How Search Engines Work

A Deep Dive into NLP, Information Retrieval and Ranking Algorithms

From typing a query to ranked results in milliseconds -- **explore the ML pipeline powering modern search.**

Covers: Tokenisation | Inverted Index | TF-IDF | BM25 | BERT Embeddings | PageRank | Vector Search

1. Crawl **2. Index** **3. NLP Parse** **4. Query** **5. Rank**

Each step explored in depth across the following slides

Step 0 | Introduction

What is a Search Engine?

The Library Analogy

Imagine a vast library with billions of books. You ask the librarian: ***"best pizza recipe"***

In milliseconds she returns the most relevant books, ranked by quality, freshness and authority.

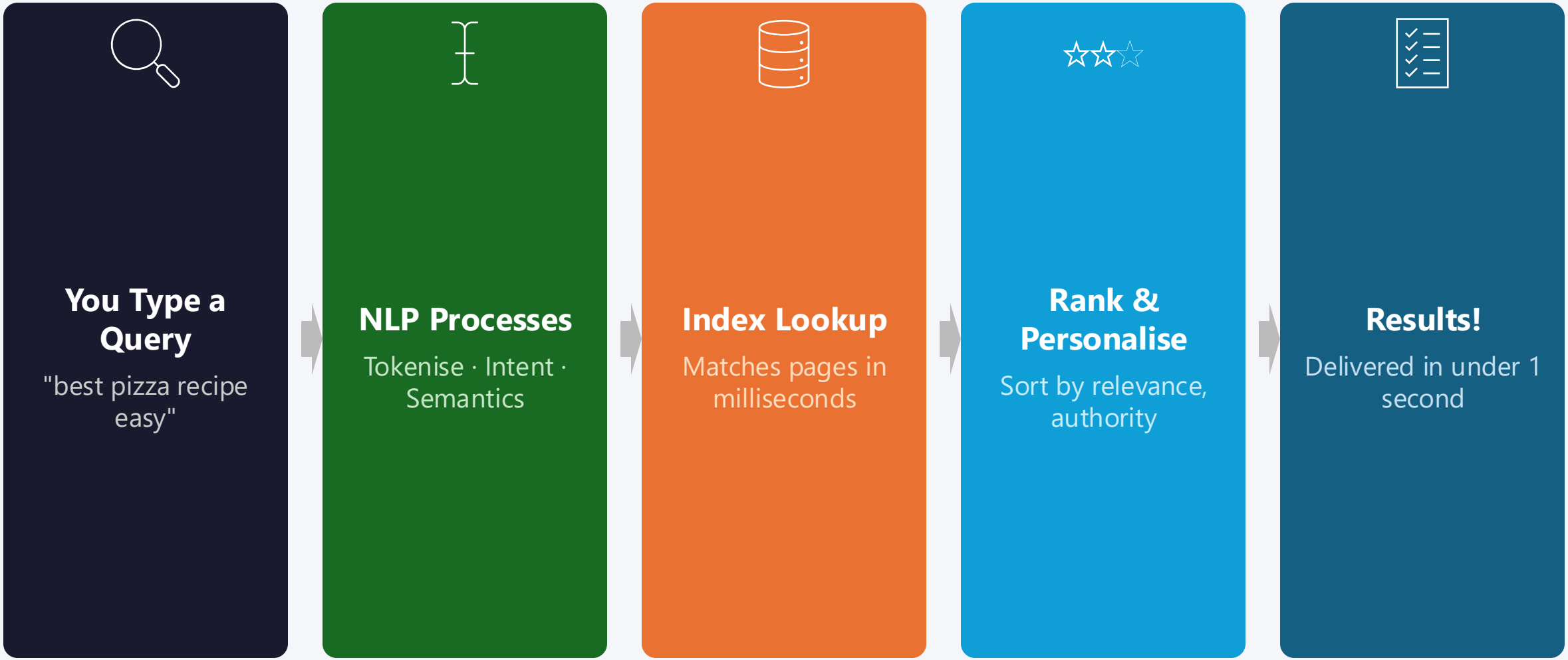
That librarian IS a search engine.
Technically: a distributed system that crawls, indexes, and serves ranked documents against user queries using probabilistic relevance models.

