self-contained sentences, list items, and table rows throughout the page. Extractive-summary re-ranking scores sentences joined from multiple portions. It can outrank a page whose best single passage earned a stronger first-pass score. (single filing)
5. Stop repeating the same inflated page signal across many URLs on one domain. Exploit detection aggregates per-resource residuals at domain level. It clips the exploited information retrieval score to its expected value before ranking function f runs. (single filing)
6. Protect first-stage retrieval: buried or unretrieved URLs have limited or no room to move in later passes. Candidate generation filters the set the ranker scores. LITE's second pass never sees pages cut by the first pass. Sliding-window pairwise prompting starts from the initial ranking and limits the number of upward-swapping passes. (well supported)
7. Keep product and use-case pages distinct and use industry terminology when targeting task-shaped queries. The task-based filing maps crawled product pages to subtasks. It orders blocks by aggregated user interaction rather than on-page markup. (single filing)

Filings: US12511325B2, US20250335486A1, WO2025226655A1, WO2024226614A2, US11989649B2, US20210406680A1, US20250139105A1, US20260023790A1, US20250124067A1, US20250209127A1

Which sources fit into the answer

Which documents reach the model?

Only selected documents from already ranked results reach the **generation prompt**. Five filings in the same Google family describe an **SRD selection engine**. It keeps the top N documents that answer the query, illustrating N as 2, 3, or another quantity. Documents below that cutoff never reach the model, even when their pages contain useful answers to the query.

Selection uses three groups of measures:

- Query-dependent measures include positional ranking, per-query selection rate, locality, and language.
- Query-independent measures include trustworthiness from the author, domain, and inbound links, plus popularity and freshness.
- User-dependent measures include profile attributes, recent queries, and recent non-query interactions.

These measures mean the same query can produce different prompt sets depending on locale, language, and submitter.

Optional documents answer related, recent, or implied queries, but enter only when direct results lack quality or diversity. A correlation or overlap magnitude must also meet a threshold, although the filings never quantify that threshold. The family's 2025 application places this selection in the description and in "some implementations," outside claims 1 through 20. Treat that application as weaker evidence than the granted family members when assessing how firmly selection is supported. The filings leave N, the weights assigned to these measures, and the method for combining them unspecified.

Filings: US11769017B1, US12118325B2, US11886828B1, US20240220735A1, US20250005303A1

Does search position control entry?

Yes, four Google DeepMind (GDM) family filings use external search rankings to exclude documents before scoring their text. The system obtains a predetermined number **p** of the highest-ranked documents as **HTML**. Everything below that cutoff misses segmentation, scoring, and **conditioning**. The filings treat the search engine as a black box and leave its ranking method entirely unexplained.

This process differs from the Google summary family, although both begin after **retrieval**. The system extracts clean text from each document, then splits it into fixed-size segments containing a predetermined sentence count. Each segment, or segments joined across documents, becomes a **conditioning input**.

The later granted filing and both applications describe an optional filter that represents the question and inputs as numerical vectors. It can use **tf-idf**, **averaged word2vec**, **GloVe**, or a **BERT-loss embedding network** to produce the representations. The filter scores **cosine similarity** or **Euclidean similarity** and keeps the n highest-ranked inputs, where n is at least 1.

Documents face the first cut (p), and text spans face the second (n), before the system generates answer data. Implementers choose p , n , segment size, and any similarity threshold; the filings supply no fixed values for these choices. Unlike the Google summary family, these filings describe no citation or link back to the source material.

Filings: US12008473B2, US12333436B2, US20230244934A1, US20240281659A1

What happens when the input overflows?

One granted Google filing describes summarizing document content before adding it to the prompt when memory limits require it. This is the set's only explicit budget rule, and it gives no count of **tokens**. It also gives no character cap or **truncation offset**.

The Google summary family already extracts part of each document, while the GDM family keeps only n segments. Together, they describe three successive reductions: fewer documents, shorter spans, and preliminary summaries if the window remains too small. The filings specify none of these alternatives: dropping a page's tail, keeping the first k tokens, or sliding a window.

Filings: US11886828B1

Which part of your page enters?

The selected span resembles the query, so the whole page need not enter the prompt for answer generation. Across the Google summary family, the **LLM input engine** selects snippets through query-term overlap. It can also select them using a word embedding within a threshold distance of the query embedding.

One variant uses a first **LLM** to select passages from the documents. Eligible content also includes image captions, recognized image text, detected object descriptors, and video transcriptions for the prompt.

The GDM family's later grant and both applications describe scoring fixed-size sentence segments independently, making segment boundaries consequential. A fact split across a segment boundary therefore competes at a disadvantage on both sides of that boundary. The system can join subsets from multiple documents, placing your span beside a competitor's in the same conditioning input.

The filings leave the overlap threshold, embedding distance, and sentence count unspecified, despite describing these methods for selecting text. In the Google family's first two cited grants, "verifiable" means comparing encoder embeddings of summary and document portions. This comparison is not a fact check, and happens after generation, so it cannot decide what enters the prompt.

Filings: US11769017B1, US12118325B2, US11886828B1, US20240220735A1, US20250005303A1, US12333436B2, US20230244934A1, US20240281659A1

Does source order change credit?

The Google summary filings describe source labels inside the prompt, but leave the order of packed documents undisclosed. **Source identifier tokens** let the model identify which document supports each summary portion. They support attribution during generation, rather than ranking sources by where they appear within the prompt's document set.

The GDM family's later grant and both applications specify a different `network input` : the question joined to one conditioning input. An optional prefix contains `k prompt tuples` showing the required answer format. These examples do not establish a document's rank among the sources supplied for generation.

The system runs each surviving passage separately and may generate an answer from it more than once. It does not list all passages in one ranked prompt, unlike the Google summary family's selected document set.

The Google family's memory-limit grant and 2024 application also describe adding familiarity text from a device or user profile. This text uses window space that would otherwise hold document spans. Interactions such as dwell, scroll pause, expand, and highlight can trigger a revised prompt. These loops appear in dependent claims and the description; treat their presence in any actual deployment as speculative.

Filings: US12118325B2, US11769017B1, US11886828B1, US20240220735A1, US20250005303A1, US12333436B2, US20230244934A1, US20240281659A1

Do private document rules apply to websites?

No, both Drive-scoped filings explicitly cover a cloud content management platform, rather than retrieval from the public web. The retrieval unit is a `document portion` : a sentence, paragraph, or section that the system represents through embeddings. The system embeds it through a **synthetic prompt** about that portion, then matches the generated prompt to the document portion using embedding similarity.

Whole-document embeddings may use a subset, such as the first five pages or a representative portion of the document. The application illustrates five pages as an example, rather than setting a threshold that implementations must follow. Permissions act as a hard gate, while last-opened and last-modified timestamps can filter which portions enter the subset. The filings leave filter order open.

The only public-web analogue here selects same-domain follow-up pages after showing a classic result, with conventional ranking happening first. That separate application describes querying an index of resources hosted by the result's domain to select follow-up pages. A page missing from that domain index cannot receive a **deep link**.

What else takes up space?

Instructions and memory occupy space that can crowd out retrieved spans when you run the assistant yourself. The OpenAI filing describes assembling prompts by swapping **state instruction blocks**. It removes the prior state's instructions to keep the prompt focused as the assistant changes operating states. It then appends **key-value pairs** to retain persistent memory in the prompt. It says nothing about web retrieval, ranking, or citation, and offers no evidence of how AI answers pick sources.

