

ACL 2023 Tutorial:

Retrieval-based Language Models and Applications

Akari Asai, Sewon Min, Zexuan Zhong, Danqi Chen

University of Washington, Princeton University

발표자: 송선영

2024/01/09

Retrieval for knowledge-intensive NLP tasks

- Representative tasks: open-domain QA, fact checking, entity linking, ...



Why do retrieval-based LMs need?

1. LLMs can't memorize all knowledge in their parameters

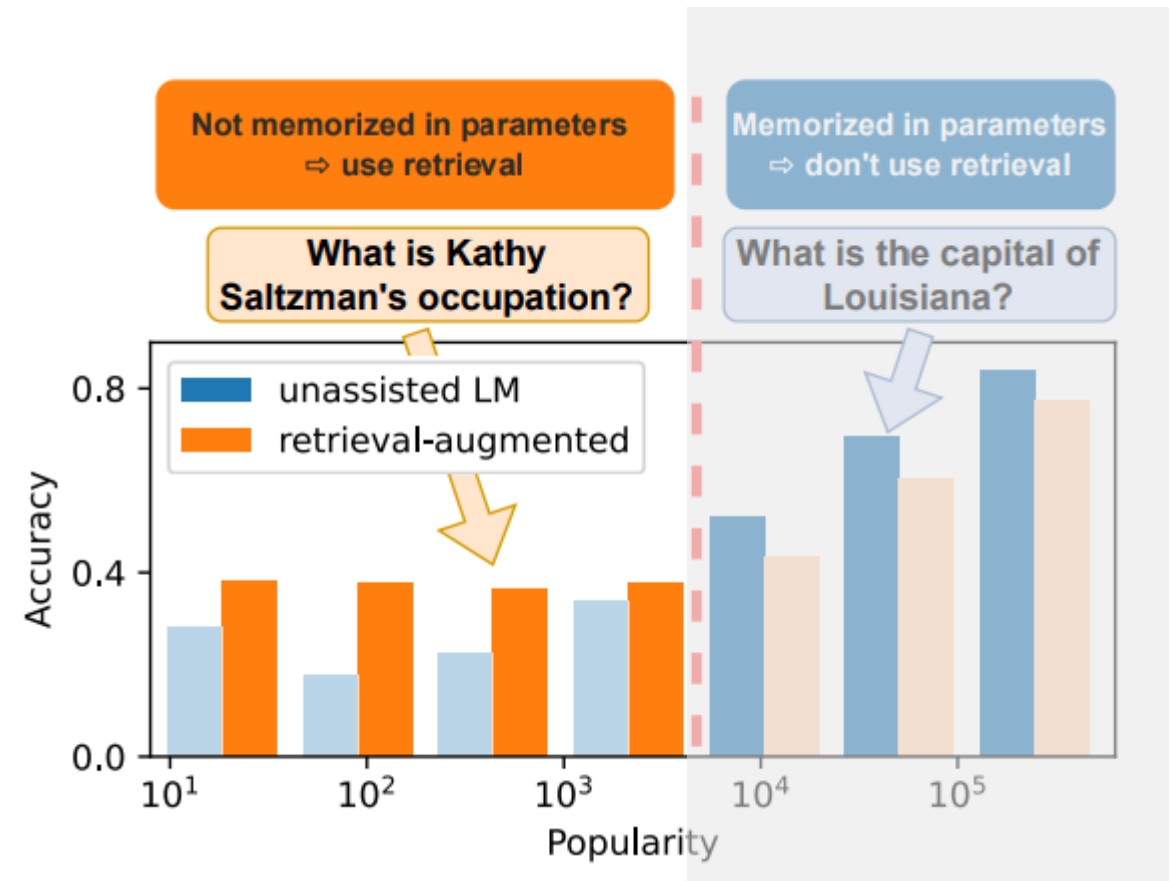


List 5 important papers authored by Geoffrey Hinton



Geoffrey Hinton is a renowned computer scientist ...
Here are five important papers authored by him:

1. **"Learning Internal Representations by Error Propagation"** (with **D. E. Rumelhart** and **R. J. Williams**) - This paper, published in **1986**, .. ✓
2. **"Deep Boltzmann Machines"** (with **R. Salakhutdinov**) - Published in **2009**, .. ✓
- ...
4. **"Deep Learning"** (with Y. Bengio and A. Courville) - Published as a book in **2016**,... ✗
5. **"Attention Is All You Need"** (with V. Vaswani, N. Shazeer, et al.) - Published in **2017**, this paper introduced the Transformer model,... ✗



Why do retrieval-based LMs need?

