Is It Possible to Build? Universal LLM Optimization Analyzer with Gemini API for Screaming Frog

Metehan Yesilyurt

AI Search & Information Retrieval Researcher · metehan.ai

Published: July 09, 2025 • Canonical: https://metehan.ai/blog/llm-optimization-analyzer-screaming-frog/

The rise of AI-driven search engines like Google's AIO & AI Mode, ChatGPT and Perplexity has created a seismic shift in how content is ranked, surfaced, and evaluated.

In this guide, we'll explore an EXPERIMENTAL custom JavaScript snippet that integrates with Screaming Frog and leverages **Gemini 1.5 Flash API** to evaluate any webpage's LLM-readiness. This script is inspired by research such as:

* "Batched Self-Consistency Improves LLM Relevance Assessment and Ranking" (arXiv, May 2025)[1]

* "C-SEO Bench: Does Conversational SEO Work?" (arXiv, June 2025)[2] We'll walk through how the script works, and how you can apply it to your own SEO audits.

Special thanks to Natzir and Victor Pan.

Important Note: Auditing and optimizing your pages based only on this Custom Javascript may break your current rankings. It's experimental. You can play with weights, customize extractors.

✨ What This Script Does

This JavaScript snippet is designed to run in **Screaming Frog's Custom JS Snippet**, extract and analyze structured content from any crawled page, and pass a comprehensive prompt to **Google's Gemini API** to assess LLM ranking readiness.

What It Evaluates:

- Thematic clarity and extractable target queries
- Paragraphs, headers, lists, and FAQs that are LLM-friendly
- Schema presence and quality
- Passage-level scoring using Gemini
- Optimization suggestions to enhance LLM performance

> ⚠️ ***Note**: While the Gemini API does not natively support batched document ranking alone, the script mimics the effect of **Batched Pointwise (PW)** scoring by passing multiple high-signal content segments (passages) in one structured prompt. This strategy is inspired by the paper's finding that batched relevance*

> *judgments led to a **+7.5% improvement in NDCG@10** over standard pointwise methods. (This improvement is valid for legal case!)*

!llmo analysis 1[3]

What Is Pointwise vs. Batched Pointwise?

- **Pointwise Ranking**: Evaluates one query-document pair at a time. It answers: "How relevant is this document to this query?" This method lacks contextual awareness and often produces noisy or inconsistent results.
- **Batched Pointwise Ranking**: Presents multiple candidate documents for the same query together in a batch, and asks the model to score or rank them comparatively. This contextual grouping allows the model to form a relative scale of relevance across documents, rather than evaluating them in isolation.

> *In the referenced research, Batched PW consistently outperformed traditional Pointwise setups across all tested models (GPT-4o, Claude Sonnet 3, Amazon Nova Pro). It also amplified the benefits of **self-consistency** — where the same prompt is asked multiple times and results are averaged for stability. When combined with batching, this method achieved superior ranking accuracy, especially in **NDCG@10**. traditional Pointwise setups, especially in **NDCG@10** — a metric that rewards the placement of highly relevant documents in the top positions.*

What is NDCG@10?

NDCG (Normalized Discounted Cumulative Gain) is a metric used to measure ranking quality. It considers both relevance and position:

- **Relevance**: How well a document matches a query
- **Discount**: Higher relevance at lower ranks (e.g., 1st or 2nd) is more valuable than at the 10th position

NDCG@10 measures the quality of the top 10 results returned by a model. It is widely used in evaluating information retrieval and search engine ranking systems.

In the *Batched Self-Consistency* paper, the researchers compared one-by-one Pointwise scoring with Batched Pointwise on a **legal search dataset using GPT-4o. They found:**

- **One-by-one PW** improved from **44.9% to 46.8% NDCG@10** with 15 self-consistency calls
- **Batched PW** improved from **43.8% to 51.3% NDCG@10**, a significant **+7.5 percentage point gain**

> ✅ *This shows that batching enables the model to better judge relevance comparatively, leading to more stable, higher-quality rankings and improved performance in LLM-driven retrieval systems. a **+7.5% improvement in NDCG@10**, meaning the LLM was able to rank the most relevant passages significantly better when given context via batching. (It's valid for legal GPT-4o case, read full case please) Get your Gemini API key here. https://aistudio.google.com/apikey[4]*

Key Sections of the Script

1. Content Extraction

We pull from all high-signal sources:

```
TEXT
const h1s = document.querySelectorAll('h1');
const paragraphs = document.querySelectorAll('p');
const lists = document.querySelectorAll('ul, ol');
```

Each passage is evaluated for:

- Word count (MIN_WORDS = 10)
- Maximum length for Gemini token constraints (MAX_CHUNK_LENGTH = 500)
- Position weighting (e.g., title = 2.0, list = 1.1)

2. Schema Detection

```
TEXT
const schemas = document.querySelectorAll('script[type="application/ld+json"]');
```

Pulls in FAQ, Article, Product, HowTo, and LocalBusiness structured data for additional LLM context.

3. Content Type Detection

Automatically classifies the page based on:

- Schema types
- URL structure
- Body keyword signals (e.g., “buy now” = product)

```
TEXT
if (/step \d|tutorial|guide|how to/.test(bodyText)) return 'technical';
```

4. Gemini Prompt Design

Builds a rich prompt with this structure:

```
TEXT
Perform the following analysis:
1. IDENTIFY TARGET QUERIES
2. LLMO SCORING (0-5 scale)
3. PASSAGE-LEVEL ANALYSIS
4. CONTENT GAPS
5. OPTIMIZATION RECOMMENDATIONS
```

This ensures that Gemini scores the **entire page semantically**, not just at document-level.