Two other filings describe cuts outside web retrieval, changing text eligibility before packing rather than specifying span truncation. The European application puts a subset of record text and **asset metadata** into a generated LLM query. That model may train only on enterprise data, with no retrieval step involved in supplying its answer context. The other US application uses per-user source parameter rules to decide which sources get searched, with a quality-controlled fallback.

What to change on the site

1. Win classic top-N ranking for queries your pages should answer. Both families take only a top N or top p of already-ranked documents into the generation set. (well supported)
2. Serve main answers as clean, extractable text in the HTML response. The GDM filings extract clean text from HTML before splitting it into segments. Text locked in scripts or images may therefore never become a conditioning input. (well supported)
3. Write self-contained passages of a few sentences that restate the query and answer it there. Snippet selection uses term overlap or embedding distance to the query. GDM scores sentence segments independently. (well supported)
4. Show revision dates and actually update the page. Freshness is a named query-independent measure for selecting documents whose content reaches the LLM. (well supported)
5. Put image captions, alt text, and video transcripts on the page. The filings name captions, recognized image text, object descriptors, and audio transcriptions as content eligible for the prompt. (well supported)
6. Give each answer an addressable heading so a verifying span can receive a link. Two filings describe rendered anchor links to the specific document portion that verified a summary sentence. (well supported)

7. Cover adjacent and follow-up queries on the same topic alongside the head term. Documents answering related and recent queries can join the summarization set when direct results are weak or homogeneous. (well supported)
8. Keep every important URL crawlable in the site index. A single filing describes selecting same-domain deep links from an index of resources hosted by the domain. A page missing from that index cannot appear after a user engages a result. (single filing)

Filings: US11769017B1, US12333436B2, US20230244934A1, US12118325B2, US11886828B1, US20240220735A1, US20250005303A1, US20250315463A1

How the sentence gets written

Once context enters the prompt, the filings describe two steps: write drafts, then select one for display.

Auto-regressive decoding means generating one token at a time. This process follows syntax, templates, or quote constraints, while selection scores several drafts and returns just one. A filing describes a proposed system, not a shipped product.

Which words can become a quote?

A quote can only reproduce a verbatim substring already in the prompt. These three filings share a 16 March 2022 priority and name DeepMind Technologies Ltd and GDM Holding LLC. They describe one auto-regressive pass that emits a claim, a document identifier, and a quote supporting that claim. Predetermined syntax tokens constrain the scores for possible next tokens, restricting which text the system can generate next. After the title-span token, only tokens continuing a supplied title keep non-zero scores. After the quote-span token, only tokens continuing an exact substring of the identified fragment keep non-zero scores. The identifier is typically the document title, which the system regenerates token for token to build the hyperlink. Section headers can also serve as identifiers.

Context sequences come from the highest-ranked search results, and each contains the search snippet plus surrounding source text. Truncation occurs at a sentence boundary within an example character window before the snippet, potentially excluding distant text from quotes. The same family samples multiple output sequences and uses a learned reward model. Human ratings of plausibility, support, and preference train that model to rank the sequences before the system selects one. If none exceeds a threshold, the system emits a default refusal instead of a candidate. The threshold, candidate count, and window sizes are examples in the filings, rather than claimed operating points.

Filings: CA3245108A1, EP4466630A1, AU2023236937B2

When does a summary show a citation?

A citation appears when an encoder, converting text into numerical representations, finds matching document content close enough. These Google LLC filings share a 30 December 2022 priority. After the system retrieves documents, they control whether any large language model runs, which model subset runs, and which documents supply the generated answer. Query text, a classifier, and features of responsive documents can select a model, a smaller model, or no model. The summary uses selected passages. Prompt selection uses snippet-to-query term overlap, similar terms, or word-embedding distance. Prompt text can include recognized image text, captions, object descriptors, and video transcriptions as source material. Injected familiarity text lets the summary omit content that the system treats as already known to the reader.

Citation follows generation as a verification step, with the encoder comparing a summary portion against the document content. It measures distance between their numerical representations, which controls whether a selectable citation appears immediately after that portion. An alternative uses source identifiers emitted by the model, and multiple verifying documents can produce multiple selectable elements. A confidence measure combines model confidence, verifier count, and the trustworthiness of the documents that verify the portion. It can annotate the portion, suppress extra results, or suppress the summary, based on confidence. Verifying source URLs can differ from rendered search results, and interaction with a result can trigger a revised summary. The filings give no numeric embedding-distance, dwell, or confidence thresholds.

Filings: WO2024145537A1, EP4627452A1

What shapes the answer used for training?

A language model program applies transformations when conditions match, then fixes the final text as its training target. This OpenAI Opco LLC published application has a 10 May 2024 priority and describes an offline teacher path. That path prepares answers for training: it selects search-result sources, writes headers, and evaluates conditions before generating section contents. It picks a guiding prompt relevant to the prompt or selected sources, then writes section contents from those sources. Example tree nodes control code, named-entity, how-to, style, technical, and weather branches. Templates include tables for sports scores or weather, while guiding prompts may constrain the answer's tone, style, or length.

The loop ends when the program is complete and no further guiding prompt applies to the response. An initial generative-engine response can also select the next prompt in the tree, shaping the next program step. Through distillation, another engine learns to reproduce the final response from the query plus results, without needing the guiding prompts itself. The live search response must include at least one citation to the search results used. The claims leave condition testing open, and the example branches do not establish a taxonomy of a live product.

Filings: US20250348482A1

Can an answer without source support win?

An unsupported reply can win when the system compares scores for supported and unsupported replies generated in parallel. The system shows the winner and discards the loser after comparing both reply types. These GDM Holding LLC filings share a 20 September 2022 priority; the first is granted, the second a published application. They sample one or more unsupported responses without retrieval and generate search queries. They also put returned-result content into a supported context input and sample candidates grounded in that supplied content. Evidence consists of a page title plus a document fragment, truncated through fuzzy string matching against the query. The user may see a returned result or subject matter derived from it in the response. Role-token log likelihoods can decide whether the system retrieves any external material at all.

Selection combines a preference score with a rule-violation score, and preference ranks supported candidates against the unsupported candidates. Per-rule violation probabilities can filter candidates at an unstated threshold or penalize them during re-ranking. The geometric mean of per-rule scores forms a combined rule-violation score, which feeds the final re-ranking score. The highest-scoring candidate appears, and example rules cover medical advice, hate or harassment, misinformation, and offensive content in responses. The preference model learns whether a response is plausible and supported by the evidence provided to the system. The description says clear supporting evidence earns higher preference scores, but that still allows an unsupported response to win.

Filings: US12450464B2, US20240104336A1

Which draft passes the quality checks?

The filing picks the candidate with the highest combined quality score, subject to thresholds for each individual criterion. This granted Google LLC filing has a 28 July 2021 priority and uses the same auto-regressive model to draft and score. For each stored criterion, the system appends natural-language attribute tokens and reads a rating from the token subset for that criterion, allowing the same model to assess different qualities. When ratings range from 0 to 1, the score of token “1” becomes the rating score for that criterion. A sum or weighted sum combines ratings into a quality score, but missing a per-criterion threshold can veto the highest-scoring candidate.

Example criteria include sensibleness, specificity, and interestingness, and the description assigns a generic reply a specificity score of 0. This filing has no retrieval or citation step; it constrains a dialog system’s answer layer rather than source selection. It gives no numeric thresholds or quality-score weights for these checks.

Filings: US12086713B2

What can beat the first draft?