2. LLMs' knowledge is easily outdated and hard to update



Who is the CEO of Twitter?



As of my **knowledge cutoff in September 2021**, the CEO of Twitter is **Jack Dorsey**....



Who is the CEO of Twitter?



 All

 News

 Images

 Shopping

 Videos

 More

Tools

About 1,090,000,000 results (0.45 seconds)

Twitter / CEO

Linda Yaccarino

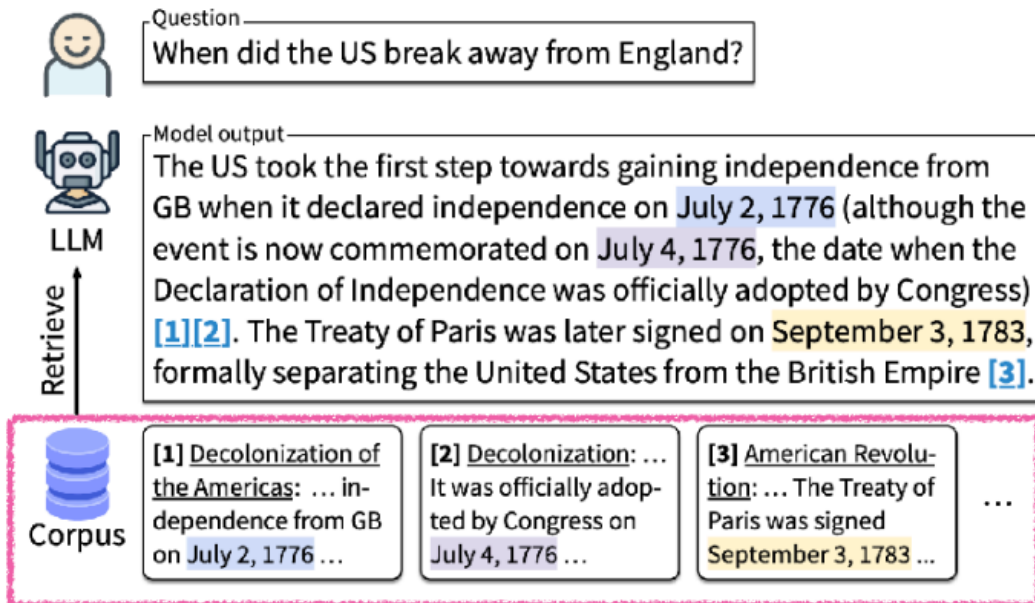
Jun 5, 2023–



Why do retrieval-based LMs need?

3. LLMs' output is challenging to interpret and verify

→ **Solution:** Generating text with citations



Toronto is a vibrant and diverse city with many attractions and activities to suit different tastes and preferences. Here are some suggestions for creating an itinerary for exploring Toronto over two days:

- **Day 1:** Start your day with a visit to the **CN Tower**, the iconic landmark that offers panoramic views of the city and beyond ¹. You can also experience the thrill of walking on the glass floor or the EdgeWalk, the world's highest hands-free walk ¹. Next, head to the nearby **St. Lawrence Market**, one of the world's best food markets, where you can sample a variety of cuisines and local specialties ². After lunch, take a stroll along **Queen West**, a trendy neighborhood with eclectic shops, galleries, cafes and street art ³. In the evening, enjoy a show at one of the many theaters or comedy clubs in the **Entertainment District**, or catch a game at the **Scotiabank Arena** if you're a sports fan.

