HEY, GOOGLE, DU HAST DA WAS LEAKEN LASSEN!

Google hat interne Dokumentationen leaken lassen. Der größte Google-Leak seit 2019 gibt uns einen tiefen Einblick in die Struktur der Suchmaschine. Dennoch wird gerade viel Halbwissen und Unwahrheit verbreitet sowie interessengeleitete Kommunikation betrieben. Johan von Hülsen zeigt, was wir aus dem Leak lernen können, was wir nicht weiter beachten sollten und wie man aufgrund der Erkenntnisse eigene Analysen angehen sollte.

Johan Hülsen

DER AUTOR

Johan ist Gründer und SEO bei Wingmen Online Marketing. Seit Jahren ist er der Funktionsweise der Suchmaschine auf der Spur und versucht, eine bessere SEO durch ein tieferes Verständnis der Suchmaschine zu ermöglichen.

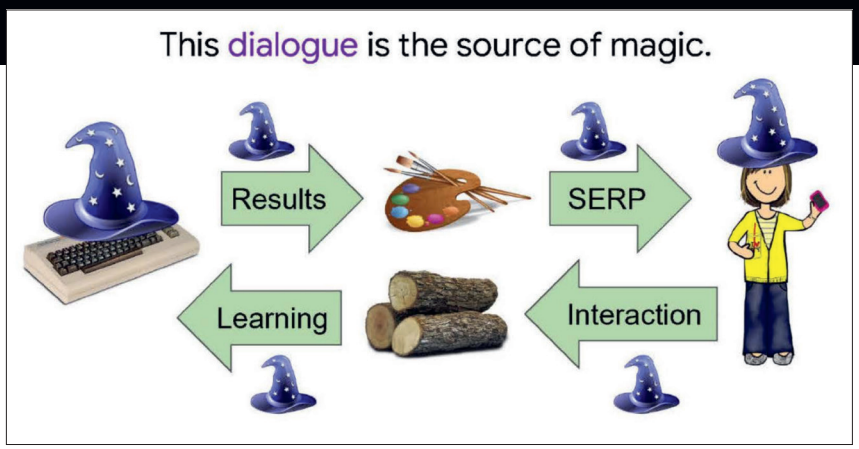


Abb. 1: Dieses Slide aus dem FTC-Verfahren zum Wert der User-Signals ging letztes Jahr durch die SEO-Community.

Trotz seiner Größe gelangen bisher relativ wenig unkontrollierte Informationen aus Google nach außen. Die letzten zwölf Monate waren hier eine Ausnahme. Zum einen gab es das FTC-Verfahren (siehe *einfach.st/pandunayak*), das untersuchte, ob Google durch die Einbindung als Standardsuchmaschine in Browsern unlautere Wettbewerbsvorteile erlangte. Schon der öffentliche Teil der Anhörungen und Beweismittel war sehr aufschlussreich. Die SEO-Community konnte mehr über Navboost (siehe e*infach.st/blind5*), die Verwendung von User-Signals und die Limitierung des Google-Index auf 400 Milliarden Dokumente erfahren (*einfach.st/index8*). Auch interne Kommunikations- und Diskussionsstränge, aus denen unterschiedliche interne Interessenlagen deutlich werden, wurden offengelegt.

Diese Informationen wurden jedoch durch einen technischen Fehler übertroffen, der interne Dokumentationen in einem öffentlichen Github-Repository verfügbar machte. Dieser Leak wurde Rand Fishkin und Michael King zugespielt, die sich entschieden, ihn zu veröffentlichen (siehe *einfach.st/leak9*). Google hat die Authentizität dieser Dokumente bestätigt (*einfach.st/verge4*).

Rand und Mike präsentierten den Leak Mitte 2024 mit einer verständlichen Rechthaberei: „Sieh da! Wir haben es schon immer gesagt und wurden dafür kritisiert." Doch dieses Framing greift zu kurz und ignoriert, wie dieser Leak deine SEO verbessern kann. Es wurde viel über den Leak geschrieben, jedoch oft wenig Hilfreiches. Viele SEO-Experten suchten nach Bestätigungen ihrer Theorien oder Arbeitsweisen, was zu massiven Fehlinterpretationen führte. Kritische und fundierte Auseinandersetzungen sind bisher selten.

Was also enthält der Leak?

Der Leak enthält die ehemals interne Dokumentation zu 2.956 Modulen. Jedes Modul hat eine Kurzbeschreibung und eine Liste der verfügbaren Attribute. **Diese Dokumentation beschreibt nicht direkt Ranking-Faktoren**, sondern die Datenstrukturen, die den Algorithmen zur Berechnung von Ranking-Faktoren zur Verfügung stehen.

> Diese Dokumentation beschreibt nicht direkt Ranking-Faktoren, sondern Datenstrukturen, die den Algorithmen zur Verfügung stehen.

Das ist wie ein großer Wocheneinkauf: Spüli, Toilettenpapier, eine Zeitschrift und jede Menge Lebensmittel. Wir können Vermutungen anstellen, aber wir wissen nicht, welches Gericht am Montag nach welchem Rezept gekocht wird. Ebenso wenig wissen wir, ob vielleicht ein Mitbringsel für die Nachbarin dabei war.

Was steckt denn jetzt im Leak?

Ungefähr 1 % der insgesamt 14.000 Attribute sind bereits als veraltet markiert und sollen nicht mehr verwendet werden. Etwa ein Drittel der Module bezieht sich nicht auf die Google-Suche, sondern auf Google+, YouTube, Maps oder die Nutzerverwaltung. Diese haben also nur indirekt mit den Suchergebnissen zu tun. Die geleakten Informationen sind jedoch sehr aktuell und enthalten Module aus dem Jahr 2023.

Die isolierte Betrachtung eines einzelnen Moduls ist wenig hilfreich, da die meisten Module über ihre strukturierten Eigenschaften auf andere Module verweisen, die wiederum auf weitere Module verweisen. In unserem „Einkauf" befinden sich also 2.596 Module. Ein Modul wird kurz beschrieben. Dann werden die Eigenschaften des Moduls mit ihren Datentypen und gegebenenfalls weiteren Erläuterungen aufgeführt.

Wie sieht ein Modul aus?