Later scores from search-quality signals or a search-trained reward model can favor another draft over the first generated sample. These Google LLC filings differ on what their scores measure and whether search content remains in the prompt at inference. **Inference** means generating output with a trained model. The first filing has a 7 February 2024 priority and associates a candidate with a stored resource before scoring. It then applies that resource’s search-quality score, using sequence similarity, semantic similarity, text matching, or embedding look-up for association. Document-level scores transfer to sentences or paragraphs, and inference scoring operates on those units rather than individual tokens.

A ranking engine derives quality scores from content and metadata, incoming versus outgoing links, references, interactions, and content type. Academic citations weigh more than a footer link, and content-type examples contrast academic papers or encyclopedias with social posts or unmoderated blogs. Reward-

model fragment scores can select the displayed fragment or the generative model handling the task, or stop remaining low-value subtasks. The claims allow direct quality scores without a reward model at inference, and the filing does not promise the publisher a citation.

The second filing has a 7 December 2023 priority and trains a preference for output conditioned on search-engine content. It prefers that output over raw output generated without that content, with a text summary of results joining the conditioned training prompt. Alternatively, the model chooses which of two raw instances makes more sense given the summaries supplied during training. After fine-tuning, the method is designed to remove search content from the prompt. At inference, an optional solicitation signal from the trained reward model scores each decoded candidate. Dialog-session context, historical sessions, device location, time, and day of week can join that score, whose threshold remains unspecified. Here, factually correct means grounded in search-result reference material; it does not mean that someone has independently verified the truth.

Filings: WO2025170579A1, US20260010575A1

What to change on the site

Make source text easy to quote, identify, and assess. These actions follow from the filings, within the evidence limits described above.

1. Write each key fact as a self-contained sentence someone can quote. Constrained decoding can only emit an exact substring of the prompt fragment. The exact-quote family's reward model learns whether that quote supports the claim. (well supported)
2. Give every citable URL a distinctive, stable, natural-language title. Use clear section headers. The DeepMind citation filings use the title, and optionally a section header, as the identifier regenerated token for token. (well supported)
3. Put quotable sentences next to the passage likely to become the search snippet. The Canadian exact-quote filing describes truncating context around the snippet. The international Google summary filing selects prompt text through snippet-to-query overlap, similar terms, or embedding distance. (well supported)
4. Structure sentences and paragraphs so each can receive its own score. Make claims other selected documents can corroborate. The first completed-draft scoring filing copies document quality scores onto sentences or paragraphs. Both summary filings raise confidence with verifier count and trustworthiness. (well supported)
5. Publish extractable body text, including captions, transcripts, and how-tos that complete the task. The second completed-draft scoring filing joins a text summary of results to the preferred training prompt. Both summary filings describe generation consuming captions and transcripts. The first completed-draft scoring filing gives a how-to that solves a task higher utility than open discussion. (well supported)

6. If the query class has a template, supply source text that fits it. The OpenAI filing's teacher writes headers, then section contents from selected sources. Conditions control branches, with templates such as sports or weather tables. (single filing)
7. Keep wording exact and stable for claims you want quoted. The three exact-quote filings describe constrained decoding that copies tokens from the fragment. A rewritten sentence may therefore cease to be a valid continuation. (well supported)
8. Avoid categories penalized by reply-layer rules, even when retrieval selects your page. The Sparrow filings score candidates for medical advice, hate, misinformation, and offensive content. Those violations can filter out or down-rank a candidate before display. (well supported)

Filings: CA3245108A1, EP4466630A1, AU2023236937B2, WO2024145537A1, WO2025170579A1, EP4627452A1, US20260010575A1, US20250348482A1, US12450464B2, US20240104336A1

The checks that run after the answer exists

What can reach the reader?

Across eight Google filings, a separate checking process can attribute, edit, flag, or withhold an already drafted answer.

The two 2026 matching applications, the matching grant, and the location-confidence grant place checks between generation and display. The annotation filing adds annotations after generating the answer and extracting fragments, displaying them at or just after client display. The retrieval-loop application and rewrite filing add another round of work when the initial answer lacks support. Unsupported statements can trigger retrieval of more **passages**. Incomplete agreement can prompt a rewrite into a second text.

The citation-control application describes two decisions, starting with whether the response generator should use retrieved passages at all. After generation, its **corroborator** attaches citation controls to text spans containing facts. These are filings, not shipped products, but their shared direction makes a separate checker worth understanding. That checker decides what readers can see, separately from the **decoder**.

Filings: US20260064780A1, US20260064984A1, US12596889B2, US12585658B1, WO2025064436A1, US20250156456A1, WO2024073087A1, US20240289395A1

Can paraphrases earn citations?

Paraphrases produce no match in the claimed check, which requires copied letters and numbers, matching fingerprints, or sufficiently few edits.

The two matching applications and matching grant describe the same check, beginning with lowercasing the text for comparison. They replace punctuation and extra whitespace with single spaces before comparing response segments with overlapping source-text windows. Each window starts at a word boundary and has a tunable, fixed character or word length for comparison. The first matching application strips ads, images, and other audiovisual content from webpages before making this comparison.

Both matching applications allow a one-to-one **alphanumeric** copy or a **hash** hit, matching computed text fingerprints. Another route uses **edit distance**, counting insertions, deletions, and substitutions. The checker divides that count by the additional-data character count and tests the result against a developer-tunable threshold. Both applications optionally use **cosine similarity** with **word2vec** or **bag-of-words** representations. This option applies only when edit distance fails the threshold but falls within a threshold range of it.

The matching grant allows a threshold quantity of matching alphanumeric characters and illustrates **shingles**, overlapping groups of words. Its six-word shingles are a worked example, rather than a fixed parameter; paraphrase produces no match. The first matching application treats unmatched wording as **generative content**, wording produced without a source match. That wording can appear without a source link.

After matching, all three filings describe a **rules engine** that checks source type. The first type, illustrated with public-domain content, can receive a source **URL**. The second type receives a source link plus a license URL, while the third type can trigger truncation. The system can truncate at N characters, optionally add an ellipsis, or omit or regenerate that text.

Both applications compare match length with developer-set integers N and M to link shorter matches and truncate longer ones. The first gives 100, 200, and 300 characters as description examples, rather than constants required by claims. The matching grant resolves ties between sources using publication date or alphabetical order when selecting a source.

Both 2026 applications describe leaving user-provided content unmodified. The second can also show a selectable pointer into the user's document. In the second application and matching grant, dependent claims or the description contain source-type and regeneration rules. **Dependent claims** incorporate another claim's requirements; **independent claims** stand without them. The independent claims instead require the user-content skip and third-party hyperlink indication. The "first type of source" is not a published classification of the open web.

Filings: US20260064780A1, US20260064984A1, US12596889B2

What happens to unchecked claims?

Depending on the filing, the checker can flag, rewrite, reject, or leave an unchecked claim without an annotation.

The retrieval-loop application scores each statement s_i against candidate passages using $\phi(e_i, s_i)$ to assess their support. It creates a citation only when the best passage clears a first threshold τ in that comparison. If every passage scores below a second, lower threshold, the statement enters set U and receives no citation. The system sends that statement back as a new query to a **dual-encoder retriever**, encoding queries and passages separately. Each round adds only supplemental passages absent from earlier iterations, and **inference budget** B limits model processing. That budget and other stopping conditions end the loop. The filing leaves τ , the second threshold, and B unspecified. It also leaves the contents of **corpus** D unspecified.