- **Day 2:** Explore the history and culture of Toronto at the **Royal Ontario Museum**, one of the largest and most comprehensive museums in North America, featuring exhibits on art, natural history, world cultures and more ⁴. Then, hop on a ferry to the **Toronto Islands**, a group of islands that offer a relaxing escape from the city, with beaches, parks, trails and amusement rides ^{3 5}. You can also rent a bike or kayak to explore the islands at your own pace. For dinner, head to **Chinatown**, one of the largest and most vibrant in North America, where you can find a variety of Asian cuisines and shops ³.

I hope this helps you plan your trip to Toronto. Have fun! 😊

Learn more:

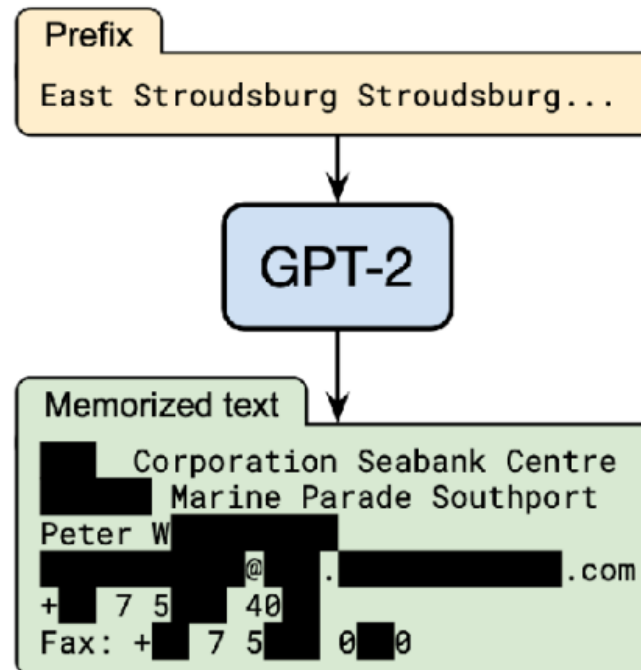
- 1. [cntower.ca](#)
- 2. [travel.usnews.com](#)
- 3. [bing.com](#)
- 4. [rom.on.ca](#)
- 5. [tripadvisor.com](#)

Why do retrieval-based LMs need?

4. LLMs are shown to easily leak private training data

- Private data: address email, phone number, ...