5. API Request

```
TEXT
xhr.open('POST', `https://generativelanguage.googleapis.com/v1beta/models/gemini-1.5-flash:generate
```

A synchronous call is made to Gemini with the prompt payload, returning JSON. Flash models are better, they have the exact latency we need.

The Output (in Screaming Frog)

The Gemini response is formatted into a structured report:

- Overall LLMO Score (0–5)
- Query coverage (relevance scores per query)
- Content gaps (missing signals LLMs expect)
- Optimization priority (low → critical)

You'll also get:

```
TEXT
=== LLMO ANALYSIS RESULTS ===
• TOP 3 POTENTIAL: Yes
• OPTIMIZATION PRIORITY: High
• CONTENT GAPS: Missing product comparisons, No pricing section
```

!llmo analysis 2[5]

Why Content Optimization Often Fails

Research from the C-SEO Bench paper highlights that **61% of optimized pages show no ranking change** in LLM-generated citation order, especially in retail domains. While traditional content optimization sometimes produces large shifts, the overall average effect is close to zero with high variance.

Breakdown from the study:

- **No ranking change**: 61%
- **Positive change**: 26.2%
- **Negative change**: 12.8%

This supports a broader insight: **Positional placement in the context window** matters far more than minor copy edits.

Positioning Effects in the LLM Context Window

The study revealed that documents appearing earlier in an LLM's context window receive significantly more visibility. Based on citation ranking experiments, here's the boost in relevance by position:

Position	Retail	Games	Books	Web	News	Debate	Average Impact
1	+2.77	+1.89	+1.60	+0.87	+0.70	+1.54	Highest gain
2	+1.78	+1.28	+1.28	+0.19	+0.45	+0.41	Positive
3	+0.67	+0.57	+0.48	-0.22	-0.01	-0.37	Mixed impact
8–10	-0.76	-0.58	-0.88	-1.74	-1.15	-2.14	Negative

> *Conclusion: improving a document's **position in the response window** is far more effective than optimizing text alone.*

Use Cases

- ✅ Optimize content for LLM-driven search like SGE/AIO/ChatGPT/AI Mode
- ✅ Identify passages that LLMs understand best
- ✅ Uncover missing FAQ/schema
- ✅ Score each page's performance across 10+ synthetic queries

⚠️ Considerations and Limitations

While the script implements a research-aligned approach to LLM relevance analysis, several limitations should be considered when interpreting its output:

1) **No Real Batched Prompt Evaluation**:

- The Gemini API used does not natively support multi-document comparative ranking. The script mimics batching by structuring multiple passages in one prompt, but actual ranking consistency may vary.

2) **No True Self-Consistency Calls**:

- The research benchmarks rely on averaging multiple generations (e.g., 15 self-consistency calls), while this script executes a single pass. This limits the stability and reliability of the LLM score.

3) **Token Limit Trade-offs**:

- Due to passage batching and structured prompts, total content length is constrained to Gemini's token budget (\~4096). Long pages may lose valuable context.

4) **Output Variance by Domain**:

- E-commerce and FAQ pages tend to yield more structured, evaluable outputs than editorial or highly visual pages. The Gemini model may underperform on abstract or non-standard layouts.

5) **Heuristic Weighting May Misfire**:

- The weighting logic (e.g., title = 2.0, paragraph = 1.0) is based on SEO intuition, not model-specific feedback. LLMs may prioritize differently.

6) **Scores Are Model-Specific**:

- All findings are tied to Gemini 1.5 Flash. Output quality, relevance assessment, and context understanding will differ across models (Claude, GPT-4o, etc.).

7) **No External Link or Entity Scoring**:

- The model does not analyze external link relevance, citation authority, or entity co-occurrence — key elements in LLM grounding and ranking.

> ✏️ *In short: this tool provides directional LLM readiness scoring, not a definitive ranking verdict. It's best used for optimization guidance, not search outcome prediction.*

Final Thoughts

Important Note: Auditing and optimizing your pages based only on this Custom Javascript may break your current rankings. It's experimental. You can play with weights, customize extractors.

This script isn't just a one-off analysis tool, it's a blueprint for what **SEO in the LLM era** looks like:

- Think passage-first, not just page-first
- Think AI-readability, not just keyword stuffing
- Think structured clarity, not complexity

By combining **Batched Self-Consistency principles** with real LLM APIs like Gemini, you can build your own **LLM Optimization Stack** and future-proof your content strategy.

Get the free Screaming Frog LLMO script here: https://github.com/metehan777/llmo-optimization-screaming-frog[6]

An LLM optimization analyzer in Screaming Frog is the AEO equivalent of a crawler. Without it you cannot measure why AI citations in ChatGPT, Perplexity and Gemini land or miss. Answer engine optimization programs need this in their audit stack.

REFERENCES & SOURCES

[1] *** "Batched Self-Consistency Improves LLM Relevance Assessment and Ranking" (arXiv, May ...** — https://arxiv.org/abs/2505.12570

[2] *** "C-SEO Bench: Does Conversational SEO Work?" (arXiv, June 2025)** — https://arxiv.org/abs/2506.11097v1

[3] **llmo analysis 1** — https://metehan.ai/wp-content/uploads/2025/07/llmo-analysis-1.png

[4] **https://aistudio.google.com/apikey** — https://aistudio.google.com/apikey

[5] **llmo analysis 2** — https://metehan.ai/wp-content/uploads/2025/07/llmo-analysis-2.png

[6] **https://github.com/metehan777/llmo-optimization-screaming-frog** — https://github.com/metehan777/llmo-optimization-screaming-frog