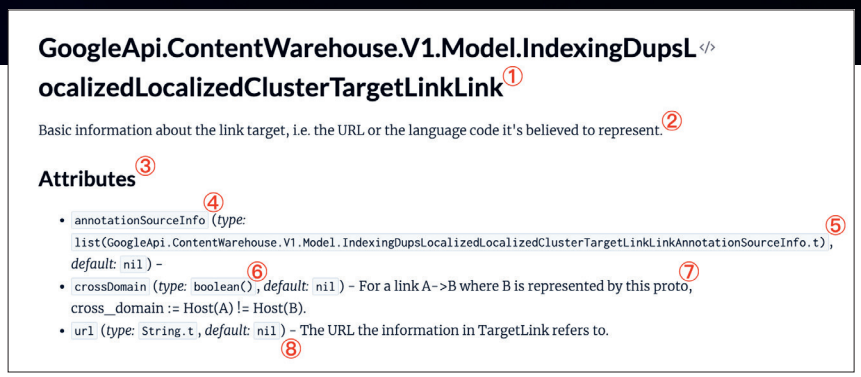
In Abbildung 2 siehst du einen Screenshot des Moduls IndexingDupsLocalizedLocalizedClusterTargetLinkLink. Den ersten Teil des Namens kannst du ignorieren, da er bei allen Modulen identisch ist. Der Name (①) verrät, dass es sich um ein Modul aus dem Bereich Indexierung (Indexing) handelt. Es befasst sich mit Duplikaten (Dups) im Zusammenhang mit Internationalisierung/Lokalisierung (Localized). Es beschreibt den TargetLink-Teil eines LocalizedClusters und die Eigenschaft Link. Der Modulname enthält also schon viele Informationen.

Oft gibt es eine Beschreibung des Moduls (②), die in diesem Fall

vergleichsweise aussagekräftig ist. Manchmal muss man die Beschreibung aus den Verweisen auf ein Modul zusammenbauen, gelegentlich ist die Beschreibung im Verweis aussagekräftiger als im Modul selbst.

Dann werden die Attribute des Moduls aufgelistet (③). Ein Attribut kannst du dir als Eigenschaft des im Titel des Moduls beschriebenen Objekts vorstellen. In diesem Fall sind es die Eigenschaften einer lokalisierten URL innerhalb einer Lokalisierungsgruppe eines Duplikats. Diesen Sachverhalt kennst du, wenn du schon einmal mit hreflang gearbeitet hast. Hier könnte beschrieben sein, dass *https://www.website-boosting.at* eine lokalisierte Variante der Seite *https://www.website-boosting.de* ist und welche Eigenschaften dafür herangezogen werden.

Jedes Attribut hat einen Titel (④) und eine Beschreibung des Datentyps. Im Fall von annotationSourceInfo ist der Datentyp ein Verweis auf ein anderes Modul (⑤) mit eigenen Eigenschaften. Die Eigenschaft crossDomain dagegen ist vom Typ boolean, das bedeutet, sie kann entweder den Wert wahr oder falsch annehmen. In diesem Fall ist das Attribut kommentiert (⑦). Der Kommentar erklärt, dass das Linkziel beschrieben wird und crossDomain wahr ist, wenn der Host (Subdomain + Domain) der verlinkenden URL nicht identisch mit dem Host der verlinkten URL ist. Proto begegnet dir in der Dokumentation häufig und ist ein Kurzwort für Protocol Buffer, eine JSON-ähnliche Datenstruktur, die kürzer und prägnanter ist. Zuletzt (⑧) sehen wir ein weiteres Attribut url, das den Datentyp String hat, also ein einfacher Text.



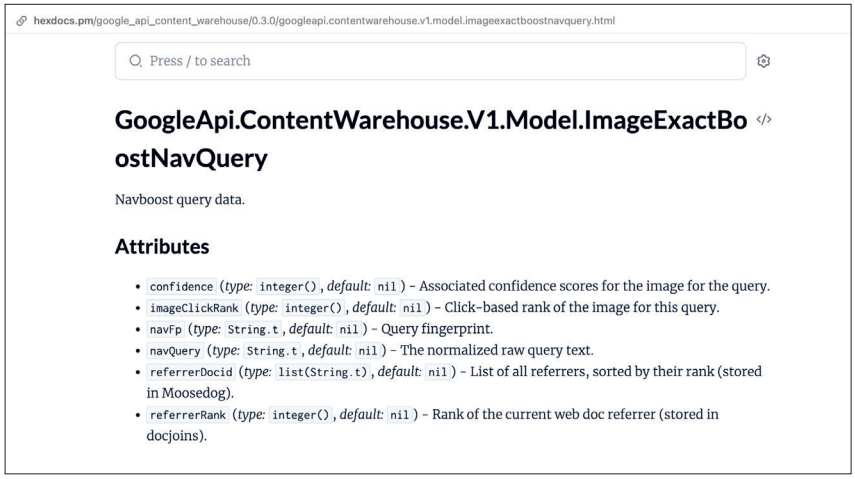
GoogleApi.ContentWarehouse.V1.Model.IndexingDupsLocalizedLocalizedClusterTargetLinkLink ①

Basic information about the link target, i.e. the URL or the language code it's believed to represent. ②

Attributes ③

- `annotationSourceInfo` (*type:* `list(GoogleApi.ContentWarehouse.V1.Model.IndexingDupsLocalizedLocalizedClusterTargetLinkLinkAnnotationSourceInfo.t)`, *default:* `nil`) – ④ ⑤
- `crossDomain` (*type:* `boolean()`, *default:* `nil`) – ⑥ For a link A->B where B is represented by this proto, ⑦ cross_domain := Host(A) != Host(B).
- `url` (*type:* `String.t`, *default:* `nil`) – The URL the information in TargetLink refers to. ⑧

Abb. 2: Screenshot eines einfachen Moduls



hexdocs.pm/google_api_content_warehouse/0.3.0/googleapi.contentwarehouse.v1.model.imageexactboostnavquery.html

Press / to search

GoogleApi.ContentWarehouse.V1.Model.ImageExactBoostNavQuery

Navboost query data.

Attributes

- `confidence` (*type:* `integer()`, *default:* `nil`) – Associated confidence scores for the image for the query.
- `imageClickRank` (*type:* `integer()`, *default:* `nil`) – Click-based rank of the image for this query.
- `navFp` (*type:* `String.t`, *default:* `nil`) – Query fingerprint.
- `navQuery` (*type:* `String.t`, *default:* `nil`) – The normalized raw query text.
- `referrerDocid` (*type:* `list(String.t)`, *default:* `nil`) – List of all referrers, sorted by their rank (stored in Moosedog).
- `referrerRank` (*type:* `integer()`, *default:* `nil`) – Rank of the current web doc referrer (stored in docjoins).

Abb. 3: Screenshot des ImageExactBoostNavQuery-Moduls aus dem Leak

Ein etwas komplexeres Beispiel (mit Navboost)

Ein ebenfalls noch einfaches, in sich geschlossenes Modul ist ImageExactBoostNavQuery. Die Beschreibung ist mit „Navboost query data." sehr kurz. Aber auch längere Beschreibungen sind oft nicht gehaltvoller. Aus dem Titel und der Beschreibung des Moduls lässt sich vermuten, dass hier ein Ranking-Boost für Bilder ermittelt wird, wenn die Suchanfrage exakt zum Bild passt und die Nutzerdaten eine Rolle spielen (Navboost ist ein Algorithmus, der aus dem Nutzerverhalten für das Ranking lernt).