The rewrite filing identifies claims and generates queries using **few-shot prompting** and **chain-of-thought prompting**. It retrieves evidence through web search or a proprietary corpus, extracting snippets with a four-sentence **sliding window**, a moving selection of consecutive sentences. That window stops at document headings, and the checker labels each claim supported, contradicted, or unaddressed.

Incomplete agreement triggers a second text intended to improve agreement, subject to edit-distance and agreement-plus-preservation checks. These checks also use their **harmonic mean** to decide whether the rewrite passes.

Reported experiments show 13% absolute attribution lift, with 10 to 20% of text changed by the process. The experiments also report intent preserved over 90% of the time. These are experimental results rather than deployed thresholds, and the AIS/NLI scorer measures evaluation performance, not runtime agreement. **AIS/NLI** combines attribution to identified sources with natural-language inference.

The annotation filing extracts fragments, including statements, complete sentences, or independent clauses, before filtering them for further checking. It retains fragments suitable for **entailment**, evidence logically supporting a statement. Its filter uses a **knowledge graph** hit or a capable/incapable classifier. That classifier judges suitability; alternatively, a **regression entailment score** filters fragments. The filter drops greetings and thanks, and each remaining fragment becomes a separate search query for comparison.

The comparison uses corroboration e and contradiction c , demonstrating incorrectness as $z = ac + (1-a)(1-e)$. Demonstration thresholds of 0.75 and 0.9 route fragments to contradicted, corroborated, or “no relevant content found”. Neither those cutoffs nor weight a are limiting requirements; the filing also allows analysis to await user action. The system can skip a site “deemed to lack credibility” or mark it weak during this process.

The location-confidence grant creates an **embedding**, a numerical representation used for comparison, for each subsequence of an `initial output sequence`. It uses **intermediate-layer hidden states**, then searches databases organised by network location. It selects neighbours by smallest embedding distance or a distance threshold, and each database has a confidence level. Network location, location offset, and last-modified timestamp raise or lower the score, with the timestamp’s direction left open. A **location offset** identifies a position within a source.

A predetermined threshold governs keeping, modifying, or rejecting each subsequence, but the grant provides no numeric threshold. Unknown locations inherit a level through embedding similarity to databases that already have levels assigned to them. The method neither retrieves documents into the prompt nor chooses citations.

The citation-control application matches spans containing facts to supporting resources. It can display a **favicon** or hyperlink control for the resource. When it identifies a single corroborating resource, the favicon itself acts as the control for that resource. Supporting resources are documents the search engine considers responsive. Selection can optionally stop at the top n results or require a minimum relevancy score for inclusion. The system strips **markup** and non-text content before processing supporting resources.

Relevant portions exceed snippet length but stay under 2000 characters; alternatively, the system concatenates the top two portions. The filing does not specify how it decides a span contains a fact. Illustrative selection settings include lengths of 300, 500, or 1000 characters and caps of top 5 or top 10.

Relevancy examples of 80% or 90% are also illustrative. Snippet-response contradiction serves as a rewrite training example, rather than a documented runtime switch for refusing an answer.

Filings: US20250156456A1, WO2024073087A1, WO2025064436A1, US12585658B1, US20240289395A1

What wording can checkers verify?

Checkers can more readily verify standalone, distinctive sentences containing facts in body text that crawlers can access.

The first matching application strips ads, images, and audiovisual content, so its text checker never sees those elements. The citation-control application also removes markup and non-text content from supporting resources before encoding them for comparison. The location-confidence grant stores each subsequence's location offset and says body content can score higher than a trailing chat forum.

The annotation filing drops fragments without a verifiable fact, then searches using each remaining fragment as a query. The rewrite filing scores four-sentence windows that stop at headings and keeps a subset maximizing coverage across claims. The retrieval-loop application pairs one passage with one statement; implied claims enter U, the set without citations.

Distinctive phrasing matters across the three matching filings: near-copies can pass the edit-distance check, while paraphrase produces no match. Your page must first rank into the retrieved set for these checks to match or cite its wording. The first matching application's initial attribution pass compares the response only with retrieved search result documents. The retrieval-loop application permits citations only to dual-encoder hits. The second matching application can skip attribution entirely for a user-pasted copy of your work in its checks.

Filings: US20260064780A1, US20240289395A1, US12585658B1, WO2025064436A1, WO2024073087A1, US20250156456A1, US20260064984A1, US12596889B2

When do checks stop?

Specified conditions stop checks or change individual segments, without requiring the checker to prove every claim first.

The first 2026 matching application checks search result documents first, then the training dataset if that search finds no match. It also allows training-set hashes to run before or after search result documents, providing another possible order. The second 2026 matching application checks the smaller user or **RAG** set first. RAG means retrieval-augmented generation. This order avoids scanning an index containing hundreds of billions to trillions of segments during initial checks. Separately, the second application says hashes can change the order so training hashes are checked before user content. The filings contain conflicting accounts; neither order can be assumed to take precedence.

The retrieval loop can reach inference budget B, while the rewrite process can fail its rewrite checks. Across the three matching filings, third-type content can trigger regeneration using **temperature**. Regeneration can also use **word or phrase biasing**. Other outcomes include subsequence rejection in the location-confidence grant, or skipped annotation in the annotation filing. The latter can instead show a “no relevant content found” annotation when its checks reach that outcome.

The filings provide neither a numeric budget nor a rule withholding the entire answer when these conditions arise. Refusal applies to a segment, statement, fragment, or subsequence, so distinguish those outcomes from withholding an entire answer. The location-confidence grant requires the system to maintain membership in its predetermined list of known knowledge base locations. A site cannot opt into that list.

Filings: US20260064780A1, US20260064984A1, US20250156456A1, WO2024073087A1, US12596889B2, US12585658B1, WO2025064436A1