→ **Solution:** Individualization on private data by storing it in the datastore

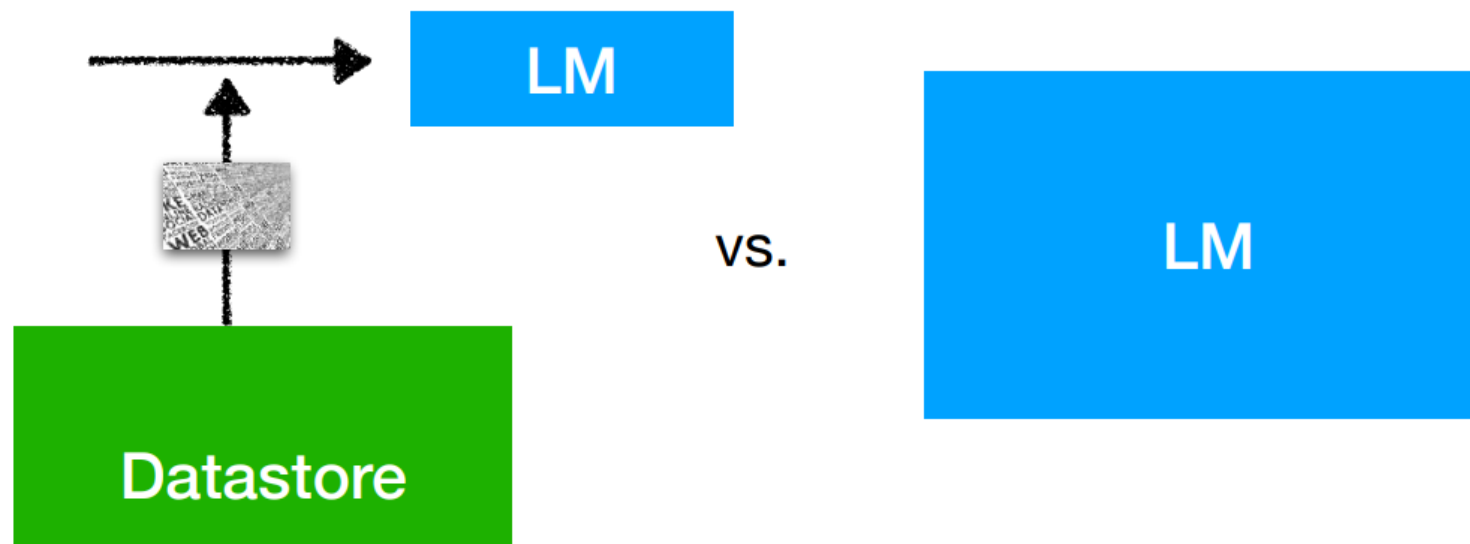


Why do retrieval-based LMs need?

5. LLMs are *large* and expensive to train and run

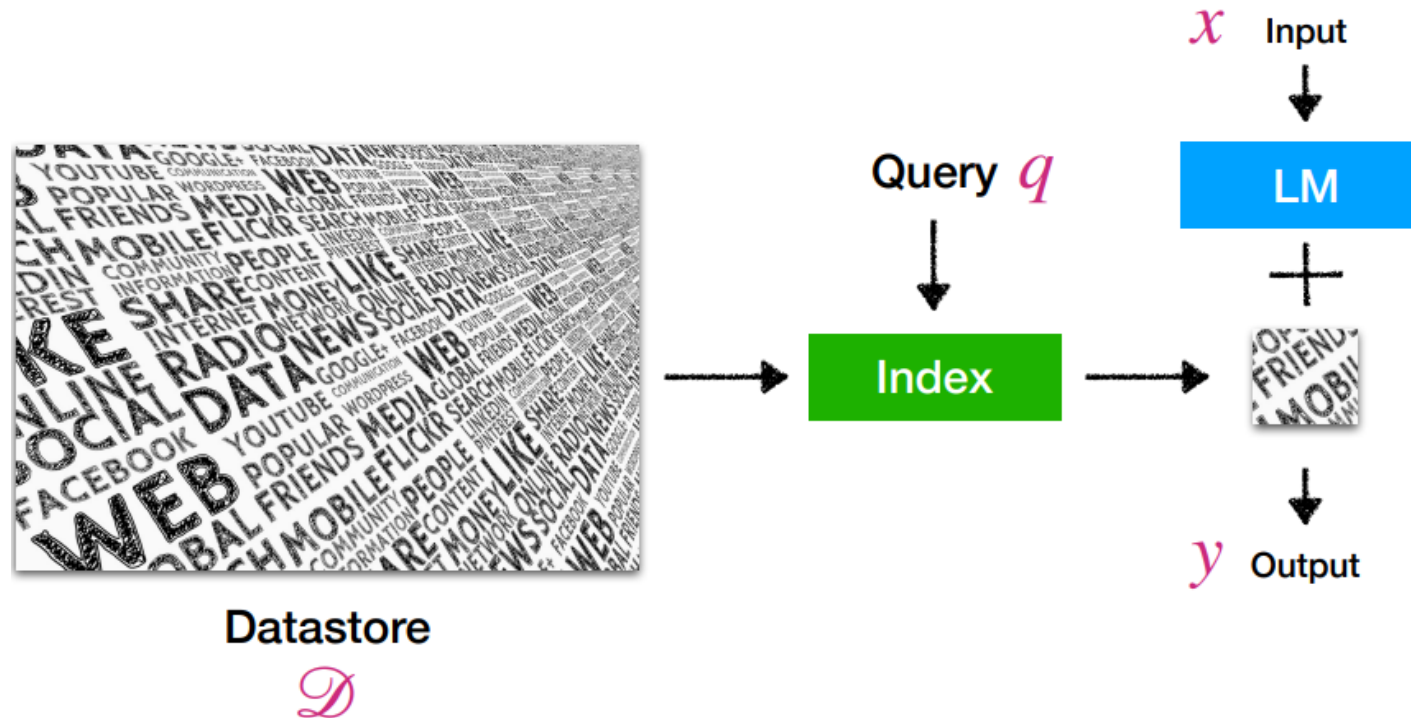
→ **Solution:** to directly access a large external database

- Improving Language Models by Retrieving from Trillions of Tokens (RETRO) (Borgeaud, Sebastian, et al. PMLR, 2022)
- RETRO obtains comparable performance to GPT-3 and Jurassic-1 on the Pile, despite using 25x fewer parameters



What is a Retrieval-based LM?

