Information Retrieval II

NLP II 2025

Jakapun Tachaiya (Ph.D.)

Outline

- Evaluation of Search Engine
- Semantic Search
- Deep Learning for Search

How do we know that the documents we retrieve are 'relevant'/'correct'?

Usual evaluation metrics

- precision
- recall
- f1 = 2 * (P + R) / (P * R)

Usual evaluation metrics

- precision
  จำนวนครั้งที่ทายถูก / จำนวนครั้งที่ทาย
  จำนวนเอกสารทายถูกว่าเกี่ยวข้อง / จำนวนเอกสารที่เอามาให้ดู

- recall
  จำนวนครั้งที่ทายถูก / จำนวนคำตอบที่ถูก
  จำนวนเอกสารทายถูกว่าเกี่ยวข้อง / จำนวนเอกสารที่เกี่ยวข้องทั้งหมด

- f1 = 2 * (P + R) / (P * R)

Click data from search log

The data are already 'automatically annotated.'

query	doc id	rank	click?
1	30	1	1
1	12	2	0
1	11	3	1
1	50	4	0
2	12	1	0
2	7	2	0
2	30	3	0
2	4	4	1

Compute Precision and Recall

precision@k = $\frac{\text{จำนวนเอกสารทายถูกว่าเกี่ยวข้อง}}{\text{จำนวนเอกสารที่เอามาให้ดู}}$

recall@k = $\frac{\text{จำนวนเอกสารทายถูกว่าเกี่ยวข้อง}}{\text{จำนวนเอกสารที่เกี่ยวข้อง}\textbf{ทั้งหมด}}$

query	doc id	rank	click?
1	30	1	1
1	12	2	0
1	11	3	1
1	50	4	0
2	12	1	0
2	7	2	0
2	30	3	0
2	4	4	1

Compute precision and recall

query	doc id	rank	click?
1	30	1	1
1	12	2	0
1	11	3	1
1	50	4	0
2	12	1	0
2	7	2	0
2	30	3	0
2	4	4	1

Precion@2 = 1/4, Recall@2 = 1/3 (สมมติมี document แค่ในตารางนี้)

Precision@4 = ?? , Recall@4 = ??

Compute precision and recall

query	doc id	rank	click?
1	30	1	1
1	12	2	0
1	11	3	1
1	50	4	0
2	12	1	0
2	7	2	0
2	30	3	0
2	4	4	1

Precision@4 = 3/8, Recall@4 = 3/3

precision@4 and recall@4 are the same on both tables??

query	doc id	rank	click?
1	30	1	1
1	12	2	0
1	11	3	1
1	50	4	0
2	12	1	0
2	7	2	0
2	30	3	0
2	4	4	1

query	doc id	rank	click?
1	30	1	1
1	11	2	1
1	12	3	0
1	50	4	0
2	4	1	1
2	7	2	0
2	30	3	0
2	12	4	0

Yes >> precision@k and recall@k can't differentiate this case!!!

Precision Recall

- Precision and recall are *acceptable* evaluation metrics but they **cannot measure the quality of ranking.**
- They don't take into account the actual level of relevance (e.g. they click and stay for longer)

NDCG - Metric for ranking

NDCG is a measure of the effectiveness of a ranking system, taking into account the **position of relevant items** in the ranked list.

- Items that are **higher in the ranking should be given more credit** than items that are lower in the ranking.

Rank	Judgment (Gain)	Discounted Gain	Discounted Cumulative Gain (DCG)	Ideal Discounted Gain	Ideal Discounted Cumulative Gain (iDCG)	Normalized Discounted Cumulative Gain (NDCG)
1	2	2/1	2.0	3/1	3.0	0.67
2	0	0/2	2.0	2/2	4.0	0.5
3	3	3/3	3.0	2/3	4.67	0.64
4	2	2/4	3.5	0/4	4.67	0.75

NDCG - Metric for ranking

Rank	Judgment (Gain)	Judgment (Boolean)
1	2	1
2	0	0
3	3	1
4	2	1

Judgment (Gain)

- The **relevance score** assigned to the document. Higher values indicate higher relevance

Item	Interaction
1	Viewed
2	Viewed
3	Clicked
4	Shared
5	Viewed
6	Viewed
7	Ordered
8	Added-to-cart
9	Viewed
10	Viewed

Interaction	Relevance Score
Ordered	4
Added-to-cart	3
Shared	2
Clicked	1
Viewed	0

NDCG - Metric for ranking

Rank	Judgment (Gain)	Discounted Gain	Discounted Cumulative Gain (DCG)	Ideal Discounted Gain	Ideal Discounted Cumulative Gain (iDCG)	Normalized Discounted Cumulative Gain (NDCG)
1	2	2/1	2.0	3/1	3.0	0.67
2	0	0/2	2.0	2/2	4.0	0.5
3	3	3/3	3.0	2/3	4.67	0.64
4	2	2/4	3.5	0/4	4.67	0.75

- **Discounted Gain:**
  - Formula:

$$\frac{G_r}{r}$$

  - The gain is **divided by its rank** to reduce the impact of lower-ranked results.

- **Discounted Cumulative Gain (DCG):**
  - Formula:

$$\sum_{i=1}^{r} \frac{G_i}{i}$$

  - The sum of discounted gains **up to rank** r.

NDCG - Metric for ranking

Rank	Judgment (Gain)	Discounted Gain	Discounted Cumulative Gain (DCG)	Ideal Discounted Gain	Ideal Discounted Cumulative Gain (iDCG)	Normalized Discounted Cumulative Gain (NDCG)
1	2	2/1	2.0	3/1	3.0	0.67
2	0	0/2	2.0	2/2	4.0	0.5
3	3	3/3	3.0	2/3	4.67	0.64
4	2	2/4	3.5	0/4	4.67	**0.75 (NDCG@4)**

- **Ideal Discounted Gain:**
  - Formula:

$$\frac{G_r^*}{r}$$

  - Similar to discounted gain, but for an **ideal ranking** where the most relevant documents are at the top.

- **Normalized Discounted Cumulative Gain (NDCG):**
  - Formula:

$$\frac{DCG}{iDCG}$$

  - A **normalized score** between 0 and 1, showing how close the ranking is to the ideal ordering.

Then, What are NDCG@4 score for this??

query	doc id	rank	click?
1	30	1	1
1	11	2	1
1	12	3	0
1	50	4	0
2	4	1	1
2	7	2	0
2	30	3	0
2	12	4	0

Then, What are NDCG@4 score for the ??

query	doc id	rank	click?
1	30	1	1
1	11	2	1
1	12	3	0
1	50	4	0
2	4	1	1
2	7	2	0
2	30	3	0
2	12	4	0

NDCG@4 = 1

Mean Reciprocal Rank - Metric for ranking

- MRR is calculated by taking the reciprocal of the rank of the **first relevant document for each query, and then taking the mean of these values over** all queries.
  - if the first relevant document is ranked first, the reciprocal rank is 1.
  - If the first relevant document is ranked fifth, the reciprocal rank is 1/5 or 0.2.
- The MRR is then the average of these reciprocal ranks over all queries.