Die einzelnen Attribute geben uns weitere Einblicke:

» **confidence:** Dieser Score bewertet die Verwandtschaft von Suchanfrage und Bild.
» **imageClickRank:** Es gibt die Ranking-Position eines Bilds für die jeweilige Suchanfrage basierend auf vergangenen Klicks an.
» **navFp:** Ein Hash oder Fingerprint der Suchanfrage zur Beschleunigung der Informationsabrufe
» **navQuery:** Der Textinhalt der Suchanfrage in normalisierter Form
» **referrerDocid:** Es listet alle Dokumente auf, die auf dieses Bild verweisen (das Bild eingebunden haben).
» **referrerRank:** Es gibt das Ranking des aktuellen Canonicals zurück.

Hieraus lassen sich einige Informationen zum Zustandekommen von Bild-Rankings ableiten:

1. Wenig überraschend wird berechnet, wie gut ein Bild zur Suchanfrage passt.
2. Die Klickrate (CTR) spielt bei Bildern eine Rolle.
3. Dass das Ranking einer Seite, die das Bild einbindet, das Bild-Ranking beeinflusst, ist ebenfalls nicht überraschend.

Gleichzeitig haben wir jetzt mehr Fragen als Antworten:

1. Wie wird die **confidence** berechnet?
2. Aus welchem Zeitraum und regionalem Zuschnitt werden die Daten für **imageClickRank** herangezogen? Wird zwischen Desktop und Mobile

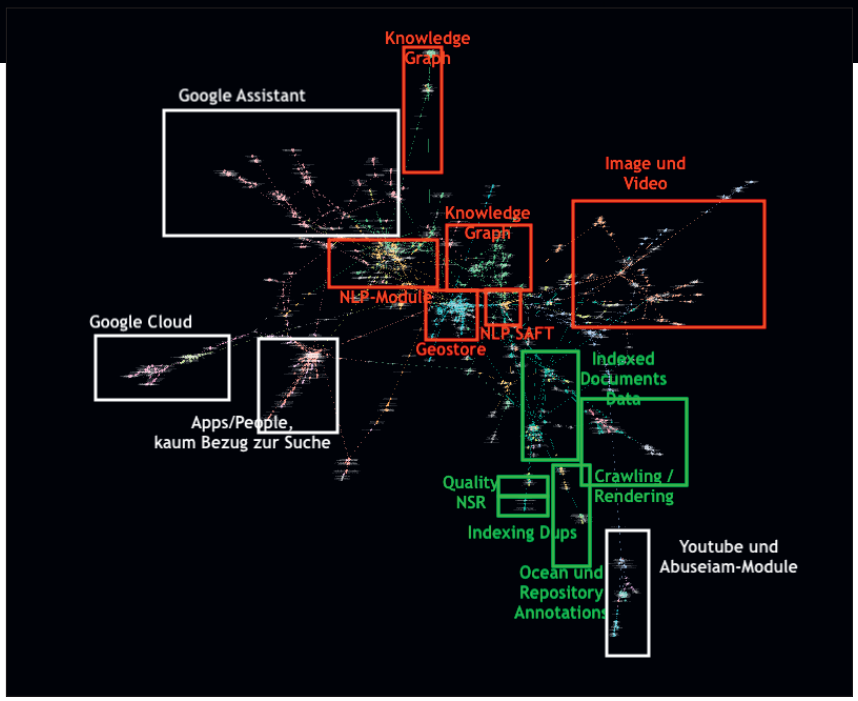


Abb. 4: Visualisierung der Beziehungen zwischen den einzelnen geleakten Modulen

unterschieden? Werden die Daten der letzten 13 Monate oder nur der letzten drei Tage verwendet?

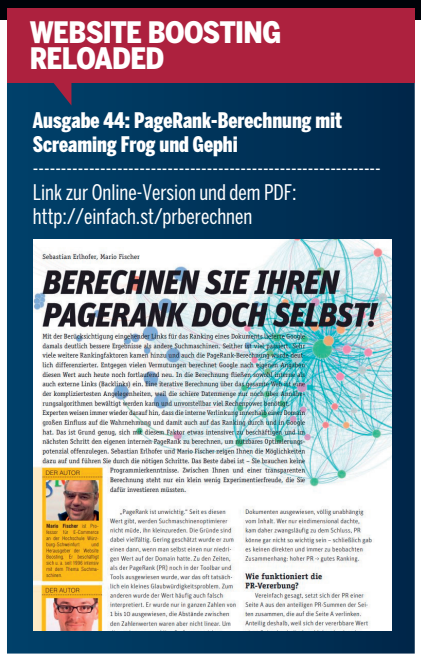
3. Wie wird die Suchanfrage normalisiert? Unterscheidet sich die Normalisierung für die Bildersuche von der organischen Suche oder wird die gleiche Normalisierung verwendet?
4. Was ist **Moosedog** (es gibt nur zwei weitere Module im Leak, die darauf referenzieren)? Moosedog könnte der Image-Index sein.
5. Bezieht sich der Rank des **referrer-Ranks** auf ein allgemeines Ranking des Dokuments oder auf das Ranking für diese spezifische Suchanfrage?
6. Was sind **docjoins**?

Wie können wir aus dem Leak Erkenntnisse ziehen?

Bei der Arbeit mit dem Leak fühlt man sich manchmal wie vor einer Korkwand mit Notizen und farbigen Fäden, die alles miteinander verbinden. Man sieht sich wie in einem Film über einen durchgeknallten Psychopathen oder ein extrem professionelles Ermittlerteam – je nachdem, ob einen gerade ein Funken der Erkenntnis trifft oder man sich immer tiefer in eine Sache eingräbt, ohne zu wissen, ob am Ende eine Erkenntnis auf einen wartet.

Da sich viele Module aufeinander beziehen, bietet sich eine Netzwerkanalyse an. Mit Screaming Frog und Gephi (einem Netzwerkvisualisierungstool, bekannt aus Ausgabe 44 der Website Boosting) lässt sich dies schnell umsetzen.

In der Visualisierung sind die zahlreichen Module zu Google Assistant und Google Cloud sowie zu Apps/People gut zu erkennen. Diese sind zwar dominant, haben aber nur einen indirekten Bezug zur Suchmaschinenoptimierung. Deutlich segmentiert sind auch die Module für Crawling und Indexierung, die enge Bezüge zu Natural-Language-Processing(NLP)-Modulen haben, sowie die Knowledge-Graph- und Geostore-Module. Schon hier zeigt sich, dass der Knowledge Graph nicht nur durch Schema.org-Daten befüllt wird, sondern auch durch Informationen aus unstrukturiertem Text.



Was lernen wir über die Google-Architektur?

Nach dieser ersten Orientierung ist es hilfreich, sich bei der Analyse der Dokumente am Aufbau einer Suchmaschine zu orientieren.

Eine Suchmaschine besteht vereinfacht aus sieben Komponenten:

» **URL-Discovery:** Die Suchmaschine muss URLs finden, um sie crawlen zu können.