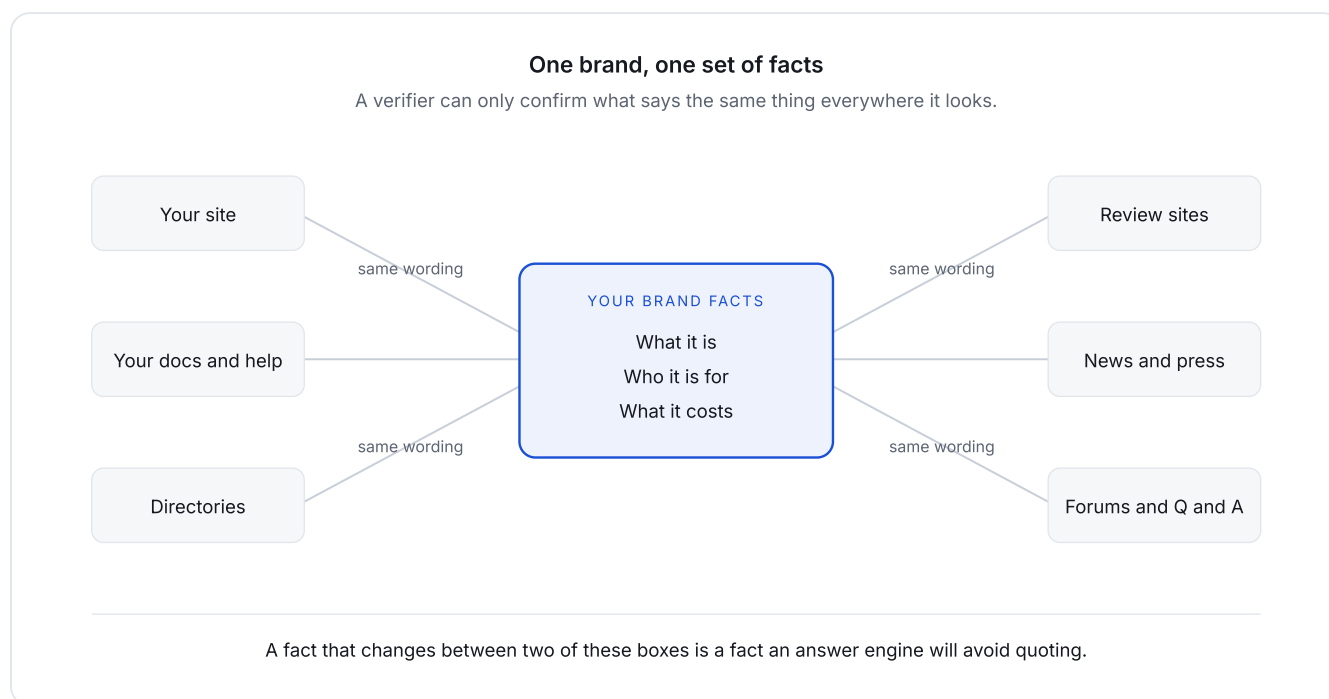
What to change on the site

1. Write self-contained sentences containing facts in the main body. Keep them out of comments, images, or ads. Several filings check sentences, fragments, or subsequences. They strip or downweight content outside the body before comparison. (well supported)
2. Keep distinctive, stable phrasing on pages you want detected as a source. The three matching filings compare normalized overlapping windows and use edit distance. Near-copies can pass that check; paraphrase produces no match. (well supported)
3. Rank for the user query and queries built from claims or fragments. Checkers retrieve from search or a dual-encoder working set. Text outside that retrieved set cannot be matched or cited. (well supported)
4. Publish a crawlable source URL and, for licensed pages, a reachable license URL. After a text match, first-type rules can attach a source link. Second-type rules can attach a license link instead of truncating. (well supported)
5. Put the sentence answering a checkable claim inside a contiguous four-sentence span under a heading. The rewrite filing extracts snippets using a four-sentence sliding window that stops at headings. (single filing)
6. Cover sub-questions a first-pass answer leaves open, without near-duplicate passages. The retrieval-loop application sends unsupported statements back as retrieval queries. It adds only supplemental passages absent from earlier iterations. (single filing)
7. Maintain site-level credibility and a recognizable favicon. The annotation filing can skip a site it deems to lack credibility. The citation-control application can use the favicon as the citation control. (speculative)
8. Treat unlicensed copyrighted text that may enter a training index as a truncation risk, rather than a citation strategy. Third-type rules in the matching filings can truncate, omit, or regenerate instead of linking. (well supported)

Filings: US20260064780A1, US20240289395A1, US12585658B1, WO2025064436A1, US20260064984A1,
US12596889B2, US20250156456A1, WO2024073087A1

CHAPTER 8

Which source gets the link



Facts that stay identical across every page are the ones a verifier can confirm.

What earns a citation?

The Google summary filing grants a citation when a generated **span** sufficiently matches document content. It compares the span's **embedding** (a numerical representation of text) with representations of document content. An **encoder** creates those representations, whose distance measures their similarity.

The granted filing checks the match after retrieval and **LLM** summary generation, but before display. Claims 3-4 and 15-16 use embedding distance to trigger a selectable link immediately after the span. The filing specifies neither the encoder type nor a numeric distance cutoff for triggering that link.

Source identifier **tokens** let the model report which document supports each generated span. These appear in the **prompt**. That report feeds **linkification**, described in blocks 260A1 and 260B. The brief does not say whether this takes priority over the embedding match. The match checks document content rather than a URL, so write clear, self-contained sentences that provide easy verification units.

Filings: US11900068B1

Which pages can supply a quote?

The DeepMind filing permits only pages supplied as **context sequences** (source text packed into model input), with natural language identifiers. Its **auto-regressive model** emits three parts in order: a claim, one supplied identifier, then text from the context sequence that identifier names (claim 1).

One disclosed syntax is `%<Claim>%(Document title)%[Quote from document]%`. Syntax tokens mark the identifier and quote regions. Some implementations overwrite **vocabulary scores**, permitting quote tokens only from the identified context sequence (claims 8-10). This restricts output at the token level instead of checking embeddings. Only some implementations use **constrained sampling** (restricting which tokens models can choose), so a deployment could emit unconstrained quotes.

The filing claims only a verbatim span from a named retrieved document, without guaranteeing sufficiency, currency, or factual correctness. The identifier typically consists of a document title or section header that resolves to a hyperlink (claims 1, 3, 12).

Filings: WO2023175089A1

What does the reader see?

The Google summary filing displays links after verified spans; the DeepMind filing displays a title, verbatim quote, and resolvable hyperlink. The Google filing allows multiple documents to verify one portion, with one link per verifying document (claim 1). Their number and trustworthiness feed a confidence measure that controls how that portion appears to readers (block 262C). The filing gives no numbered confidence tiers.

The DeepMind filing uses a learned **reward model** to score each combination of query, response, and quote. It ranks N candidates and displays the highest-scoring candidate. A score below an unspecified threshold can trigger a default refusal (claims 4, 5). The filing leaves both N and the refusal threshold unspecified.

Both filings show users a checkable pointer into a retrieved document, but differ on whether several sources can share one statement.

Filings: US11900068B1, WO2023175089A1

Which pages can still earn citations?

Only pages in the retrieved candidate set can receive links or supply quotes under either query-time filing. The Google summary filing selects search result documents using three groups of measures, described in block 254A:

- Query-dependent measures include positional ranking, selection rate, locality, and language.

- Query-independent measures include trustworthiness from author, domain, and inbound links, plus popularity and freshness.
- User-dependent measures include profile attributes, recent queries, and recent non-query interactions.

Documents from related, recent, or implied queries enter only when query-responsive results have low quality or lack diversity. They must also meet correlation or overlap thresholds, whose cutoffs the filing leaves unspecified (blocks 254-259).

User-dependent measures mean two testers can see different cited sources for the same query. The other two briefs have no counterpart to that personalization.

The DeepMind filing sends a derived query to a search engine and selects context sequences from returned documents. Its examples take the top k results, with k set to 5, 10, 20, or 100. It then shortens long documents around the **search snippet**. Those figures are examples rather than claimed constants. Query and result features in the Google summary filing also control whether the system uses any LLM at all (claims 8-9).

Filings: US11900068B1, WO2023175089A1

Can a later search find sources?

The Google summary filing can find verification candidates through a fresh search on the generated summary portion itself. A document absent from the generation prompt can therefore earn a citation link through this search (block 358B).

This is the largest difference between the two attribution filings. The DeepMind filing limits quote sources to context sequences packed from search results. The Google summary filing allows generation and citation to use different document sets.

The Google system still selects prompt snippets by query terms, similar terms, or query-embedding distance (block 260A1). The filing gives no numeric threshold. Prompt content can also include auto-generated image captions, text recognized inside images, detected object descriptors, and video transcriptions (block 260).

After display, interaction signals can trigger a revised summary (block 458). These include link selection combined with a **dwell time** threshold, scroll pausing, expanding, and highlighting. The filing leaves the dwell time threshold unspecified.

Filings: US11900068B1, WO2023175089A1

Does page ranking choose citations?

The Google training application adjusts **attention weights** (values controlling how strongly tokens interact) during training, rather than choosing query-time citations. A trainer splits a document into tokens, obtains per-document signals, and computes conventional **query-key-value attention**. It increases weights in the **attention matrix** when the page meets a relevancy threshold. If the page fails, it decreases or maintains those weights (claims 5, 6, 10).