Then, What are MRR score for these 2 ??

query	doc id	rank	click?
1	30	1	1
1	12	2	0
1	11	3	1
1	50	4	0
2	12	1	0
2	7	2	0
2	30	3	0
2	4	4	1

query	doc id	rank	click?
1	30	1	1
1	11	2	1
1	12	3	0
1	50	4	0
2	4	1	1
2	7	2	0
2	30	3	0
2	12	4	0

Mean Reciprocal Rank (MRR) Results:

Table	MRR
Original Table (Left)	0.625
Re-Ranked Table (Right)	1.0

Limitations of click data

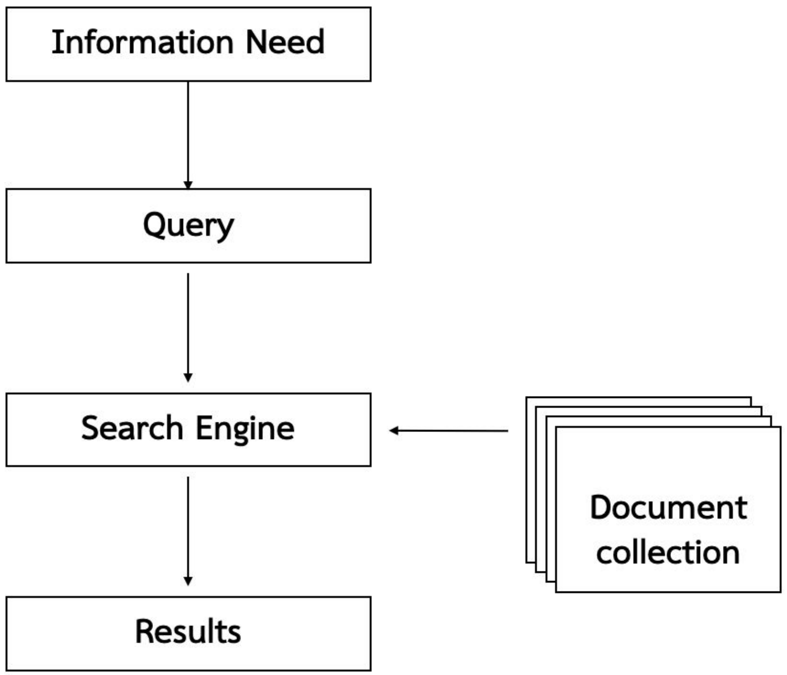
- ถ้า doc ที่ดีกว่านี้มันไม่อยู่ใน search results แล้วทำไง
  - cold start
- คลิกเยอะแล้วดีจริงเหรอ
- คลิกน้อยแล้วแย่จริงเหรอ - หาเจอเลย

Classic search model

- Information need: want to find a restaurant nearby
- Query: search with Wongnai
- Result: click at restaurant page

Does clicking mean visiting a restaurant?

What does intrinsic evaluation actually evaluate?



Which is the better metric?

Clickthrough Rate
(CTR)

อัตราการสั่งอาหาร
อัตราการจองโรงแรม
อัตราการสั่งซื้อสินค้า

A/B testing (Online testing)

- A/B testing is a method of **comparing** the performance of **two or more versions of a search engine**, typically with the goal of improving the user's experience.
- In an A/B test, **users are randomly** assigned to one of two or more groups, each of which is **shown a different version** of the search engine. The performance of each version is then compared based on metrics such as click-through rate, conversion rate, or time spent on the site.
- A/B testing can be used to evaluate all aspects of the search engine, such as the ranking algorithm, the search results page layout, word segmentation, or the wording of the search query box.

Steps to A/B Testing

- Evaluate the new system(s) on click data
- Divert small amount of traffic to the new system e.g.
  - 1% new system
  - 99% status quo
- Wait and monitor
- Roll out to more

Wait and monitor

- Compute the metrics that we want
- We need to run the experiment for a long time
    - The traffic is small
    - The difference is usually small e.g. order rate from 0.50% to 0.51% might translate to thousands per year.
- How do we know the systems are significantly different?

A/B Testing

Pros

- It measures if we address the information need directly.
- We can use just any metrics that matter to us.

Cons

- It takes a lot of time.
- It requires a good infrastructure to redirect users.

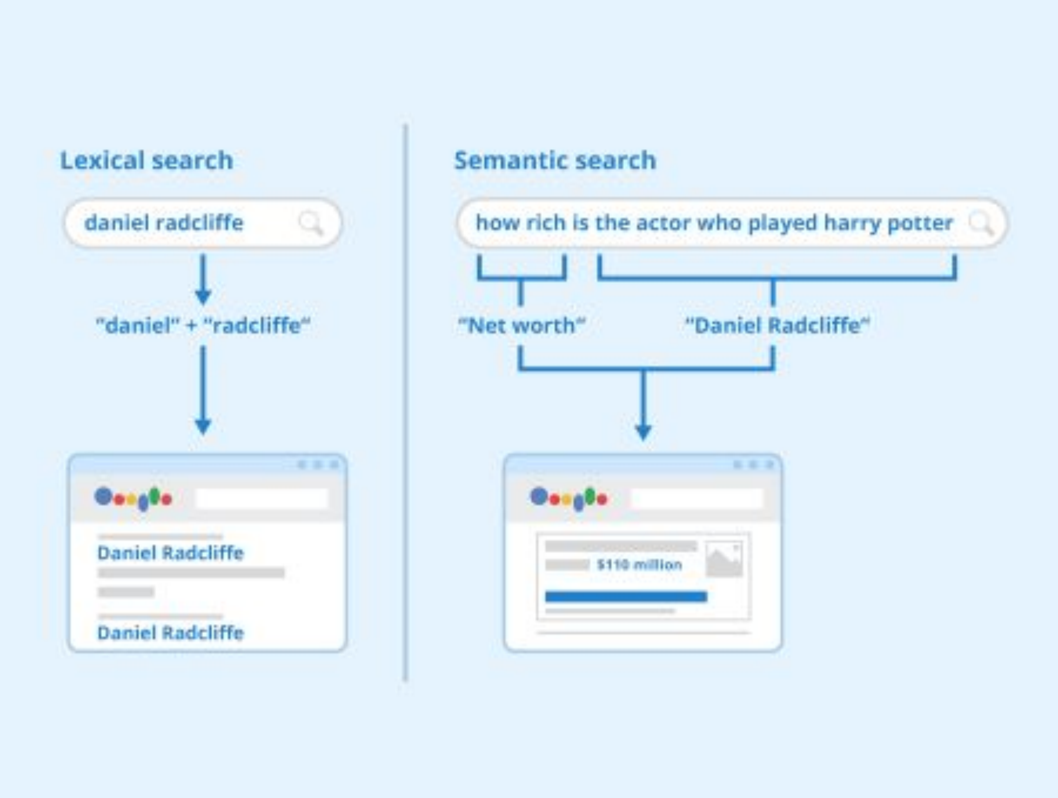
Semantic Search

Consider these search queries

We can use many different phrases to refer to the same thing!