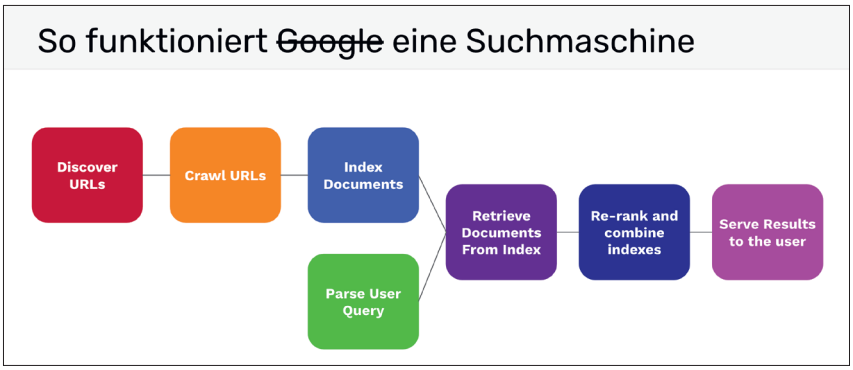


Abb. 5: Das Lieblings-Slide des Autors der letzten drei Jahre illustriert die vereinfachte Prozesskette einer Suchmaschine.

fetched, as well as source IP and ports. It is recommended to use trawler::DestinationIP()/HasDestinationIP() accessors, which return a proper IPAddress.

- `GeoCrawlEgressRegion` (*type:* `String.t`, *default:* `nil`) – If present, the last hop of the fetch was conducted using floonet and this is the location of floonet egress point. It is different from EgressRegion and FlooEgressREgion because it is a Trawler transparent routing configured in the geo crawl rules(go/da-geo-crawl).
- `HSTSInfo` (*type:* `String.t`, *default:* `nil`) – Set to: o

Abb. 6: Googles Leak ist voller Anspielungen. Hier: regionales Crawling mit dem Flohnetzwerk (bekannt aus Harry Potter).

- **Crawling:** Die Suchmaschine muss URLs abrufen, um den Content extrahieren und bewerten zu können. Aus dem Leak wissen wir, dass diese Komponente Trawler genannt wird.
- **Indexing:** Der gecrawlte Content muss bewertet und in den Index geschrieben werden. Diese Komponente heißt Alexandria.
- **Query Processing:** Eine Suchanfrage muss in Einzelbestandteile zerlegt werden.
- **Ranking:** Die verarbeitete und optimierte Query wird an den Index geschickt und die wichtigsten Ergebnisse werden extrahiert.
- **Re-Ranking:** Die Ergebnisse werden noch einmal verfeinert und optimiert.
- **SERP-Presentation:** Die Ergebnisse werden ausgegeben und um vertikale Ergebnisse (Bilder, News, Video, Maps, Werbung …) ergänzt. Hier spielt Mustang eine wichtige Rolle. Diese Komponente wählt nicht nur Bilder aus, sondern übernimmt auch das Snippet-Design.

URL-Discovery

Um überhaupt einen Index aufzubauen, muss eine Suchmaschine URLs finden. Neben der internen und externen Verlinkung sind Sitemaps die wichtigste Quelle für neue URLs. Auch wenn dieses Problem für die meisten Seiten als gelöst betrachtet werden kann, finden wir im Leak einige Informationen, die unser Verständnis erweitern können:

Im Modul *PerDocData*, das wesentliche Informationen über Dokumente bereitstellt, werden mehrere PageRank-Werte thematisiert. Unter anderem wird ein *crawlPagerank* erwähnt: „This field is used internally by the docjoiner to forward the crawl pageranks from original canonicals to canonicals we actually chose; outside sources should not set it, and it should not be present in actual docjoins or the index."

Wir können hier spekulieren, dass die Vermutung vieler SEO-Experten zutreffend ist (oder war), dass es einen separaten PageRank für die Crawling-Priorisierung gibt – unabhängig von der PageRank-Berechnung für das Ranking.

Crawling

Nachdem eine URL entdeckt wurde, wird sie in das Crawling-Backlog geschrieben, aus dem sie priorisiert abgerufen wird. Die technische Umsetzung dieses Prozesses übernimmt das Trawler-Modul. Dies ist jedoch der kleinste und am wenigsten spannende Teil der Prozesskette.

Seit Jahren diskutieren SEO-Experten über das Crawl-Budget. Wesentliche Informationen zum Crawl-Budget finden wir beispielsweise in HtmlrenderWebkitHeadlessProtoStyle. In diesem Modul scheint das Crawl-Budget definiert zu werden. Wir lernen, dass es eine maximale Anzahl an parallelen Requests gibt – und dass ClientTrafficFraction, also der Traffic einer Seite, Impact auf das Crawl-Budget hat.

Auch weitere Fragestellungen, wie die Erfassung von Redirects sowie die Verwendung unterschiedlicher User-Agents und der beim Rendering extrahierten Daten, erhalten neue Grundlagen.

Im Modul *HtmlrenderWebkitHeadlessProtoDocument* erfahren wir, welche Informationen Google beim Rendern sammelt. Neben der Länge, der Breite und allen Elementen fällt auf, dass Google an verschiedenen Stellen den Titel und die URL speichert (mal die aufgerufene URL, mal die URL, die per URL history.push verändert wird). Im Modul *HtmlrenderWebkitHeadlessProtoStyle* wird deutlich, wie viele Informationen Google auch über die Optik einer Seite speichert.

Eines der größten Probleme für Google und Seitenbetreiber ist es, zu ermitteln, wann eine URL neu gecrawlt werden sollte. Das Modul *CrawlerChangerateUrlChange* gibt Aufschluss darüber, welche Informationen in die Priorisierung von URLs für den nächsten Crawl eingehen könnten.

Indexing

Googles Indexierungsprozess (*Alexandria*) wertet die gecrawlten Dokumente aus, extrahiert den Main Content und Supplemental Content, zerlegt den Content in Tokens und fügt die Dokumente zu den Posting Lists hinzu. Eine Posting List enthält alle relevanten Dokumente zu einem Token. Dieser invertierte Index dauert länger als das einfache Schreiben des Dokuments in einen Index, sorgt aber dafür, dass eine Suchanfrage wesentlich schneller ausgeführt werden kann, da nicht erst alle Dokumente durchsucht werden müssen.

Ein guter Ausgangspunkt für die Analyse der Indexierung und der Daten, die zu einem indexierten Dokument zur Verfügung stehen, ist das Modul *per-*

DocData. Als Erstes fällt auf, wie viele Attribute zur Kennzeichnung von Spam vorhanden sind (das Wort „Spam“ kommt in diesem Modul 47-mal vor).

Beim Scrollen durch die Liste fällt auch die Anzahl der Datumsfelder und Freshness-Informationen auf. Hier wird ein *FreshnessTwiddler* beschrieben, der im Rahmen des Re-Rankings neue Dokumente boosten soll. SEO-Experten kennen dieses Konzept bisher abstrakt als „Query deserves Freshness“. Hier bekommen wir jedoch einen kleinen Einblick in die technische Umsetzung dieses Konzepts.