Data Flow — Our Running Example

Query: ***"best pizza recipe easy"***

→ **Step 1:** Tokenise → [best] [pizza] [recipe] [easy]
→ **Step 2:** Remove stop words → [best] [pizza] [recipe] [easy]
→ **Step 3:** Index lookup → match 1,200 docs
→ **Step 4:** Score (TF-IDF + BM25) → rank top 10
→ **Step 5:** Personalise → return results

How It All Connects — The Full Journey



All 5 stages happen in under 1 second every time you press Enter!

Key Takeaways

1

Crawling

Bots discover pages by following links

2

Indexing

Content stored in a giant organised catalogue

3

NLP

AI understands your words, intent and meaning

4

Ranking

200+ signals score every matching page

5

Personalise

Results tailored to you, your location, history

Now you understand what happens every time you hit "Search" — you're no longer a user, you're an expert!

OUR DATA

10 Websites — Our Raw Data

Each website is simplified to one sentence of 5 words for this demonstration.

S1 Pizza dough recipe made easy

S2 The best homemade pizza tips

S3 Buy pizza ovens online today

S4 History of pizza in Italy

S5 Best pizza places in Chicago

S6 Pizza nutrition facts and calories

S7 How to make pizza sauce

S8 The best pizza dough recipe

S9 Easy workouts for busy people

S10 Top ten Italian restaurants nearby

💡 Note: S9 and S10 are off-topic pages — the engine must learn to filter them out during ranking.

STEP 1

Crawling — The Bot Visits Every Page

INPUT
10 URLs to visit
site1.com, site2.com ...

CRAWLER BOT
HTTP GET → Download
Parse HTML → Extract text

OUTPUT
10 raw text documents
stored in the index pipeline

SAMPLE OUTPUT

S1 Pizza dough recipe made easy

S2 The best homemade pizza tips

S5 Best pizza places in Chicago

... and 7 more sentences collected (S3, S4, S6–S10)

Think of it like...
A librarian visiting every bookshop in town and writing down what each book says — that's what a web crawler does with web pages.

STEP 2

Tokenisation — Splitting Text into Words

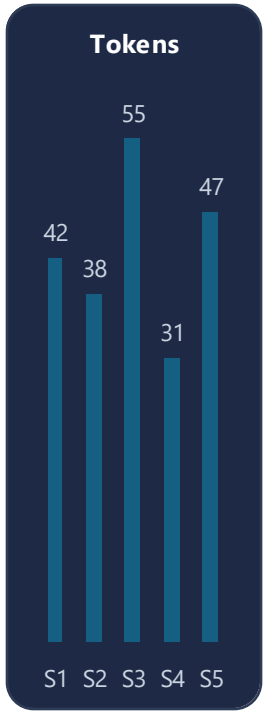
BEFORE (raw sentence)
"Pizza dough recipe made easy"

SPLIT

AFTER (5 individual tokens)
[pizza] [dough] [recipe] [made] [easy]

💡 **Why tokenise?** The engine needs individual words to build a searchable index. Without splitting, it can only match entire sentences.