- รับสมัครครูมัธยม
- รับสมัครครูม.ปลาย
- Lehrer in Berlin
- Lehrerin in Deutschland

Lexical Search VS Sematic Search



https://www.seobility.net/en/wiki/Semantic_Search

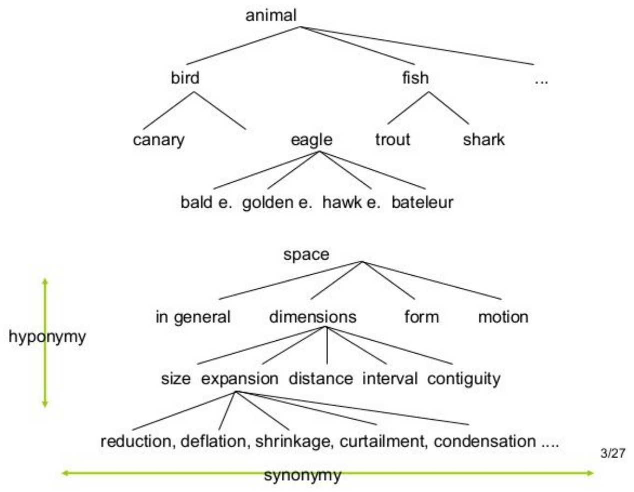
Query Expansion

A technique used in modern search engines to improve the quality and relevance of search results. Query expansion is the process of reformulating a user's query by **adding** or suggesting **additional terms that are related** to the original query terms.

Use lexical semantics to expand queries with WordNet

Token	Lexical relation	Terms
รับ	-	รับ
สมัคร	antonymy	สมัคร, จ้าง
อาจารย์	synonym	อาจารย์, ครู
โรงเรียน	hypernym	โรงเรียน, สถานศึกษา
มัธยม	-	มัธยม

Wordnet



Computational Lexical Semantics with Word-Embedding

```
>>> w2v_model.most_similar('อาจารย์')
[('คณาจารย์', 0.5376085042953491),
 ('ลูกศิษย์', 0.4775567650794983),
 ('ครู', 0.4513567388057709),
 ('นักศึกษา', 0.44001448154449463),
 ('ศาสตราจารย์', 0.4223988950252533),
 ('ศิษย์เก่า', 0.4189813733100891),
 ('อาจารย์พิเศษ', 0.4124056398868561),
 ('ศิษย์', 0.40856611728668213),
 ('นักเรียน', 0.40179842710494995),
 ('รองศาสตราจารย์', 0.3998578190803528)]
```

Contextual Expansion on phrases with Embeddings

Instead of just synonyms, NLP models can find **contextually similar words**:

- Train a **Word2Vec or FastText model** on a corpus to find similar terms.
- Use **BERT embeddings** to find words used in similar contexts.

Example:

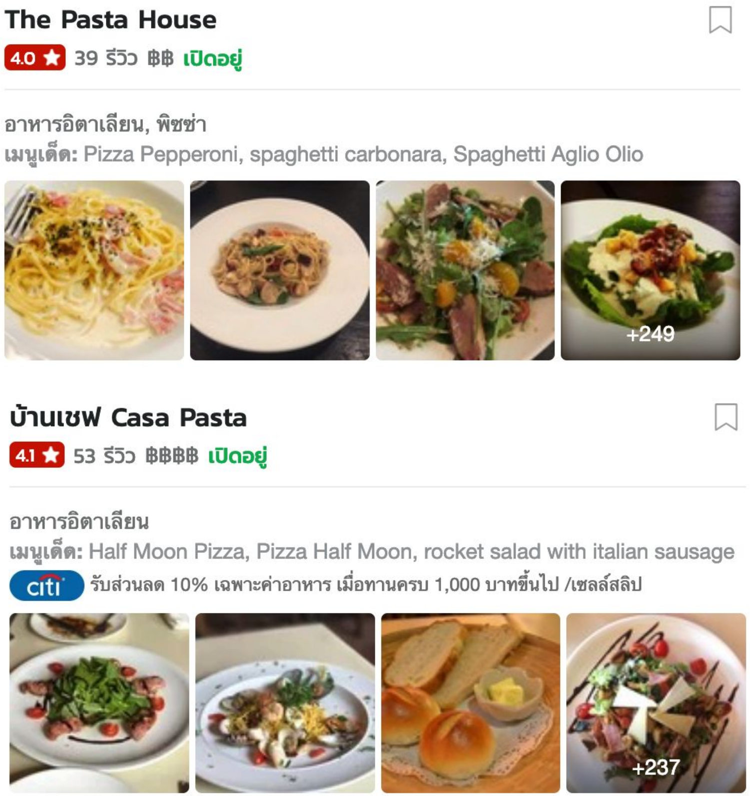
- Input: "*artificial intelligence*"
- Expansion: *["machine learning", "neural networks", "deep learning"]*

Steps for Keyword Expansion

1. **Preprocess the Seed Keyword**
   a. **Tokenization**: Split text into individual words or phrases.
   b. **Lemmatization/Stemming**: Convert words to their root form (e.g., "running" → "run").
   c. **Stopword Removal**: Eliminate common words (e.g., "the," "and," "is") that do not add meaning.
2. **Synonym & Lexical Expansion**
   a. Use **WordNet** or **thesaurus-based** methods to find synonyms and related terms.
   b. Identify **hypernyms** (broader terms) and **hyponyms** (narrower terms).
3. **Contextual Expansion with Word Embeddings**
   a. Use **Word2Vec, GloVe, or FastText** to find semantically similar words.
   b. Apply **BERT or Transformer-based embeddings** for context-aware expansion.
4. **Retrieve Most Similar Candidates**
   a. Compute cosine similarity between word embeddings.
   b. Identify words with high similarity scores based on corpus-specific data.
5. **Manual Review & Refinement**
   a. Remove **ambiguous or irrelevant** terms that do not align with the intent.
   b. Group keywords by **search intent** (informational, transactional, navigational).