> Es werden verschiedene PageRank- und Indy-Rank-Werte gespeichert.

Außerdem auffällig: Es werden verschiedene PageRank- und IndyRank-Werte gespeichert. Neben dem bereits erwähnten crawlPagerank gibt es noch:

- homepagePagerankNs (PageRank der Homepage)
- ScaledIndyRank (und mehrere experimentelle Werte dazu; IndyRank ist eine Version des PageRank, die weniger anfällig für Manipulation ist)
- pagerank (und mehrere experimentelle Werte dazu)
- toolbarPagerank

Nur die ersten beiden scheinen nicht „deprecated“, also nicht veraltet, zu sein.

Bei der Fülle an Informationen, allein schon im *perDocData*- und *IndexingDocjoinerDataVersion*-Modul, hilft es, sich eine Struktur zu schaffen. Dieser Artikel hat nicht den Anspruch, alle Module und Attribute durchzugehen und zu beschreiben, sondern bietet eine Hilfestellung für eigene weiterführende Analysen. In Bezug auf die Indexierung gibt es verschiedene Faktoren, die besonders interessant sind. Interessierst du dich für eines der Themen, dann gib einfach einen der kursiv geschriebenen Begriffe in die Suche des Leaks, zum Beispiel unter *einfach.st/hexdocs*, ein.

Spam-Identifikation

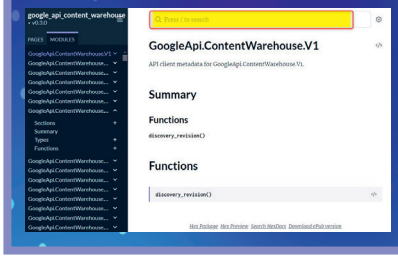
Spam ist ein riesengroßes Thema für Google. Die aktuellen öffentlichen Zahlen aus dem Jahr 2021 zeigen, dass Google über 200 Milliarden Spam-URLs pro Tag aussortiert (*einfach.st/spamreport2021*). Im Leak sehen wir, dass es dabei nicht nur darum geht, Spam-Dokumente gar nicht erst zu indexieren, sondern auch für indexierte Dokumente eine Spam-Wahrscheinlichkeit zu berechnen. Neben den zahlreichen Elementen, in denen Spam direkt benannt wird (glücklicherweise sind diese Elemente durchsuchbar), werden auch Dokumente klassifiziert, die sich an der Grenze zu Spam oder Porn befinden (Modul *QualityFringeFringeQueryPriorPerDocData*). Neben domainweiten Spam- und Porn-Signalen (beispielsweise *SpamBrainData* und *ClassifierPornSiteData*) gibt es auch viele dokumentbezogene Klassifikatoren (*DocLevelSpamScores* oder *ClassifierPornClassifierData*). In Bezug auf Porn ist offensichtlich, dass die Klassifikation im Wesentlichen auf der Erkennung von Bildern beruht (beispielsweise Attribut *imageBasedDetectionDone*).

E-E-A-T und YMYL

Während für die meisten SEO-Experten die Spam- und Porn-Klassifikation weniger relevant ist, sind E-E-A-T (Klassifikation von Content auf Basis von Experience, Expertise, Authoritativeness und Trustworthiness) und YMYL („Your Money or Your Life Query“-Klassifikation, also Suchanfragen, deren Ergebnisse besondere Qualitätsansprüche haben, weil sie Auswirkungen auf die Gesundheit oder den Geldbeutel haben können) von großer Bedeutung und wurden in den letzten Jahren oft diskutiert.

TIPP

Unter einfach.st/hexdocs kann selbst nach Attributen und Modulen gesucht werden. Eine gute Anlaufstelle ist auch 2596.org.



Im Leak sehen wir davon wenig. E-E-A-T kommt im Leak nicht vor. Experience und Expertise werden zwar benannt, allerdings nur im Modul *AppsPeopleOzExternalMergedpeopleapiMapsExtendedData*, in dem Daten über Local Guides gesammelt werden. Auch Trustworthiness wird nicht explizit als Trustworthiness benannt.

> E-E-A-T kommt im Leak nicht vor.

Für Authoritativeness gibt es zwei hervorragende Ausgangspunkte für die weitere Analyse:

- **CompressedQualitySignals:** Dies ist ein Modul, das die Grundlage für das Product-Review-Update zu sein scheint und Rückschlüsse auf die unterschiedlichen Qualitätssignale in Bezug auf Product-Review-Seiten zulässt.
- **IndexingDocjoinerDataVersion:** Dies ist ein Modul, in dem wir lernen, dass Authorship ein Konzept ist, das tatsächlich im Code verankert ist (*qualityAuthorshipAuthorAnnotations*). Auch TopicEmbeddings (also identifizierte Themen) werden zur Autoritätsbewertung herangezogen (*qualityAuthorityTopicEmbeddings*).

Außerdem hat Google für zwei Themen besondere Autoritätssignale: isCovidLocalAuthority und isElectio-

- `secondarySiteChunk` (*type:* `String.t`, *default:* `nil`) – Secondary NSR sitechunk. When present, it provides more granular chunking than primary sitechunks (see quality/nsr/util/sitechunker.h for details).
- `articleScore` (*type:* `number()`, *default:* `nil`) – Score from article classification of the site.
- `versionedData` (*type:* `list(GoogleApi.ContentWarehouse.V1.Model.QualityNsrNSRVersionedData.t)`, *default:* `nil`) – Versioned map of NSR values for experimenting with the next release.
- `siteLinkIn` (*type:* `number()`, *default:* `nil`) – Average value of the site_link_in for pages in the sitechunk.
- `ymylNewsV2Score` (*type:* `number()`, *default:* `nil`) –
- `isElectionAuthority` (*type:* `boolean()`, *default:* `nil`) – Bit to determine whether the site has the election authority signal, as computed by go/election-authority
- `sitePr` (*type:* `number()`, *default:* `nil`) –
- `priorAdjustedNsr` (*type:* `list(GoogleApi.ContentWarehouse.V1.Model.QualityNsrVersionedFloatSignal.t)`, *default:* `nil`) – NSR – prior. Estimate of whether the site is above/below average NSR in its slice.

Abb. 7: Das QualityNsrNsrData-Modul enthält viele Hinweise über Faktoren, die Google bei der qualitativen Bewertung von Seitenbereichen interessieren könnten.

nAuthority aus dem QualityNsrNsrData-Modul zeigen, dass Google bei Wahlen und Covid-Informationen die Autorität einer Seite spezifisch misst. Beide Attribute werden lediglich als Boolean gespeichert. Es gibt also keinen Score, sondern das Programm gibt zurück, ob eine Seite als Autorität gilt oder nicht.