Site	Word 1	Word 2	Word 3	Word 4	Word 5
S1	Pizza	dough	recipe	made	easy
S2	The	best	homemade	pizza	tips
S3	Easy	chicken	recipe	for	beginners
S4	Running	shoes	review	and	guide
S5	Best	pizza	places	in	Chicago
S6	Healthy	easy	meals	for	kids
S7	Pizza	oven	temperature	and	timing
S8	The	recipe	for	perfect	pasta
S9	Easy	workouts	for	busy	people
S10	Best	restaurants	near	Times	Square



STEP 3

Stop Word Removal — Filtering Noise

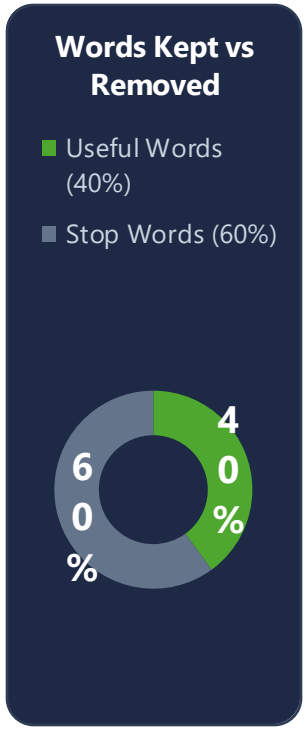
BEFORE (5 tokens)
~~[the]~~ [best] [homemade] [pizza] [tips]

DROP

AFTER (4 meaningful tokens)
[best] [homemade] [pizza] [tips]

💡 **Why remove stop words?** Words like “the”, “and”, “in” appear everywhere and don’t help distinguish one page from another.

Site	Word 1	Word 2	Word 3	Word 4	Word 5
S1	Pizza	dough	recipe	made	easy
S2	~~The X~~	best	homemade	pizza	tips
S3	Easy	chicken	recipe	~~for X~~	beginners
S4	Running	shoes	review	~~and X~~	guide
S5	Best	pizza	places	~~in X~~	Chicago
S6	Healthy	easy	meals	~~for X~~	kids
S7	Pizza	oven	temperature	~~and X~~	timing
S8	~~The X~~	recipe	~~for X~~	perfect	pasta
S9	Easy	workouts	~~for X~~	busy	people
S10	Best	restaurants	near	Times	Square



STEP 4

Stemming — Reducing Words to Roots

BEFORE (varied forms)
recipes, recipe, cooking

STEM

AFTER (common root)
recip, recip, cook → all map to same root

💡 **Why stem?** Searching "recipes" should also find "recipe" — stemming merges all variants so nothing is missed.

Site	Word 1	Word 2	Word 3	Word 4	Word 5
S1	pizza	dough	recip	made	easi
S2	—	best	homemad	pizza	tip
S3	easi	chicken	recip	—	beginn
S4	run	shoe	review	—	guid
S5	best	pizza	place	—	chicago
S6	healthi	easi	meal	—	kid
S7	pizza	oven	temperatur	—	time
S8	—	recip	—	perfect	pasta
S9	easi	workout	—	busi	peopl
S10	best	restaur	near	time	squar

STEP 5

POS Tagging + Named Entity Recognition

Example: **best**ADJ **pizza**NOUN **places**NOUN **Chicago**LOC

● Adjective ● Noun ● Named Entity (Location)

💡 **Why POS + NER?** Knowing "Chicago" is a location (not a verb) lets the engine understand intent and rank location-relevant pages higher.

Site	Word 1	Word 2	Word 3	Word 4	Word 5
S1	pizza NOUN	dough NOUN	recip NOUN	made VERB	easi ADJ
S2	—	best ADJ	homemad ADJ	pizza NOUN	tip NOUN
S3	easi ADJ	chicken NOUN	recip NOUN	—	beginn NOUN
S4	run VERB	shoe NOUN	review NOUN	—	guid NOUN
S5	best ADJ	pizza NOUN	place NOUN	—	**chicago** **GPE**
S6	healthi ADJ	easi ADJ	meal NOUN	—	kid NOUN
S7	pizza NOUN	oven NOUN	temperatur NOUN	—	time NOUN
S8	—	recip NOUN	—	perfect ADJ	pasta NOUN

STEP 6

Inverted Index — The Core Data Structure

TRADITIONAL INDEX
Page → Words (S1 contains: pizza, dough…)

FLIP

INVERTED INDEX
Word → Pages (pizza → S1, S2, S5…)

💡 **Why invert?** Instead of scanning every page for a word, look up the word and instantly get all matching pages — from O(n) to O(1).