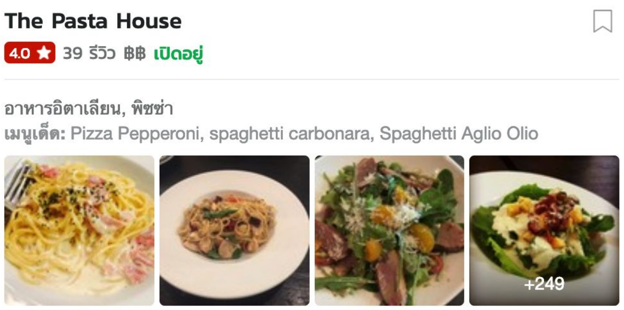
Query understanding

พาสต้า โรแมนติก สีลม ไม่แพง



Query tagging

พาสต้า โรแมนติก สีลม ไม่ แพง

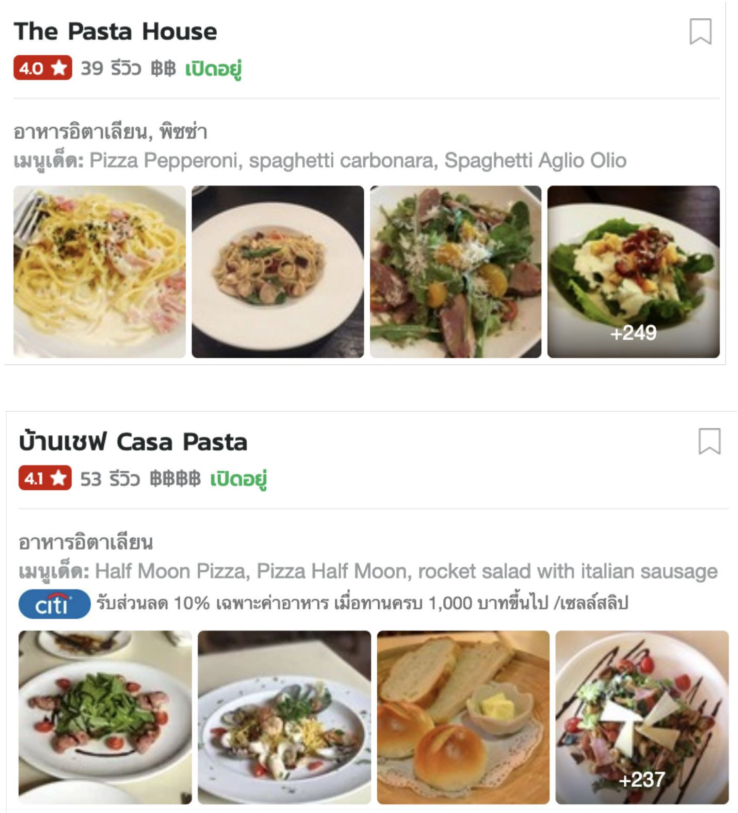


พาสต้า Category:Italian โรแมนติก Location: สีลม Attribute: ฿฿	Category:Italian Location: สีลม Attribute: ฿฿	Category:Italian Location: อุดมสุข Attribute: ฿฿฿฿

1. Querying expansion & tagging
2. Convert to vector space
3. Match with unsupervised methods (TF-IDF)

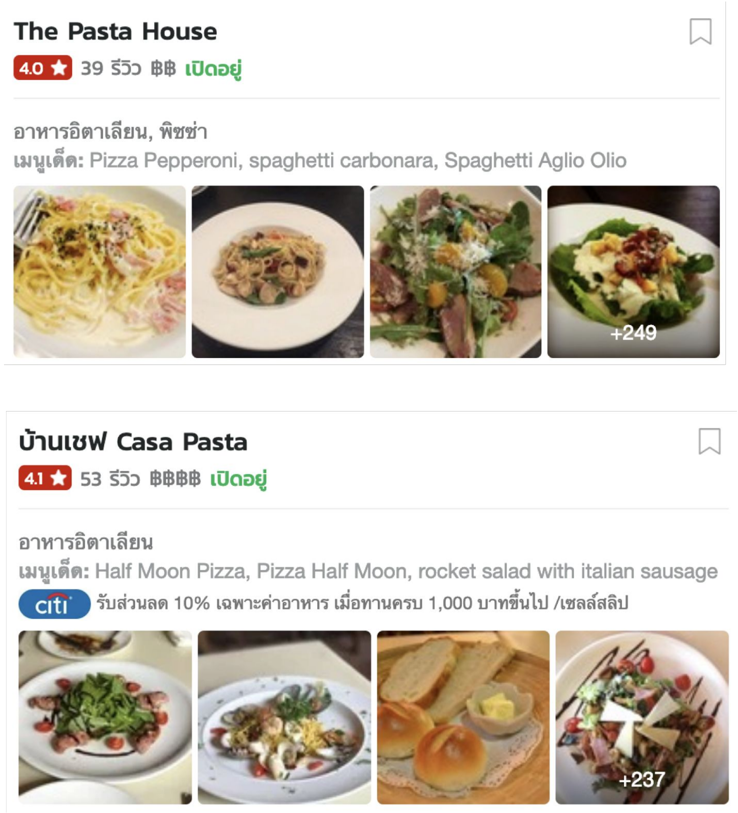
Users don't tell you everything

- Distance
- Star ratings
- Open now?



Users don't tell you everything

- Distance
- Star ratings
- Open now?
- Recency
- New restaurant?



Feature engineering for search

TermScore(q, d) - โดยใช้ TF-IDF

QueryExpansionScore(q, d) - จากวิธีการ keyword expansion

Rating score(q, d)

…

Distance score(u, d)

Recency score(u, d)

feature
0.8
0.7
3.4
…
…
0.7
2.7

Types of features

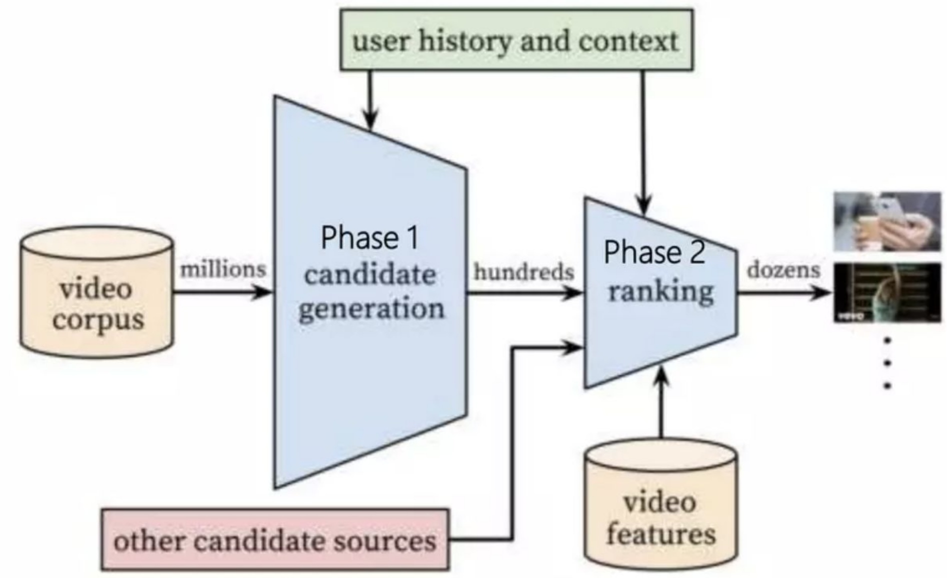
- Query independent features i.e. computing from document only
- Query dependent features i.e. computing from q and d