Claim 2 lists author, ranking, traffic statistics, creation date, modification date, and an interaction metric consisting of page clicks. Claim 7 covers static ranking signals, including inbound links. The application gives no numeric threshold and describes no query-time attention adjustment. It therefore cannot prove that an AI answer boosts high-ranking pages at answer time. The adjustment applies only to pages entering the **training corpus**, as claim 10 specifies.

Author, inbound links, freshness, and popularity also appear as query-independent selection measures in the Google summary filing (block 254A). The two Google filings favour similar qualities, but that overlap does not establish that the same module chooses citations. The DeepMind filing uses search-engine ranking only to select documents for the example top k.

Filings: US20250103662A1, US11900068B1, WO2023175089A1

What must a page provide?

The DeepMind filing requires a distinct natural language identifier and a continuous passage beside the snippet to copy. The model emits that identifier token by token. The input therefore includes a unique, stable title, or a unique section header for a long document. The model then emits that title or header in its output (claims 1, 3, 12).

Shortened fragments start at a sentence or paragraph boundary to keep the quote well formed. Constrained sampling can quote only a continuous span from that sequence, built around the search snippet. The quoted passage needs a stable URL so the system can resolve the identifier to a hyperlink.

The Google summary filing requires no model-generated title. It needs content with an embedding close to the summary span, plus trust and ranking signals that secure selection.

Filings: WO2023175089A1, US11900068B1

What to change on the site

1. Write each citable fact as a short, self-contained sentence. The Google summary filing checks a summary portion against document content using embedding distance (claims 3-4). The DeepMind filing starts shortened fragments at a sentence or paragraph boundary. (well supported)

2. Keep that sentence in a continuous passage beside the likely search snippet. The DeepMind filing shortens documents around the snippet and can restrict quotes to that sequence (claims 9, 10). The Google summary filing selects prompt snippets by query terms or query-embedding distance (block 260A1). (well supported)
3. Give every indexable URL a distinct title whose tokens stay stable, and give long pages distinct section headers. The DeepMind filing packs that title or header as the identifier the model must emit. The system can resolve it to a hyperlink (claims 1, 3, 12). (single filing)
4. Earn inbound links and publish under named authors on a reputable, crawlable domain. Author, domain, and inbound links feed the Google summary filing's trustworthiness measure for selecting documents to summarize (block 254A). Author and link-based ranking boost training-time attention in the Google training application (claims 2, 7). (well supported)
5. Update content when facts change so the modification date reflects a real edit. Freshness is a query-independent selection signal in the Google summary filing (block 254A). It is also a listed relevancy signal in the Google training application (claim 2). (well supported)
6. Aim for retrieval of individual passages through both the query and restatements of the fact, including outside today's top results. The DeepMind filing can quote only retrieved context sequences (claim 11). The Google summary filing can still attach a citation from a fresh search on the summary portion (block 358B). (well supported)
7. Add machine-readable captions, on-image text, and video transcriptions to non-text assets. The Google summary filing can pack auto-generated image captions, recognized text, object descriptors, and video transcriptions into the prompt (block 260). (single filing)
8. Treat more than the top result as a possible citation source. The Google summary filing lets one span carry one link per verifying document. Confidence rises with the count (claim 1; block 262C). The DeepMind filing emits one identifier per described output sequence (claim 1). (well supported)

Filings: US11900068B1, WO2023175089A1, US20250103662A1

How answers change beyond search

These filings describe work beyond retrieval followed by generation. They cover drafting, refreshes triggered by crawling, training, changes after ordinary ranking, and context across conversation turns. A patent is a filing, not a shipped Search, Gemini, Drive, or OpenAI product.

Whose history shapes the answer?

Three filings describe how user history and device context shape snippets, shortcut answers, or suggested prompts. They cover search history, prior queries, and recently opened files, but each interface needs separate consideration.

The first filing describes an overlay with a long press as a claimed trigger. A lasso or scribble selects a region; a segmentation model separates objects within images and links visual features to an object. Stored data includes a bitmap, source data (URL, origin, author, publisher), and search history. Snippets form groups around shared topics, including a particular entity. Claim 16 places the snippet beside a model-generated response and web results on a search-results interface. Text-fragment start/end markers and scroll position reopen the source with the saved portion highlighted. The claims leave those three surfaces' order unspecified, and the filing leaves capture opt-out unstated.

The second filing describes a shortcut on retrieved content, selected when a resource-specific model's confidence score meets selection criteria. A bidding process can also contribute; insufficient publisher training data suppresses the shortcut. Tapping builds a prompt from the query, prior queries, session data, location, and recent actions. Follow-ups can come from FAQ subdomains. OCR, reading text from images, validates each generated answer against the source page at a claimed pixel coordinate. The description lets owners request exclusion; the filing claims no effect on native organic ranking. It gives no confidence formula.

The third, granted filing concerns a cloud content-management platform's own documents, rather than public web ranking. Document portions become generated prompts and embeddings (numerical representations for comparing content). Images contribute through OCR; audio and video contribute through transcripts. Last-modified, last-opened, creation, or last-comment timestamps determine which documents enter the working set. Embedding similarity to a recently opened document can also determine membership. Permissions metadata filter documents before the model answers. Independent claims cover prompt generation; the filing gives no numeric threshold.

Filings: WO2025183818A1, EP4544424A1, US12608562B2

Which pages qualify as evidence?

The filing describes fact classification using result data sets that match the fact's topic and meet a recency threshold.

Checks happen during drafting in a link-notes interface before publication or indexing. A language model parses the draft and classifies segments as fact or opinion. Only facts enter checking; queries generated from those statements retrieve result data sets. Topic association, recency, and credibility scores further filter rows; the annotated page's own credibility level weights its details. Each resource suggestion includes a title, a deep link, and passages matched to the fact. The description says notes and interactions may later adjust the web resource's rankings, tagging, embedding, and indexing.

The filing gives no numeric recency, credibility, or confidence values. Ranking adjustment appears in the description, rather than a claimed formula. This offers no evidence that Search fact-checks drafts today.

Filings: US20260072977A1

When can stored answers change?

The filing describes a crawl or upload starting a refresh of stored answers, without a fresh question.

Historic prompts, prompt vectors, and outputs already sit in a data store. A new document becomes a query vector; k-nearest-neighbor search (finding the k closest vectors) identifies update candidates. Similarity measures include inner product or cosine. Other options are Euclidean or Hamming distance. The chosen k is much smaller than N, the total vector count. One example uses one million vectors with k less than one hundred.

For each neighbor, the system compares the historic output with the new document against an update distance threshold. This threshold sets the difference needed for rewriting; in the embedding example, greater distance triggers a rewrite. When the two are too close, the system builds no input sequence. A separate presentation distance threshold withholds a rewrite when the new sequence is too similar to the old one. The description's example format includes a link to the new document.

The filing leaves k, both thresholds, and the embedding architecture unspecified. The claims do not require the updated sequence to cite the new document. The description also applies the process to dialog, translation, and code; search answers are one embodiment.

Filings: WO2026015136A1

Which pages teach the model?

The filing describes offline training that feeds the base model top search results and teaches a distilled model to cite responsive sources.

Offline training happens before live use; a base model supplies outputs to train a distilled model. The filing belongs to OpenAI Opco LLC, rather than Google, and describes training rather than live ranking.