Zu YMYL gibt es einige konkrete Datenpunkte:

» Das perDocData-Modul benennt einen **ymylHealthScore**: Hier wird der Scoring-Wert eines YMYL-Health-Classifiers gespeichert. Dabei handelt es sich vermutlich um einen Wert, der angibt, ob ein Dokument für „Your Money or Your Life"-Querys qualifiziert ist oder nicht.

» Außerdem wird ein **ymylNewsScore** beschrieben, der das Dokument auf Basis des YMYL-News-Classifiers bewertet.

» Das *QualityFringeFringeQueryPriorPerDocData*-Modul enthält außerdem eine YMYL-Einschätzung auf Basis anderer Inhalte der Seite.

» Ebenfalls zu News gibt es eine weitere Erwähnung im *QualityNsrNsrData*-Modul von **ymylNewsV2Score**. Hieraus können wir schließen, dass das YMYL-Signal zu News in zwei Versionen existieren könnte: einmal in Bezug auf das einzelne Dokument und einmal in Bezug auf die Domain oder einen Seitenbereich einer Domain.

Insgesamt lässt der Leak den Schluss zu, dass YMYL und E-E-A-T eher Denkkonzepte und Sammlungen von Signalen sind als fest verankerte Scores.

Helpful Content und Quality

Ebenso wie bei YMYL und E-E-A-T gibt es auch keinen Helpful-Content-Score im Leak. So einfach macht Google SEO-Experten die Analyse des Leaks nicht. Es ist jedoch dokumentiert, dass der Helpful-Content-Classifier (*einfach.st/hcufaq*) auch auf Domainebene arbeitet. In diesem Zusammenhang sind alle Module, die Nsr im Titel enthalten, enorm spannend. Nsr könnte für NewSiteRank stehen. Diese Abkürzung kennen SEO-Experten bereits aus dem Realtime Boost Design Doc des Leaks von 2019 (*einfach.st/leaks1*).

Die *Nsr*-Module arbeiten dabei auf Basis von Seitenbereichen. Google versucht, eine Domain in unterschiedliche (URL-)Bereiche zu zerlegen und diesen Bereichen Eigenschaften zuzuweisen. Teilweise wird von den Seitenbereichen dann wieder auf einzelne Dokumente zurückgeschlossen (*QualityNsrPQData*). Dabei werden neben Linkinformationen auch Informationen über die potenzielle Seitenqualität (beispielsweise *deltaPageQuality*) gespeichert. Bei der Verarbeitung der Seitenbereiche (Chunks) werden Rückschlüsse auf Basis der Artikelqualität gezogen (*articleScoreV2*). Aber auch Abweichungen der Qualität zwischen unterschiedlichen Seitenbereichen werden ermittelt (*siteQualityStddev*). Zusätzlich werden User-Signale (*impressions, chromeInTotal*) erhoben und es wird ermittelt, ob es bei dem Seitenbereich eher um E-Commerce, Video oder User Generated Content geht.

Für die nächste Diskussion über die Anzahl der Werbemittel ist ein Verweis auf den *clutterScore* hilfreich.

Links

Natürlich sind Links für Google weiterhin ein wichtiges Signal. Ausgangspunkt für eine tiefere Analyse ist das *Anchors*-Modul. Hier werden alle Signale zu einer Zielseite gesammelt. Links werden dabei nur zu einem Dokument gespeichert – nicht zu einer URL. Auf den ersten Blick fällt auf, wie viele Linksignale von Google nicht beachtet werden beziehungsweise dass die Anzahl der nicht beachteten Links gespeichert wird.

Die einzelnen Links werden dann im *AnchorsAnchor*-Modul näher beschrieben. Neben dem Datum der ersten Erfassung eines Links wird der Linktext, die Qualität der Linkquelle, gespeichert. Links werden kategorisiert und für die PageRank-Kalkulation gewichtet. Es gibt zusätzliche Signale für Links von Newsseiten.

> Bei den Modulen und Attributen zu Links ist es besonders schwer, zu erkennen, welche vielleicht heute noch aktiv genutzt werden und welche nicht und welche vielleicht nur experimentellen Status hatten.

Weitere wesentliche Erwähnungen neben den PageRank-Attributen und den Anchor-Modulen gibt es im *RepositoryWebrefSimplifiedAnchor*-Modul. Über Penguin-Penaltys ist *IndexingDocjoinerAnchorStatistics* sehr aussagekräftig.

Insgesamt sind die Informationen zu Links und Anchors nicht nur sehr zahlreich, sondern auch über viele verschiedene Module verteilt, die teilweise sehr isoliert voneinander zu existieren scheinen. Das macht es bei Links besonders schwer, zu bewerten, welche Signale jemals über den experimentellen Status hinausgekommen und tatsächlich produktiv genutzt worden sind und welche davon heute noch genutzt werden.

Offensichtlich ist aber, welchen Wert Google Links zur Ermittlung der Qualität beimisst, um gute Dokumente zu erkennen, vor allem aber auch um Spam zu erkennen. Einen Hinweis auf Disavow-Nutzung wird man dagegen vergeblich in den geleakten Dokumenten suchen.

Offensichtlich ist aber, welchen Wert Google Links zur Ermittlung der Qualität beimisst, um gute Dokumente zu erkennen, vor allem aber auch um Spam zu erkennen. Einen Hinweis auf Disavow-Nutzung wird man dagegen vergeblich in den geleakten Dokumenten suchen.

Canonical und hreflang

Neben dem Kampf gegen Spam hat Google ein großes Problem: Der gleiche Inhalt ist auf unterschiedlichen URLs zu finden. Das hat verschiedene Gründe: Duplicate Content durch den Seitenbetreiber, Content-Diebstahl, aber auch legitime Content-Lizensierung und -Übernahme. Dazu kommt auch, dass das Internet im Fluss ist. URLs werden geändert. Manchmal gibt es dabei Redirects (über Status-Code, Meta-Refresh, JavaScript …).

Diese Komplexität bildet Google ab, indem über jedes Dokument ein Verfahren namens Simhash angewendet wird. Damit wird (für den Main Content einer Seite) ein Fingerabdruck (im Leak oft fingerprint/fp oder hash genannt) generiert. Wird dann eine neue URL gecrawlt, wird der Fingerabdruck erneut generiert und geprüft, ob dieser Content bereits bekannt ist oder nicht.

Ist der Simhash-Wert bereits vorhanden, indexiert Google das Dokument nicht neu, sondern fügt die neue URL dem Dokument hinzu. Im Index stehen also nicht URLs, sondern Dokumente und jedes Dokument kann durch mehrere URLs vertreten werden.

Ist der Simhash-Wert bereits vorhanden, indexiert Google das Dokument nicht neu, sondern fügt die neue URL dem Dokument hinzu. Im Index stehen also nicht URLs, sondern Dokumente und jedes Dokument kann durch mehrere URLs vertreten werden.