ID	Feature Description	Category
1	$\sum_{q_i \in q \cap d} c(q_i, d)$ in body	Q-D
2	$\sum_{q_i \in q \cap d} c(q_i, d)$ in anchor	Q-D
3	$\sum_{q_i \in q \cap d} c(q_i, d)$ in title	Q-D
4	$\sum_{q_i \in q \cap d} c(q_i, d)$ in URL	Q-D
5	$\sum_{q_i \in q \cap d} c(q_i, d)$ in whole document	Q-D
6	$\sum_{q_i \in q} idf(q_i)$ in body	Q
7	$\sum_{q_i \in q} idf(q_i)$ in anchor	Q
8	$\sum_{q_i \in q} idf(q_i)$ in title	Q
9	$\sum_{q_i \in q} idf(q_i)$ in URL	Q
10	$\sum_{q_i \in q} idf(q_i)$ in whole document	Q
11	$\sum_{q_i \in q \cap d} c(q_i, d) \cdot idf(q_i)$ in body	Q-D
12	$\sum_{q_i \in q \cap d} c(q_i, d) \cdot idf(q_i)$ in anchor	Q-D
13	$\sum_{q_i \in q \cap d} c(q_i, d) \cdot idf(q_i)$ in title	Q-D
14	$\sum_{q_i \in q \cap d} c(q_i, d) \cdot idf(q_i)$ in URL	Q-D
15	$\sum_{q_i \in q \cap d} c(q_i, d) \cdot idf(q_i)$ in whole document	Q-D
16	$\|d\|$ of body	D
17	$\|d\|$ of anchor	D
18	$\|d\|$ of title	D
19	$\|d\|$ of URL	D
20	$\|d\|$ of whole document	D
21	BM25 of body	Q-D
22	BM25 of anchor	Q-D
23	BM25 of title	Q-D
24	BM25 of URL	Q-D
25	BM25 of whole document	Q-D
26	LMIR.ABS of body	Q-D
27	LMIR.ABS of anchor	Q-D
28	LMIR.ABS of title	Q-D
29	LMIR.ABS of URL	Q-D
30	LMIR.ABS of whole document	Q-D
31	LMIR.DIR of body	Q-D
32	LMIR.DIR of anchor	Q-D
33	LMIR.DIR of title	Q-D
34	LMIR.DIR of URL	Q-D
35	LMIR.DIR of whole document	Q-D
36	LMIR.JM of body	Q-D
37	LMIR.JM of anchor	Q-D
38	LMIR.JM of title	Q-D
39	LMIR.JM of URL	Q-D
40	LMIR.JM of whole document	Q-D
41	Sitemap based term propagation	Q-D
42	Sitemap based score propagation	Q-D
43	Hyperlink based score propagation: weighted in-link	Q-D
44	Hyperlink based score propagation: weighted out-link	Q-D

45	Hyperlink based score propagation: uniform out-link	Q-D
46	Hyperlink based propagation: weighted in-link	Q-D
47	Hyperlink based feature propagation: weighted out-link	Q-D
48	Hyperlink based feature propagation: uniform out-link	Q-D
49	HITS authority	Q-D
50	HITS hub	Q-D
51	PageRank	D
52	HostRank	D
53	Topical PageRank	Q-D
54	Topical HITS authority	Q-D
55	Topical HITS hub	Q-D
56	Inlink number	D
57	Outlink number	D
58	Number of slash in URL	D
59	Length of URL	D
60	Number of child page	D

Learning to rank

- Use TF-IDF to narrow down the candidates
- Rank the candidates using other sources



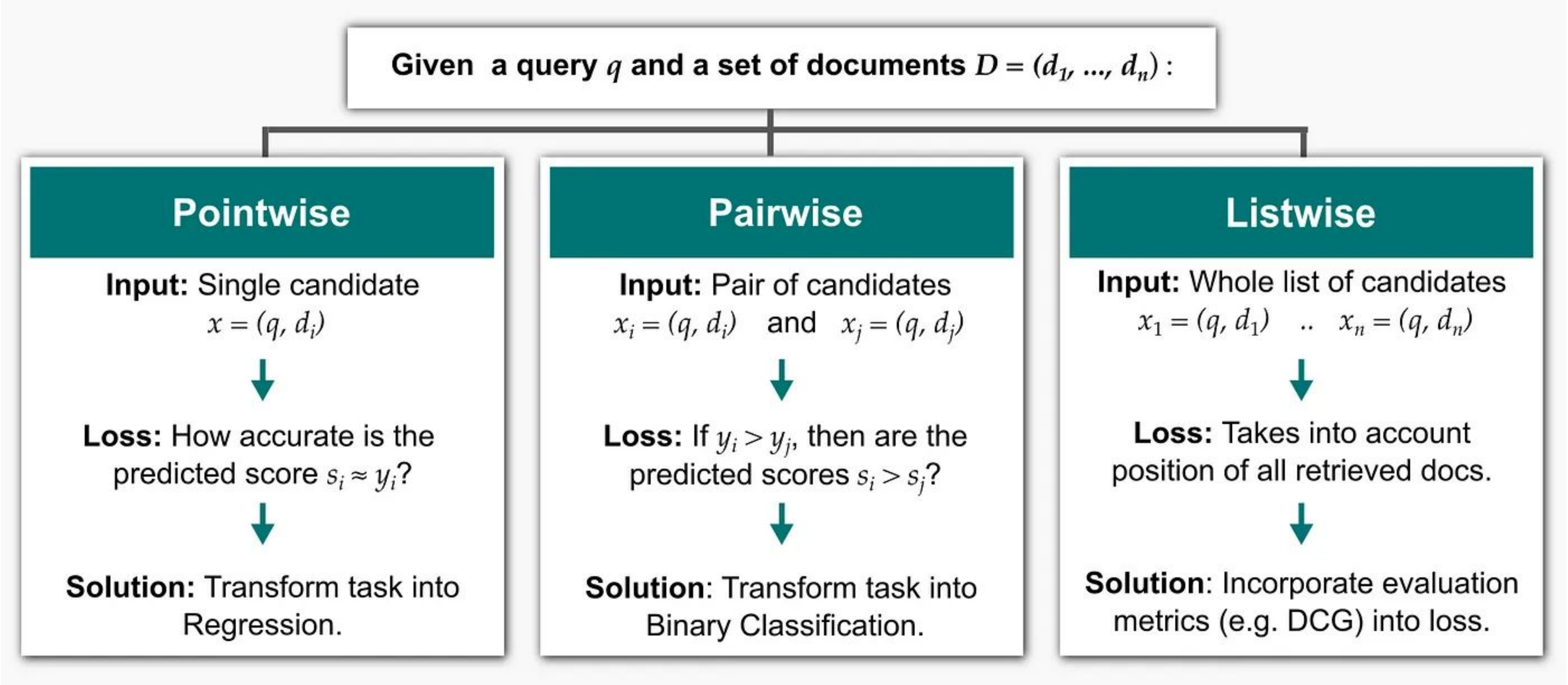
Learning to rank algorithms

"Learning to Rank" (LTR) refers to a set of machine learning techniques that are specifically designed to solve ranking problems. In essence, these **algorithms aim to order a list of items** (like web pages, products, or videos) based on their relevance to a given query or user.

Core Idea: Instead of simply classifying items as "relevant" or "irrelevant," LTR algorithms focus on **determining the optimal order** of those items.

Three types of learning to rank algorithms

- Pointwise
- Pairwise
- Listwise



https://towardsdatascience.com/learning-to-rank-a-complete-guide-to-ranking-using-machine-learning-4c9688d370d4

Pointwise Approach

Concept: Treats ranking as a **regression or classification** problem for **individual** documents.

How It Works:

- Each document is assigned a relevance score.
- The model predicts a score for each document independently.
- Uses standard regression or classification techniques.

Example: Predicting relevance scores for search results like:

Query: "best laptop 2025"

Doc1: "Laptop A review" → 0.95

Doc2: "Laptop B features" → 0.87

Algorithms: Linear Regression, Decision Trees, Deep Learning.