Source selection starts from the search index or from analysing content for responsiveness. The description treats the top 3, 5, 10 or 20 results as the most relevant pool; these are examples. Only sources that answer the query enter the generation prompt. A language-model program then branches according to the query and sources; topic or format categories are expressly examples. A query that “can be improved” may be rewritten before retrieval. Training instructs citations to the specific pages behind the answer; the served response carries at least one citation.

The filing gives no responsiveness or query-improvement scores. It does not establish how any live OpenAI engine selects pages today.

Filings: US12619603B2

Does grouping results change their rankings?

Both filings describe reorganising already-ranked documents after conventional retrieval, without claiming changes to the underlying web rankings.

They share priority 2023-05-16 and the same mechanism. A selector takes the top X ranked results, with examples ranging from ten to one hundred. A query breadth value can suppress the layer for narrow answer-type queries. A detected-theme count at or below a threshold can suppress display; examples are one or two themes.

The model summarises passages alongside titles, metadata, and neighboring passages. Here, passages are HTML paragraphs or sections bounded by headers. Training uses header-passage pairs; the model may paraphrase a header or standardise its wording into the summary. Similar summaries form clusters. The summary nearest the centroid becomes the displayed theme phrase. Theme ranking uses a unique result signal: the number of distinct documents mentioning the theme. It also uses query-relevance, quality, authority, and popularity signals. Textual descriptions of images and videos count as passages. Unstructured content supplies themes, rather than the knowledge graph.

The filings specify no clustering algorithm, similarity metric, or concrete X. The first filing is granted; the second is an application. “Mentioning” a theme means a passage summary belongs to its cluster, rather than providing a verified citation.

Filings: US12158907B1, US20240386046A1

What page structure helps extract facts?

The filing describes an extractor that needs correct page structure and a contiguous literal (an uninterrupted sequence of text).

The DOM must be correctly nested and use standard tags. The value must share its field's text node. The filing describes WebFormer working after parsing. Answer or index layers then consume structured fields.

For a named field, WebFormer selects the exact text span holding its value. H2H attention (which connects HTML elements) runs over the DOM graph plus sibling edges at edge distance 2. Removing H2H produced the largest measured drop. The tag name supplies half of each HTML token embedding. H2T attention connects HTML elements to text and updates an HTML token only from its own text nodes. The highest-probability span wins; the filing states no confidence threshold.

Error analysis found substring, multiple-occurrence, and range mistakes, including extracting 19.90 from 19.90-26.35. Labeling selected the first occurrence when an answer repeated. Schema.org annotations served as ground truth. The experiments used Common Crawl.

This provides neither a ranking nor a citation signal. Nothing shows Search uses WebFormer.

Filings: US20250148025A1

Can publishers influence these training processes?

Publishers have no lever in these filings: one corrects retrieval during training, while two score analytics interpretations.

The first filing trains a dual encoder. It avoids re-encoding the whole corpus at each step. A correction model maps stale buffer embeddings to corrected embeddings. The k embeddings most similar to the query embedding identify the subset to encode afresh. Independent claims leave use of the corrector at serving time optional. The trainer controls k, drift loss, and the buffer refresh schedule. Corrected embeddings are not a public ranking signal.

The other two filings concern analytics insights cards rather than web ranking. They score interpretations after cards render, using signals publisher pages never generate. A first model scores a query-interpretation pair; a second combines that score with temporal and cluster features. A follow-up query counts as a search refinement when it arrives within a predetermined interval and exceeds a similarity threshold. That refinement provides a weighted indication of an incorrect interpretation; viewing a report provides a weighted indication of a correct interpretation. Numeric thresholds such as 0.8 or 75% are examples only. Neither filing selects or cites documents.

Filings: US20250363376A1, US12380100B2, US20230334045A1

What to change on the site

1. Keep a stable public URL and an in-page locator (text-fragment start, end, scroll position) for every passage containing claims. The snippet filing stores source and location data so a saved crop reopens with that portion highlighted. (single filing)

2. Name entities on the page and make individual objects easy to isolate visually. Write body copy with clear content-item boundaries. Snippets can form groups around a shared entity; a lasso plus segmentation extracts the object from the selected region. (single filing)
3. Keep pages fresh and on-topic, with a title matching the claim and passages that are easy to extract. During drafting, fact checking keeps result data sets only when they match the fact's topic and meet a recency threshold. It then suggests a title, deep link, and matched passages. (single filing)
4. Get the page crawled and add facts missing from the already-served answer, in a form an updated sequence can reference. A new document starts neighbor search over stored prompt vectors. Excessive similarity to the historic output prevents a rewrite; excessive similarity between the rewrite and historic sequence withholds it. (single filing)
5. Earn a top ranking for the target query. Make the page's answer explicit, self-contained, and citable. Distillation selects from the index's top results, keeping only responsive sources and citing those pages; thematic grouping also reads only the top X ranked documents. (well supported)
6. Organise clear passages under descriptive headers. Write precise titles and metadata, and fill image and video metadata, tags, and captions. Both thematic filings use HTML-defined passages and header-passage pairs for summaries and clusters, treating media textual descriptions as passages. (well supported)
7. Keep each field value as one contiguous literal inside a correctly nested DOM with meaningful HTML tags. Declare those values in schema.org where possible. The structure transformer uses sibling relationships and tag embeddings; Common Crawl experiments used site-provided schema.org as labels. (single filing)
8. Keep key facts as parseable text at stable page coordinates. Maintain an FAQ page or subdomain if you want suggested follow-ups. Conversation shortcuts validate generated answers against page content at a claimed pixel coordinate through OCR. Follow-ups can come from FAQ subdomains. (single filing)

Filings: WO2025183818A1, US20260072977A1, WO2026015136A1, US12619603B2, US12158907B1, US20240386046A1, US20250148025A1, EP4544424A1

Five things the filings do not support

Is matching the typed query enough?

Matching the user's wording alone can miss every search described here, because the filings treat it as input.

Two filings combine session searches; two others describe **facet subqueries**. Another rewrites a pronoun into a search for the named thing, while another combines original **tokens** with model-emitted terms. Two further filings construct new searches from chat state, so consider the conversation beyond the typed query.

Filings: US11762848B2, US12242472B2, US20250117381A1, WO2025080561A1, US20250173519A1, WO2024229375A1, US20240289407A1, WO2024182453A1

Does retrieval compare one whole page?

Retrieval can compare smaller units than pages, so one numerical page representation cannot explain these filings.

The FDE filing uses **k-nearest-neighbors** over a **Chamfer/MaxSim proxy**. It does not use **cosine similarity** of a **pooled page embedding**.

Other filings store numerical representations of phrases, sentences, text sections, tokens, or documents; the stored unit varies. A retriever using tokens encoded with context can admit a document on just one matching token hit. Retrieval searches neighboring representations of that unit; two further filings instead describe models that emit identifiers.

Filings: US20250363101A1, WO2024215344A1, US20250217373A1, US11003865B1, US20250165469A1, WO2023154385A1

Will a paraphrase earn a citation?

Paraphrases produce no match in the string-match family, whose three filings compare text windows after lowercasing and collapsing punctuation. They use alphanumeric copying, equal **hashes**, or **edit distance**. They treat unmatched wording as generative content, which can appear without a source link in the output.

Two constrained-quote filings allow only an exact substring from a packed fragment. Another filing compares numerical text representations for citations, requiring a self-contained sentence close to the generated span. A restated gist does not suffice, so preserve a sentence that can stand alone as a source.