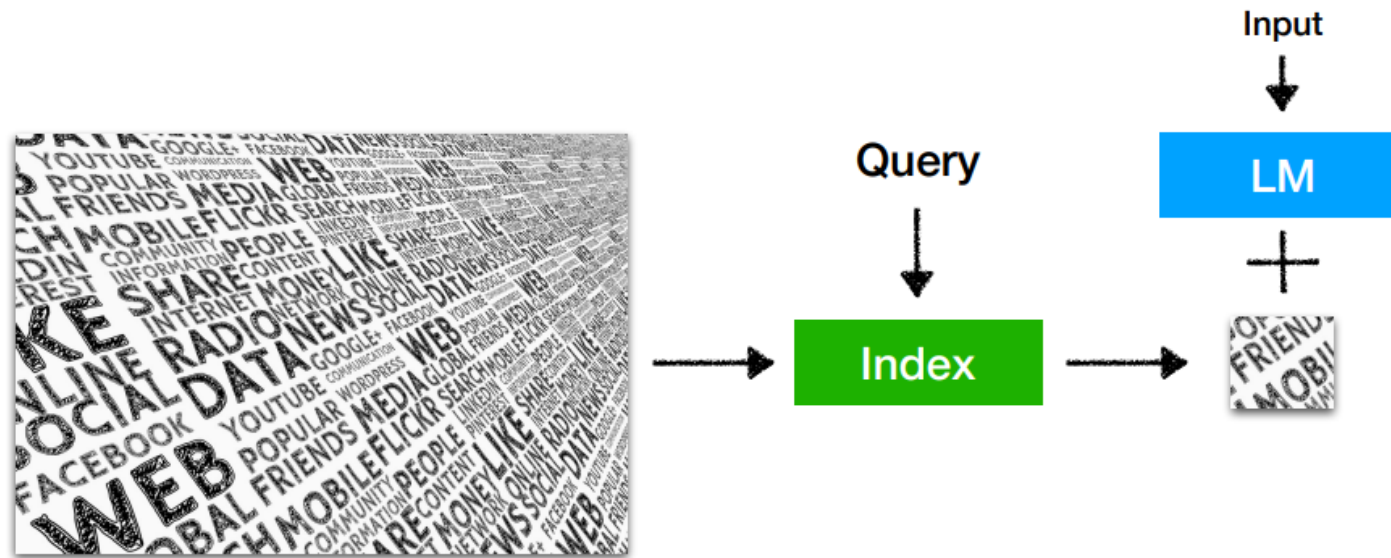
- Retrieval-based LM is a language model (LM) that uses an external datastore at test time



Inference

1. Datastore

- consists of raw text corpus
- It's very large-scale consisting of at least billions or trillion tokens