Pointwise

query	doc id	rank	click?	Term score	Title Score	Recency
1	30	1	1	0.6	0.2	5
1	12	2	0	0.4	0.1	10
1	11	3	1	0.35	0.5	3
1	50	4	0	0.2	0.5	2
2	12	1	0	0.9	0.2	4
2	7	2	0	0.2	0.6	2
2	30	3	0	0.1	0.5	1
2	4	4	1	0.1	0.1	4

$$\sum_w tf_{w,Q} \cdot \frac{tf_{w,D}}{tf_{w,D} + \frac{k|D|}{avg|D|}} \cdot \log \frac{|C|}{df_w}$$

Pointwise

y (Label) X (Features)

query	doc id	rank	click?	Term score	Title Score	Recency
1	30	1	1	0.6	0.2	5
1	12	2	0	0.4	0.1	10
1	11	3	1	0.35	0.5	3
1	50	4	0	0.2	0.5	2
2	12	1	0	0.9	0.2	4
2	7	2	0	0.2	0.6	2
2	30	3	0	0.1	0.5	1
2	4	4	1	0.1	0.1	4

A simple **linear ranking model** scores each document based on a weighted sum of features:

$$f(x) = w_1 \cdot \text{Term Score} + w_2 \cdot \text{Title Score} + w_3 \cdot \text{Recency} + b$$

where:

- w_1, w_2, w_3 = learned weights for each feature
- b = bias term

Deep Learning for Search

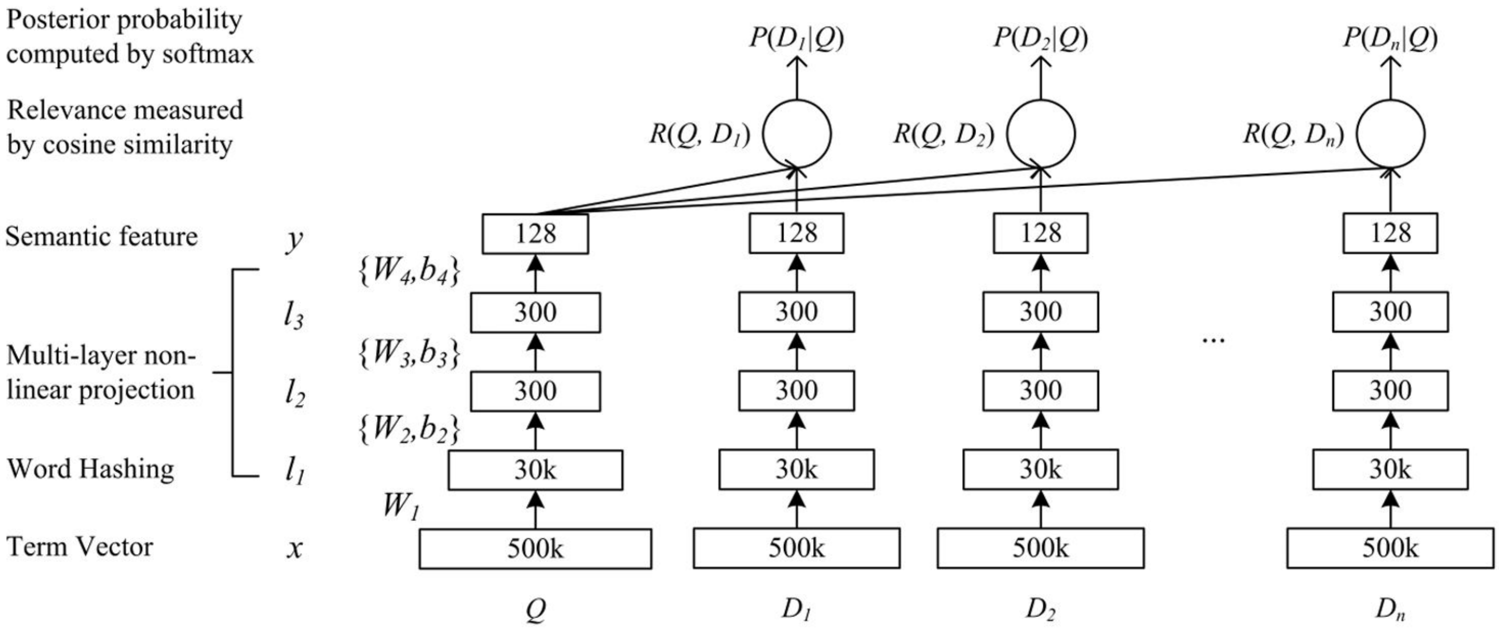
Vector space model

- We embed the query and the documents into embeddings by counting and weighting a bag of terms. Can we do better?
- Deep learning NLP model should capture the meaning better than a bag-of-word feature vector.

Word hashing technique

- It is a poor man's word embedding.
- A term/word embedding is a bag of character trigrams. Why should this work?
- Example: ความสวย → **ควา วาม** ามส มสว **สวย**

Deep Semantic Similarity Model (2013)



Convolutional DSSM (2014)

Replace feedforward net with a convolutional layer

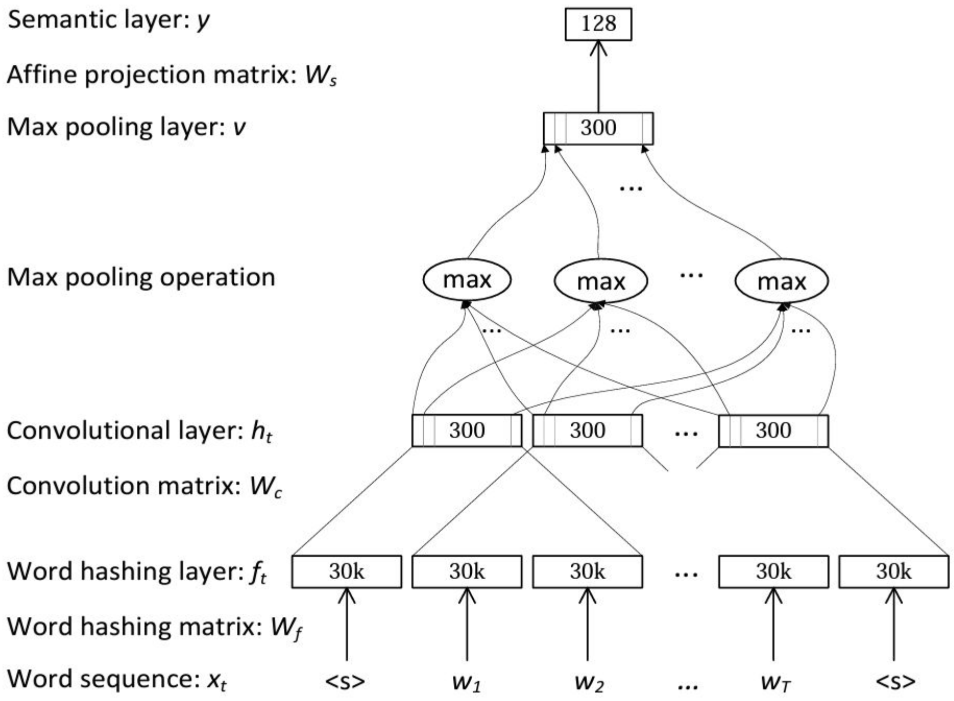


Figure 1: Illustration of the C-DSSM. A convolutional layer with the window size of three is illustrated.