Filings: US20260064780A1, US20260064984A1, US12596889B2, CA3245108A1, WO2023175089A1, US11900068B1

Do reported results establish live product behaviour?

A filing claims a method; it does not establish shipped behaviour in Search, Gemini, or ChatGPT products.

The two filings in the **graph reranker** family report **open-domain question-answering**, or ODQA, metrics on Natural Questions and TriviaQA. Neither ties its mechanism to Google Search; the separate **pairwise neural ranker** describes embodiments using application-store logs.

Three further filings retrieve Drive, custom-corpus, and review-database collections respectively, rather than ranked pages from the web. Convergence across families offers a usable signal, but one independent claim cannot establish a live ranking rule.

Filings: US12511325B2, US20250335486A1, US11989649B2, US20250103826A1, US20240362093A1, US20250252121A1

Can later scoring rescue a missed page?

Later scoring depends on initial retrieval: LITE's second pass never scores documents excluded by its first-pass cut.

Candidate generation filters the pairwise ranker's scoring set; GDM excludes documents below cutoff p from segmentation, scoring, and conditioning. Sliding-window pairwise prompting starts with the initial ranking and limits bubble-up passes, which move documents upward through comparisons.

One Google summary filing can attach a citation through a fresh search on the generated text span. That searches the draft afresh, rather than ranking an unfetched URL in a second pass.

Filings: WO2024226614A2, US11989649B2, US12333436B2, US20250124067A1, US11900068B1

Your plan for better search visibility

Make your pages available

During days 1 to 30, make important pages accessible, competitive in search, and clear about their subjects.

1. Make every important URL crawlable for inclusion in search. Publish each important URL into the public document collection that the search process actually searches. The filings describe initial retrieval that never scores documents absent from its encoded, searchable collection.
2. Serve main answers as clean, extractable HTML text. Filings about GDM describe extracting HTML text before dividing it into sections. Text locked in scripts or images may never become part of the input that conditions the model.
3. Win conventional rankings among the top N results for the queries that your pages should answer. Both summary families select only the top N or top p already-ranked documents for generation.
4. Publish one page that fully answers each aspect you sell or explain, titled as a clear single-problem question. Two filings describe retrieval against simple, relevant questions that large language models generate from the original query. These filings state that whole-query results address each of the query's individual aspects only superficially.
5. Put the content that distinguishes your topic at the opening of every page and every passage. Direct indexing links leading tokens to a document identifier. Combining the first block can represent the whole document; portions used for diversity contain less than 500 characters.
6. Give titles and URL paths human-readable subject names, because retrieved titles and URLs annotate labels describing search intent. Later quotation filings treat the title as the identifier that the text-generating component must output.

Filings: US20240362093A1, US11003865B1, US20250315463A1, US12333436B2, US20230244934A1, US11769017B1, US20250117381A1, WO2025080561A1, WO2023154385A1, US11694034B2, US20260072965A1, US20240232637A9, CA3245108A1, WO2023175089A1

Make each passage answer a question

During days 31 to 60, make individual passages answerable and discoverable through the different questions surrounding your topic.

1. Write each retrievable claim as a self-contained sentence, paragraph, or short section that restates and answers the query. Several filings describe retrieving a portion of a page rather than the page as a whole. Search snippets are selected using shared terms or distance between numerical representations of the snippet and query meanings.

2. Name subjects, their attributes, and full combinations of specifications in the page text, including its titles. Session rewriting, combining related inquiries, and expanding queries search subject-plus-attribute strings, rather than pronouns or a gist.
3. Put key terms in titles, in bold, high on pages, and near related query terms. Three related-inquiry filings extract suggested search parameters using frequency, location, proximity, font, size, and alignment.
4. Rank for rewritten, expanded, machine-generated, and grouped queries around your topic, alongside the typed query. These machine queries add documents to the summarized set and recover pages sharing no query words.
5. Keep each searchable unit on one topic, separating distinct aspects into passages with distant numerical representations. Representing a page covering multiple topics with a single set of numerical values blurs those topics. The unique topics inferred from an eliminated result can become constraints that exclude results in later searches.
6. Add image captions, alt text, and video transcripts to your pages. Filings name captions, recognized image text, object descriptions, and audio transcripts as possible model-input content.

Filings: US20250217373A1, US12118325B2, US12333436B2, US20250173519A1, US11762848B2, WO2024229375A1, US12242472B2, US20210406260A1, WO2024182453A1, US20240354557A1, US20260072965A1, US11769017B1, US11900068B1

Make your evidence easy to cite

During days 61 to 90, make facts easy to quote, verify, and locate within trustworthy, distinctive pages.

1. Write key facts as short, self-contained, quotable sentences in continuous passages beside likely search snippets, under linkable headings. Restricted text generation copies exact text sequences from fragments beside snippets; two filings render links to verifying portions.
2. Give citable URLs distinctive, stable, natural-language titles, and give long pages their own distinct section headings. DeepMind citation filings pack that title or heading as the identifier the model must reproduce token for token.
3. Earn incoming links and publish under named authors on a reputable, crawlable domain. Update the content whenever its facts change, so the modification date reflects a real content edit. Author, domain, incoming links, and freshness inform trustworthiness and selection measures for the summarized set.
4. Cover adjacent queries, follow-up questions, and queries derived from claims on the same topic, without near-duplicate passages. Documents from related queries can join the set when it provides weak material for an answer. Unsupported statements become new retrieval queries, which add only passages that the system has not shown earlier.
5. Make sentences, list items, and table rows independently scorable, with claims other selected documents can corroborate. Re-ranking results using summaries of extracted text involves scoring sentences joined together for that evaluation. Confidence in a citation rises with the number of sources verifying it and with their trustworthiness.

6. Stop repeating the same inflated patterns of on-page signals across many URLs on your site. Avoid categories covered by rules for generated responses, including medical advice, hate, misinformation, and offensive content. When attempts to exploit retrieval scoring are detected, an outlying information retrieval score is capped at the domain level. Rules in the Sparrow family can filter a draft or lower its ranking before the system displays it.

Filings: CA3245108A1, WO2023175089A1, US12118325B2, EP4466630A1, US11900068B1, US11769017B1, US20250103662A1, US20250156456A1, WO2024073087A1, US20250139105A1, WO2024145537A1, WO2025170579A1, US20260023790A1, US12450464B2, US20240104336A1

Check whether the changes worked

Check rankings, passage accessibility, citation readiness, and selection behavior to assess whether your pages meet the described conditions.

1. Each target URL ranks among conventional top N or top p unpaid results for the main query. It also ranks for at least one rewritten, expanded, machine-generated, or grouped variant of that query. The filings describe packing these search results into the material used to generate answers.
2. Every claim you want cited stands alone in body text, outside ads, images, and comments. Place it under a heading at the page or passage opening; name its subject in title and sentence.
3. Answers appear as extractable HTML text, and the media on the page have captions or transcripts. Each citable passage has a stable URL and an in-page locator. Snippet assembly and quotation shortening therefore have a well-formed span of text available to copy.
4. Visible revision dates match real edits, and unique titles retain exactly the same sequence of tokens. The same claim does not recur as a near-duplicate on other URLs within the domain.
5. When the system shows a listing, users select it rather than skipping it, including from lower positions. Filings describing ranking through pairwise comparisons treat engaged items as positives and skipped items above them as negatives.

Filings: US11769017B1, US12333436B2, WO2024229375A1, US20260064780A1, WO2023154385A1, CA3245108A1, US11900068B1, WO2025183818A1, WO2023175089A1, US20260023790A1, US11989649B2, US20210406680A1