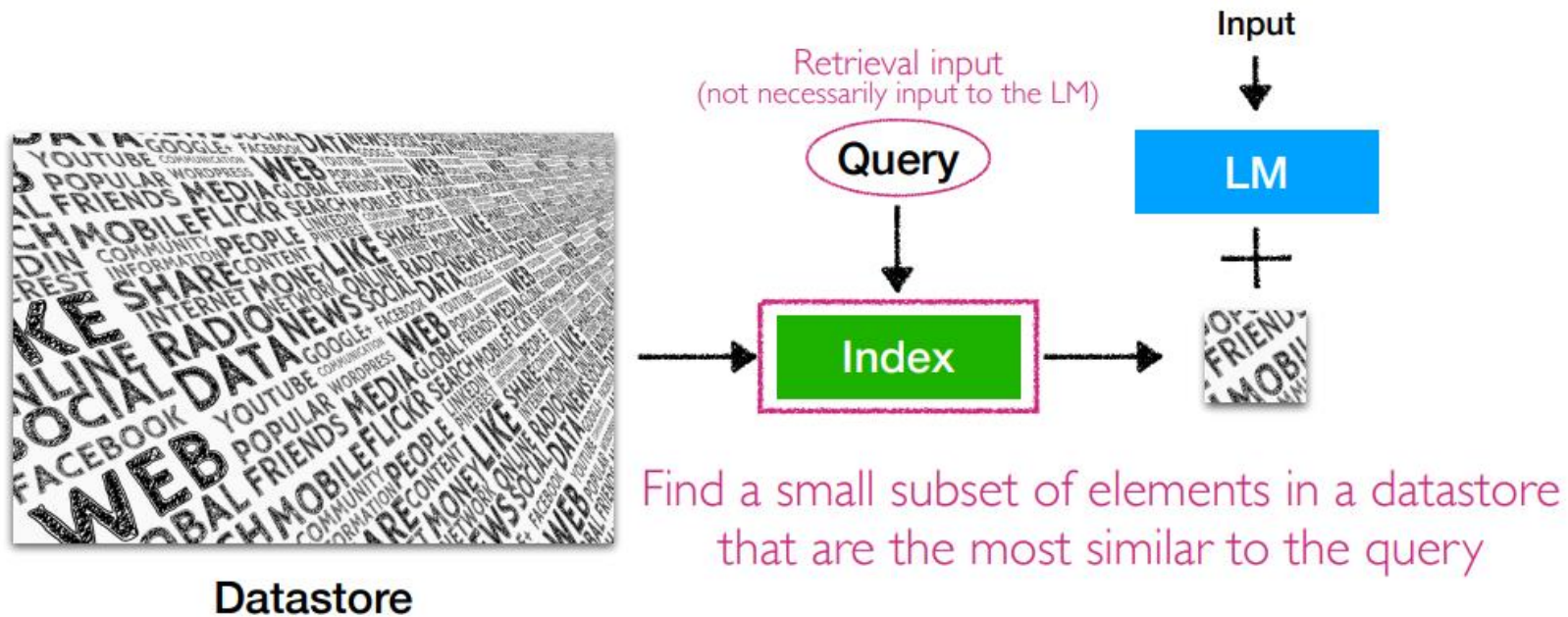
Datastore
Raw text corpus

At least billions~trillions of tokens
Not labeled datasets
Not structured data (knowledge bases)

Inference

2. Index

- Input: query
- Output: a small subset of elements in a datastore that are the most similar to the query



Inference

2. Index

- **Similarity score** is a score that represent how similar two pieces of text is in semantically
 - **TF-IDF**

Example $\text{sim}(i, j) = \text{tf}_{i,j} \times \log \frac{N}{\text{df}_i}$

$\text{tf}_{i,j}$: # of occurrences of i in j
 N : # of total docs
 df_i : # of docs containing i

- **Mapping the text into the dense vector**
 - Using the neural encoder
 - The similarity score would be an inner product between these two vectors

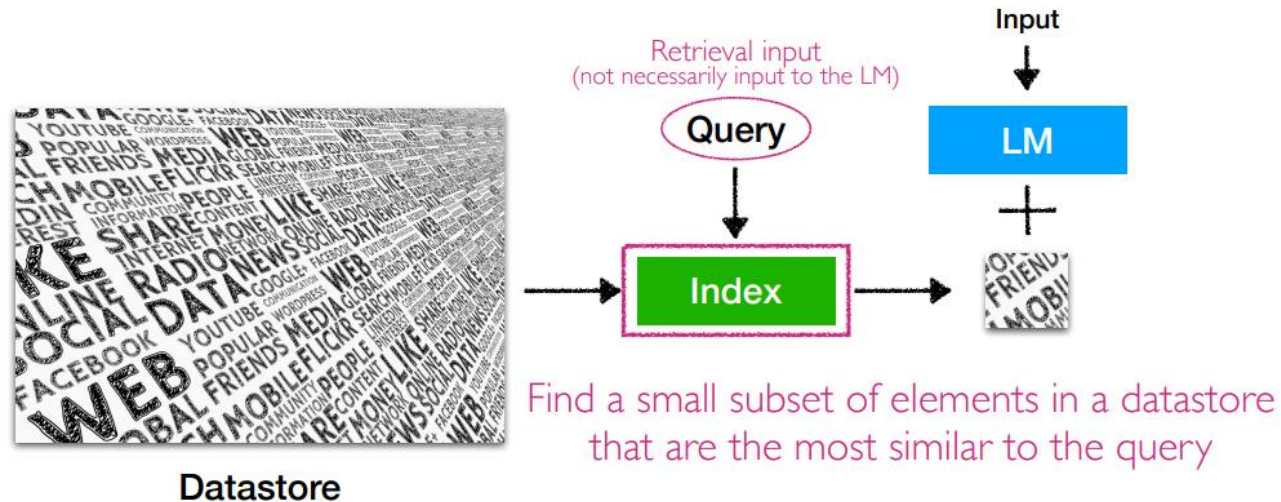
Example $\text{sim}(i, j) = \text{Encoder}(i) \cdot \text{Encoder}(j)$