Results

CNN does work better. And deep learning doesn't disappoint us.

Table 1: Comparative results with the previous approaches.

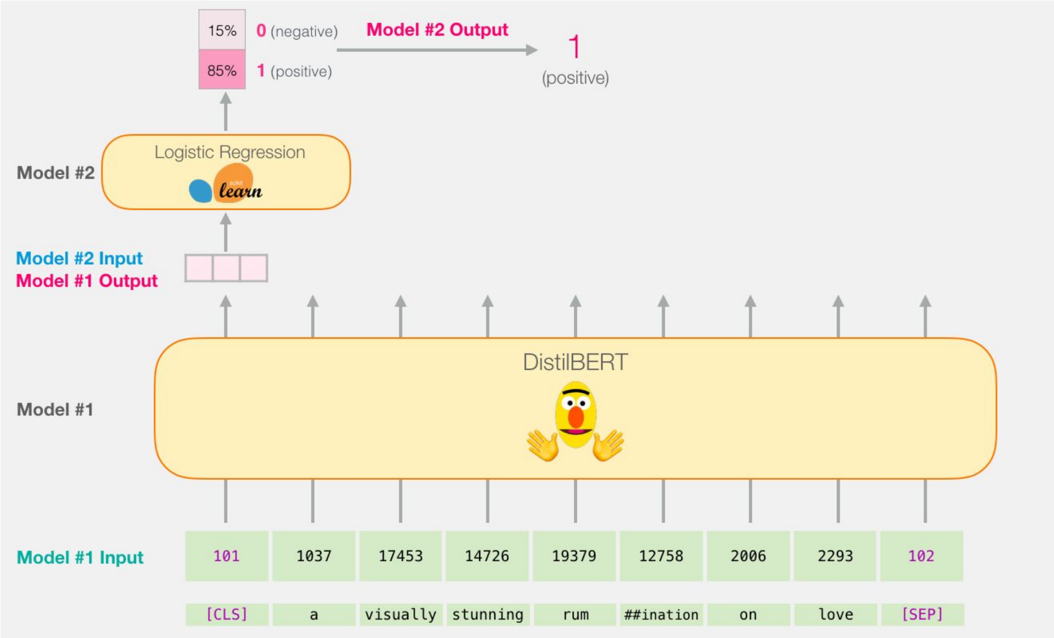
#	Models	NDCG@1	NDCG@3	NDCG@10
1	BM25	0.305	0.328	0.388
2	ULM	0.304	0.327	0.385
3	WTM	0.315^{α}	0.342^{α}	0.411^{α}
4	PTM (len $\leq$ 3)	0.319^{α}	0.347^{α}	0.413^{α}
5	DSSM	0.320^{α}	$0.355^{\alpha\beta}$	$0.431^{\alpha\beta}$
6	**C-DSSM win =3**	$\mathbf{0.342}^{\alpha\beta\gamma}$	$\mathbf{0.374}^{\alpha\beta\gamma}$	$\mathbf{0.447}^{\alpha\beta\gamma}$

Shen, Yelong, et al. "Learning semantic representations using convolutional neural networks for web search." *Proceedings of the 23rd international conference on world wide web*. 2014.

Transformer (BERT) 2019

Use pretrained transformer (language model) to convert query/document into an embedding

This is called 'encoding'