Stemmed Word	Found in Sites	Count
pizza	S1, S2, S5, S7	**4**
recip	S1, S3, S8	**3**
easi	S1, S3, S6, S9	**4**
best	S2, S5, S10	**3**
dough	S1	**1**
homemad	S2	**1**
tip	S2	**1**
chicken	S3	**1**
beginn	S3	**1**
run	S4	**1**
shoe	S4	**1**
review	S4	**1**
guid	S4	**1**
place	S5	**1**
chicago	S5	**1**

STEP 7

Query Processing — From Search to Results

🔍 **best pizza recipe**

1. Tokenise
best pizza recipe

2. Stem
best pizza recip

3. Index Lookup
Look up each term

4. Index Lookup Results
"best" → S2, S5, S8 "pizza" → S1, S2, S3, S5, S6, S7, S8 "recip" → S1, S8
Union of all results = {S1, S2, S3, S5, S6, S7, S8, S10} = 7 candidate pages

5. Score with TF-IDF → Rank these 7 pages by how relevant each query word is to that page.
Pages that mention query terms often (TF) and where those terms are rare across all pages (IDF) score highest.

🏆 **Initial TF-IDF Ranking:** S8 > S1 > S2 > S5 > S7 > S3 > S6 > S10
Next step: Re-rank using semantic understanding, authority signals, and personalisation →

STEP 8

TF-IDF Scoring — How Relevance Is Calculated

TF — Term Frequency

How often does the search term appear in this specific page?

Example: “pizza” appears in S1 (1 time out of 5 words) → TF = 0.20

Higher TF = the page talks a lot about this term

IDF — Inverse Document Freq.

How rare is this term across all pages? Rare words are more distinctive.

Example: “recipe” in 2 of 10 pages → IDF = high (rare = valuable)

Higher IDF = the term is more discriminating

TF-IDF Score = TF × IDF (summed across all query terms for each page)

STEP 9

Re-Ranking — Three Signals Beyond Keywords

Semantic Understanding

Does the page actually answer the user's question, or just happen to contain the same words?

Example: *"best pizza recipe" → A page about "pizza dough recipe" matches intent even without the word "best".*

Freshness + Authority

Is this source trustworthy and up-to-date? Measured by PageRank, domain authority, and publish date.

Example: *A recipe from a known food site outranks an old forum post, even with similar keyword matches.*

Personalisation

Does the user's history, location, or device suggest they prefer certain types of results?

Example: *If you often click Italian recipes, the engine boosts Italian pizza results for you.*

Final Score = TF-IDF + Semantic + Authority + Personal

Think of it like... TF-IDF finds books that mention your topic. Re-ranking decides which are worth reading — the well-written, trusted, recent ones rise to the top.

Key Facts: Google uses 200+ ranking signals • PageRank still matters but is just one of many • Modern engines use BERT/LLMs for semantic understanding • Mobile-friendliness and Core Web Vitals also influence rank

FINAL OUTPUT

Final Results Page — What the User Sees

🔍 **best pizza recipe**

#1 S8 The best pizza dough recipe
Highest TF-IDF + semantic match (all 3 query terms)

#2 S1 Pizza dough recipe made easy
Strong keyword overlap: pizza + recipe

#3 S2 The best homemade pizza tips
Semantic match for "best pizza", authority boost

#4 S5 Best pizza places in Chicago
Partial match: best + pizza, but different intent

#5 S7 How to make pizza sauce
Related topic, lower keyword match

#6 S3 Buy pizza ovens online today
Commercial intent ≠ recipe intent

#7 S6 Pizza nutrition facts and calories
Weak relevance to "recipe" query