$\text{Encoder}(i)$ and $\text{Encoder}(j)$: Maps the text into an h -dimensional vector

Inference

2. Index

- Input: query
- Output: a small subset of elements in a datastore that are the most similar to the query



Can be a totally separate research area on how to do this fast & accurate

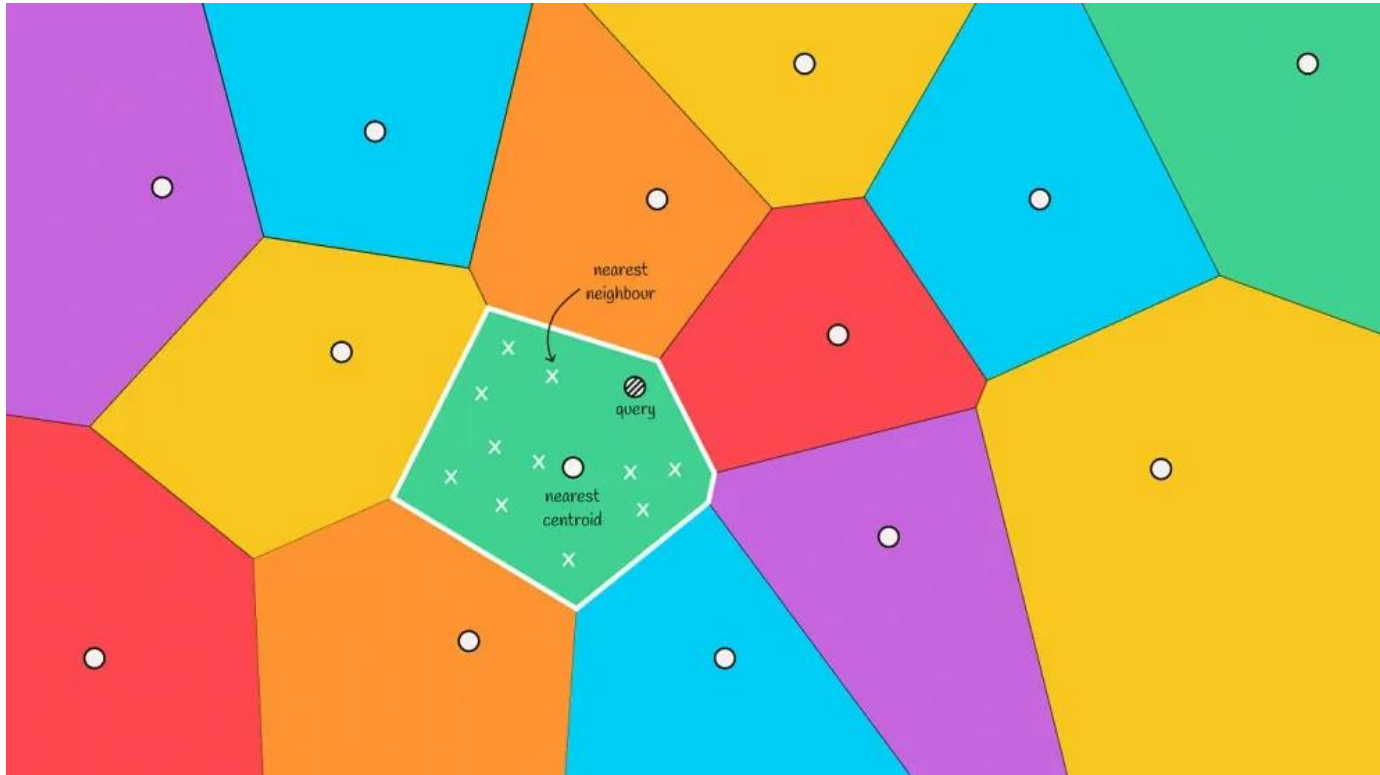
Index: given q , return $\arg\text{Top-}k_{d \in \mathcal{D}} \text{sim}(q, d)$ through fast nearest neighbor search