Zu den zusammengefassten URLs gehören auch Ländervarianten in der gleichen Sprache. Wie Google identifiziert, für welche Länder ein Dokument verfügbar ist, erfährt man im *CountryAttachment*-Modul. In *CompositeDoc* wird ein guter Überblick über die Komplexität gegeben.

In *CompositeDocForwardingDup* werden vermutlich URLs gesammelt, die weiterleiten. Hier kann dann definiert werden, ob alle (oder nur ein Teil der Ranking-Signale dieser URL) an das Dokument vererbt werden sollen. In *CompositeDocExtraDup* dagegen sind Duplikate gelistet, die nicht weiterleiten. Alle verfügbaren Alternate Names werden in *CompositeDocAlternateName* gespeichert. *CompositeDocIndexingInfo* informiert dann, seit wann eine URL nicht mehr die dominierende Canonical-URL ist.

Content und Structured Data

Bei der Indexierung wird das Dokument nicht als Fließtext gespeichert, sondern in Attribute zerlegt. Neben den *QualitySalientTermsSalientTermSet*, die wichtige Begriffe speichern, werden auch Entitäten erfasst. Wer sich für Organic Shopping interessiert, sollte mit dem Modul *QualityShoppingShoppingAttachment* beginnen. Für Content und Structured Data in Verbindung mit Google Discover sind die Module *SocialPersonalizationKnexAnnotation* und *QualitySherlockKnexAnnotation* hilfreich.

Obwohl *RepositoryWebrefWebrefMustangAttachment* und *QualityRankembedMustangMustangRankEmbedInfo* veraltet sind, bieten diese Module eine gute Übersicht darüber, wie Google mit Entitäten arbeitet. Einen noch tieferen Einblick ermöglicht *RepositoryWebrefMention*. Hier wird deutlich, dass Entitäten nicht nur auf das Dokument bezogen, sondern auch im Query-Parsing extrahiert werden.

Nach dem Indexieren der Seite wartet die Suchmaschine auf eine Suchanfrage eines Nutzers. Die Suchanfrage wird dabei in Einzelteile zerlegt und um Synonyme sowie Meta-Daten ergänzt. Meta-Daten können Informationen

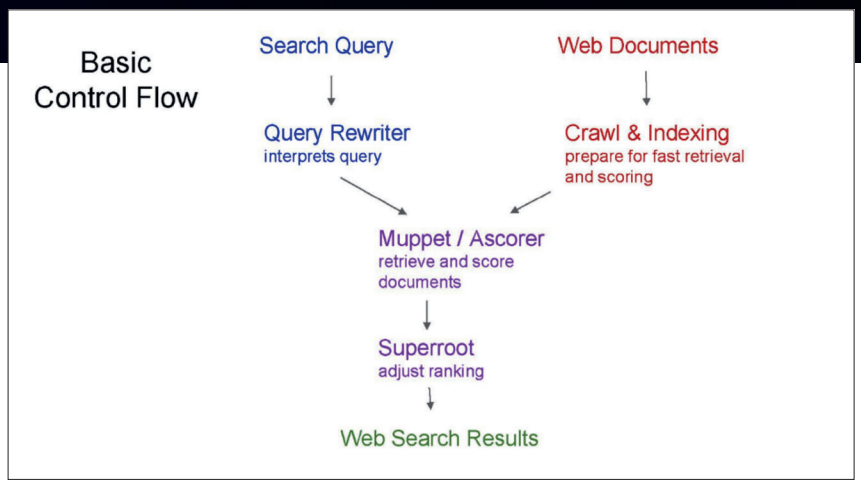


Abb. 8: Im FTC-Verfahren wurde diese interne Darstellung von Google gezeigt, die darlegt, wie das Ranking zusammengestellt wird und welche Systeme daran beteiligt sind.

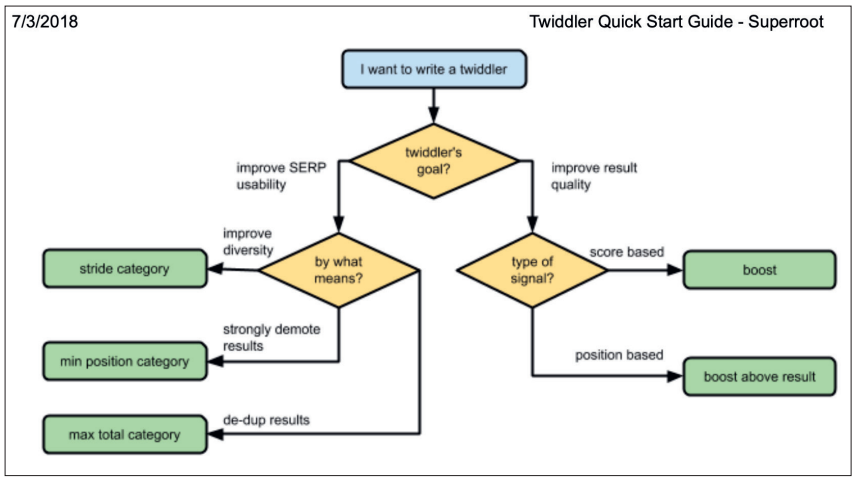


Abb. 9: Der „Twiddler Quick Start Guide“ beschreibt, wie Entwickler eigene Twiddler zur Optimierung der Suchergebnisse schreiben können und welche Gestaltungsmöglichkeiten sie dabei haben.

wie Endgerät, Sprache oder Lokalität umfassen. Diese Meta-Daten sowie die Auswahl und Gewichtung der Synonyme beeinflussen das Ranking. Interessanterweise sind im Leak vergleichsweise wenige Informationen zum Query-Parsing der Google-Suche zu finden.

Ranking

Die Suchanfrage mit ihren Meta-Daten wird im Index abgefragt. Die Ergebnisse werden anhand von Sprache, Lokalität und SafeSearch-Einstellungen gefiltert und für die einzelnen Entitäten und Tokens aus dem Index abgefragt. Im Index liegen die Ergebnisse bereits priorisiert (*selectionTierRank*) und können anhand der indexierten Daten (*compositeDoc*) gerankt werden. Dabei spielt das objektive Mapping zwischen Suchanfrage und Dokument die größte Rolle. Dennoch spielen hier schon *Nsr*-Qualitätseinschätzungen, Spamfilter, PageRank/Indyrank und Nutzerdaten eine Rolle.

Re-Ranking

Im Re-Ranking werden die Ergebnisse in Beziehung zueinander gesetzt, oft durch Programme namens Twiddler. Diese sind nicht Bestandteil der geleakten Dokumente. Wir sehen nur, welche Daten zur Verfügung stehen, nicht ob oder wie sie verwendet werden. SEO-Experten kennen Twiddler aus dem Leak von 2019 (*einfach.st/twiddler4*), wo ihre Funktionsweise und das Re-Ranking beschrieben wurden. Twiddler justieren das Ranking, indem sie nicht nur berücksichtigen, ob ein Dokument zur Suchanfrage passt, sondern auch heranziehen, wie es sich zu anderen relevanten Dokumenten verhält.