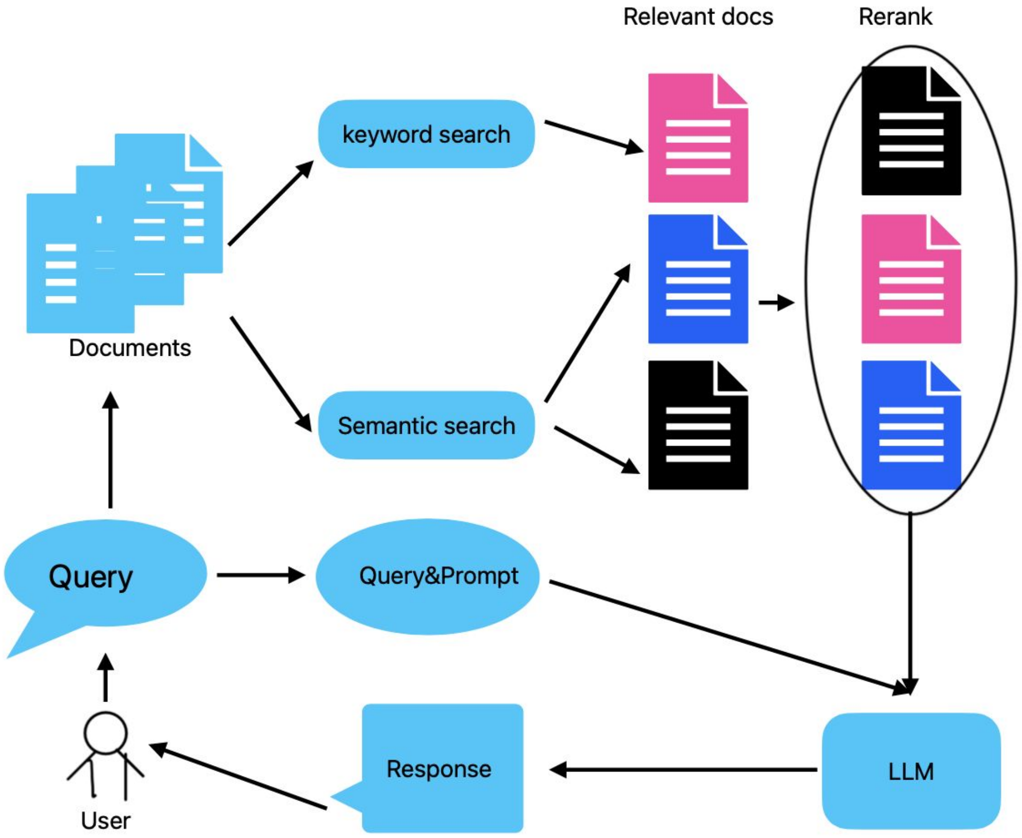


Searching in LLM Era

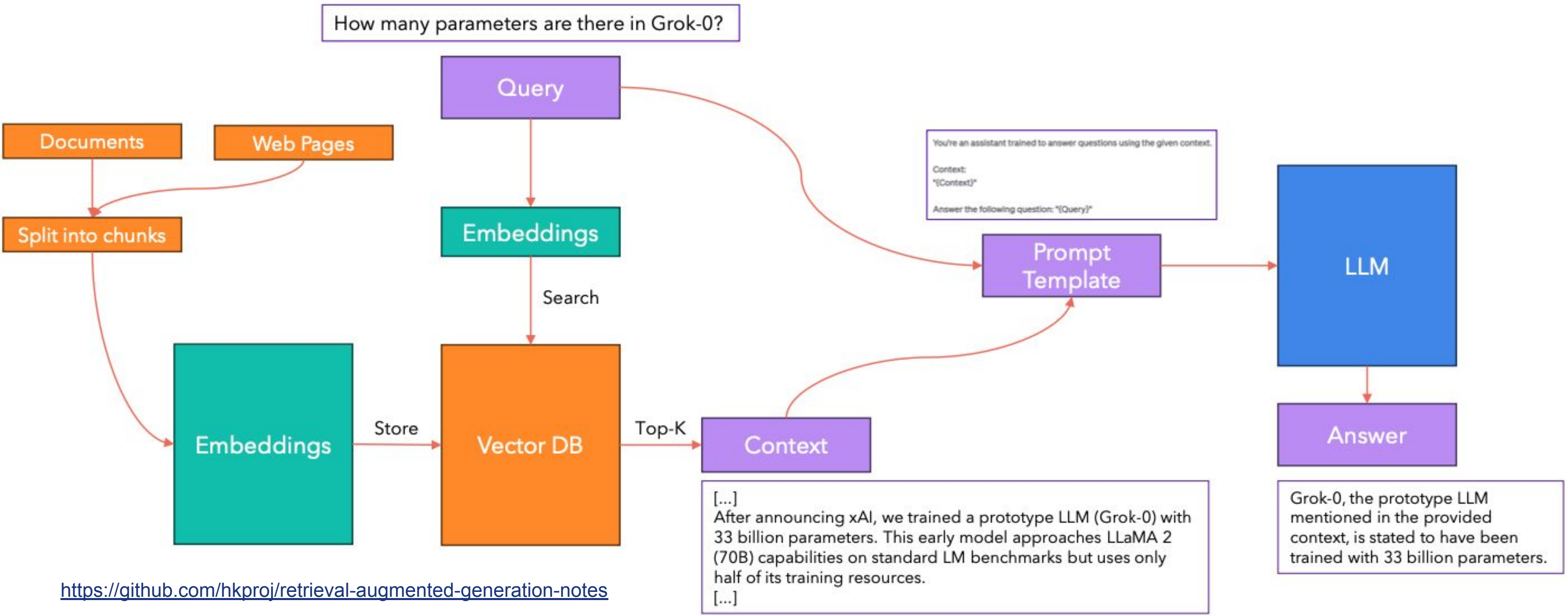
Retrieval-Augmented Generation (RAG) is a **hybrid AI approach** that combines:

1. **Information Retrieval (IR)**: Fetches relevant documents from an external knowledge base.
2. **Generative AI (LLMs - Large Language Models)**: Uses the retrieved documents to generate contextually accurate responses.

RAG helps models stay **factual, up-to-date, and grounded in real-world data** by integrating external **retrieval systems** before generating text.

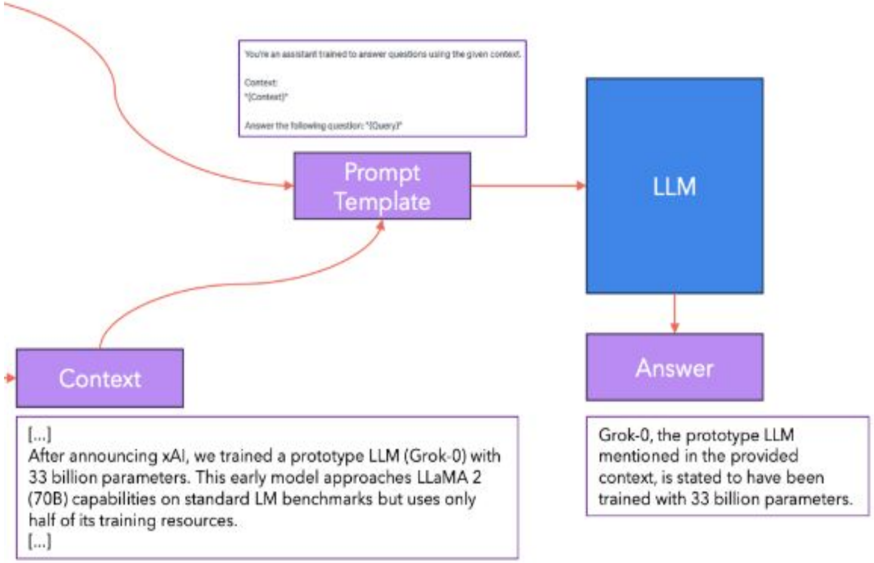


RAG Pipeline



https://github.com/hkproj/retrieval-augmented-generation-notes

Adding Source Data in the Context of Prompt



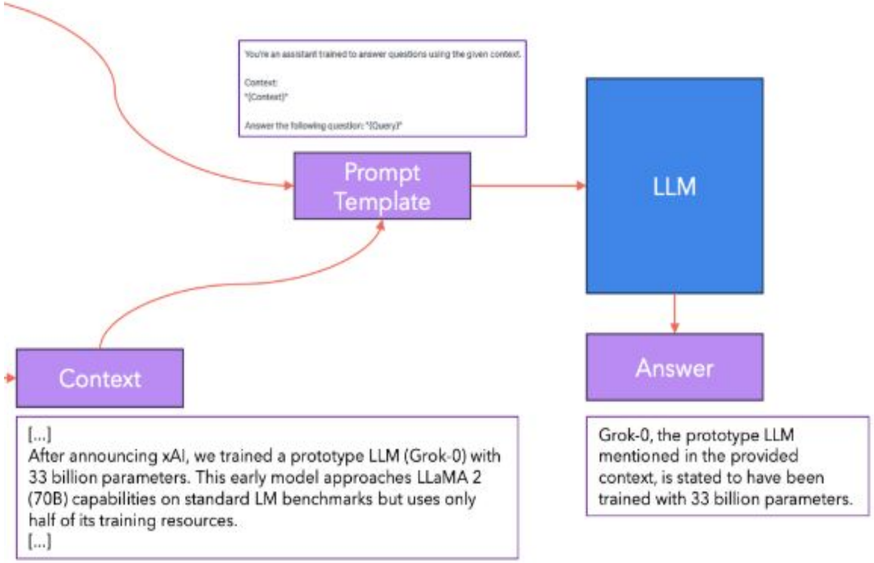
Your task is to answer the following question. To help you with this, relevant texts are provided. Please base your answer on these texts.

Question:
How many parameters are there in Grok-0?

Relevant Text 1:
After announcing xAI, we trained a prototype LLM (Grok-0) with 33 billion parameters….

Relevant Text 2:…..

Adding Source Data in the Context of Prompt



Your task is to answer the following question. To help you with this, relevant texts are provided. Please base your answer on these texts.
Please note that your answers need to be as accurate as possible and faithful to the facts. If the information provided is insufficient for an accurate response, you may simply output "No answer!".

Question:
How many parameters are there in Grok-0?

Relevant Text 1:
After announcing xAI, we trained a prototype LLM (Grok-0) with 33 billion parameters.…

Relevant Text 2:…..

Conclusion

- **Search is integral to many** modern applications.
- Information retrieval systems require you to understand the **queries and also the context** of searching in order to address the information needs of the users (what they actually want).
- There is **no one solution for search engine**. NLP engineers are often required to improve search experience.