Der Leak enthält zahlreiche Verweise auf Daten, die für verschiedene Twiddler verfügbar sind. Einer der wenigen erwähnten Twiddler ist der *SiteboostTwiddler*. Er wird in der Erklärung eines Attributs namens *topPetacatTaxId* erwähnt. Hier speichert Google die Top-Kategorie einer Domain, um die Autorität einer Seite für diese Kategorie im Ranking zu nutzen. Die Kategorisierung erfolgt durch eine Reihe von Modulen namens *FatCat*, die den Content mit einem System namens *Rephil* kategorisieren. Die Kategorisierung einer Domain basiert auf PageRank oder Navboost-Impressions. Ein verwandtes Signal ist *qualityTwiddlerDomainClassification*, das Informationen über den Typ der Domain bereitstellt, sodass Twiddler auch Ratgeber für Produktsuchen berücksichtigen können.

Das Modul *ImageQualitySensitiveMediaOrPeopleEntities* stellt Informationen bereit, mit denen ein Twiddler Bilder bevorzugen soll, die besonders wahrscheinlich zum Dokument gehören. Das Modul *spamtokensContentScore* ermöglicht dem *SiteBoostTwiddler*, nutzergenerierten Spam zu benachteiligen.

Im „Twiddler Quick Start Guide“ hat Google beschrieben, dass es einen Twiddler gibt, der offizielle Seiten für entsprechende Suchanfragen bevorzugt. Im Modul *perDocData* finden wir einen Verweis auf *queriesForWhichOfficial*. Google speichert also zu einem Dokument ab, für welche Suchanfragen diese Seite eine offizielle Ressource ist, und berücksichtigt dabei Sprache und Land, in denen die Suchanfrage gestellt wird.

Viele SEO-Experten vermuten auch, dass das Modul *QualityNavboostCrapsCrapsData* in einen Twiddler eingebun-

den wird. Dieses Modul ist eine zentrale Stelle, um sich über die Signale zu informieren, die in Navboost eingehen. Navboost ist ein System, in dem Google Klick- und Nutzungsdaten nutzt, um sicherzustellen, dass wirklich die besten Ergebnisse ranken.

> Navboost ist ein System, in dem Google Click und Usage-Daten nutzt, um sicherzustellen, dass auch wirklich die besten Ergebnisse ranken.

Serving

Nach dem Re-Ranking werden die Suchergebnisse mit den Ergebnissen der vertikalen Suche (Bilder, Videos …) zusammengefügt und die Snippets für die einzelnen Ergebnisse erstellt. Für weitere Informationen über die Snippet-Generierung sind die folgenden Module gute Startpunkte:

- *SnippetExtraInfo*
- *QualityPreviewRanklabSnippet*
- *QualityPreviewRanklabTitle*
- *RepositoryAnnotationsRdfaRdfa-RichSnippetsApplication* (für Informationen über Rich Snippets)

Doch was tue ich nur?

Diese Einführung zeigt, wie komplex die Analyse des Leaks sein kann. Schon diese kurze erste Übersicht zeigt, warum bei der Anzahl der Module sowie der Komplexität der Attribute niemand bei Google mehr wissen kann, wie „das Ranking" funktioniert. Und verständlicherweise schreckt diese Komplexität ab.

Gleichzeitig ist ein tieferes Verständnis der Funktionsweise von Google sehr hilfreich, um die richtigen Entscheidungen in der täglichen und strategischen SEO-Arbeit zu treffen.

> Ein tieferes Verständnis der Funktionsweise von Google ist enorm hilfreich, um die richtigen Entscheidungen in der täglichen, vor allem aber auch strategischen SEO-Arbeit zu treffen.

Hier kommen drei Tipps für die Auseinandersetzung mit dem Leak:

1. **2596.org**: KI denkt sich Dinge aus, aber für Einsteiger ist 2596.org von Matt Hodson ein guter Startpunkt. Neben den einzelnen Modulen des Leaks listet die Seite KI-Zusammenfassungen der Module und mögliche Interpretationen. Wie bei jedem LLM-Output sollte man dies mit Vorsicht betrachten, aber für eine Erstbewertung ist es oft hilfreich.
2. **Komplexitätsreduktion durch Fokus**: Es hilft, nicht alles auf einmal verstehen zu wollen. Da die Module intensiv miteinander verknüpft sind, ist es ohnehin schwierig, bei einem Thema zu bleiben. Es ist hilfreich, sich einfache Fragen zu stellen. Gute Ausgangsfragen könnten sein: Berücksichtigt Google die Schriftgröße von Text und wenn ja, wo? Wo verwendet Google OCR-Erkennung bei Bildern? Gibt es Hinweise darauf, ob das H1-Tag besonders gewichtet wird? Wie geht Google mit Überschriften um?
3. **Komplexitätsreduktion durch Ignoranz**: Der Leak enthält viele Module, die nichts mit SEO zu tun haben. Daher ist es sinnvoll, als Erstes bei einem Modul zu identifizieren, ob es etwas mit den SEO-Prozessen zu tun hat. Module mit *abuseIam*, *assistant* oder *SocialGraph* im Namen können zunächst vernachlässigt werden.

Dieser Leak ist eine große Bereicherung der SEO-Arbeit. Auch wenn gerade viele nur herauslesen, was ihre Theorie bestätigt, kann dieser Leak ein enormer Qualitätsboost für die SEO-Branche sein.

SEO-Experten können wieder lernen, dass Google im Kern eine Maschine ist – eine komplexe Maschine, aber dennoch eine Maschine. Wir sollten uns abgewöhnen, Googles Suchergebnisse zu anthropomorphisieren.

Gleichzeitig gibt es eine Basis, auf der Gerüchten, Mythen und Halbwahrheiten begegnet werden kann und wir uns um die wichtigen Themen kümmern können.

Und zu guter Letzt zeigt uns der Leak noch mehr Dinge auf, die wir noch nicht wissen. Wir wissen jetzt, dass es ein System namens *Goldmine* gibt, in dem Google unter anderem Entitäten extrahiert. Aber wie funktioniert es? Was macht es noch? Wir wissen, dass Google bis zu 20 Versionen eines Dokuments speichert. Aber werden dabei nur große Änderungen berücksichtigt oder alle? Und wozu wird diese Historie genau verwendet?

> Zu guter Letzt zeigt uns der Leak noch mehr Dinge auf, die wir noch nicht wissen.

Dieser Leak macht Lust auf mehr. Er ist eine enorme Goldgrube für uns SEO-Experten, um als Branche besser zu werden – aber dies gelingt nur, wenn wir unsere Scheuklappen und Vorurteile ablegen und zu einem Teil auch unsere Eigeninteressen. Wer sich mit diesem Leak offen beschäftigt, der kann nur besser in seiner SEO-Tätigkeit werden.

Auf, frisch ans Werk!