k elements from a datastore

Inference

2. Index

- FAISS (Facebook AI Similarity Search)
 - Facebook AI 연구팀에서 만든 벡터 클러스터링 및 similarity search library



Inference

2. Index

- FAISS (Facebook AI Similarity Search)

```
!pip install faiss
```

```
import faiss
import torch
```

```
database_size = 100000
num_queries = 1
dimension = 10
```

```
db_vector = np.random.random((database_size, dimension)).astype('float32')
query_vector = np.random.random((num_queries, dimension)).astype('float32')
```

```
# Build an index와 index에 벡터 더하기
faiss_index = faiss.IndexFlatL2(dimension)
faiss_index.add(db_vector)
```

```
k = 4 # k-th nearest
Distance, Index = faiss_index.search(db_vector[:5], k)
print(f'Index:\n {Index}\n')
print(f'Distance:\n {Distance}')
```

```
Index:
[[ 0 97535 93445 39043]
 [ 1 90328 71553 22453]
 [ 2 4693 9260 97107]
 [ 3 85724 90148 789]
 [ 4 10436 75005 68128]]
```

```
Distance:
[[0.          0.09631573 0.12315865 0.14702493]
 [0.          0.11033922 0.12349739 0.1381162 ]
 [0.          0.08538684 0.09451801 0.0999269 ]
 [0.          0.09128504 0.09506419 0.12409284]
 [0.          0.11630931 0.1257291  0.13669585]]
```

```
Distance, Index = faiss_index.search(query_vector, k)
print(f'Index:\n {Index}\n')
print(f'Distance:\n {Distance}')
```

```
Index:
[[40365 14002 12745 83120]]
```

```
Distance:
[[0.07748464 0.0808847  0.08550636 0.10243338]]
```

Inference

2. Index

Method	Class name	index_factory	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	IndexFlatL2	"Flat"	d	4*d	yes	brute-force
Exact Search for Inner Product	IndexFlatIP	"Flat"	d	4*d	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	IndexHNSWFlat	"HNSW, Flat"	d, M	$4*d + x * M * 2 * 4$	no	
Inverted file with exact post-verification	IndexIVFFlat	"IVFx, Flat"	quantizer, d, nlists, metric	4*d + 8	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	IndexLSH	-	d, nbits	$\text{ceil}(\text{nbits}/8)$	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	IndexScalarQuantizer	"SQ8"	d	d	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	IndexPQ	"PQx", "PQ"M"x"nbits	d, M, nbits	$\text{ceil}(M * \text{nbits} / 8)$	yes	
IVF and scalar quantizer	IndexIVFScalarQuantizer	"IVFx, SQ4" "IVFx, SQ8"	quantizer, d, nlists, qtype	SQfp16: $2 * d + 8$, SQ8: $d + 8$ or SQ4: $d/2 + 8$	no	Same as the IndexScalarQuantizer
IVFADC (coarse quantizer+PQ on residuals)	IndexIVFPQ	"IVFx, PQ"x"nbits	quantizer, d, nlists, M, nbits	$\text{ceil}(M * \text{nbits}/8) + 8$	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	IndexIVFPQR	"IVFx, PQy+z"	quantizer, d, nlists, M, nbits, M_refine, nbits_refine	M*M_refine+8	no	

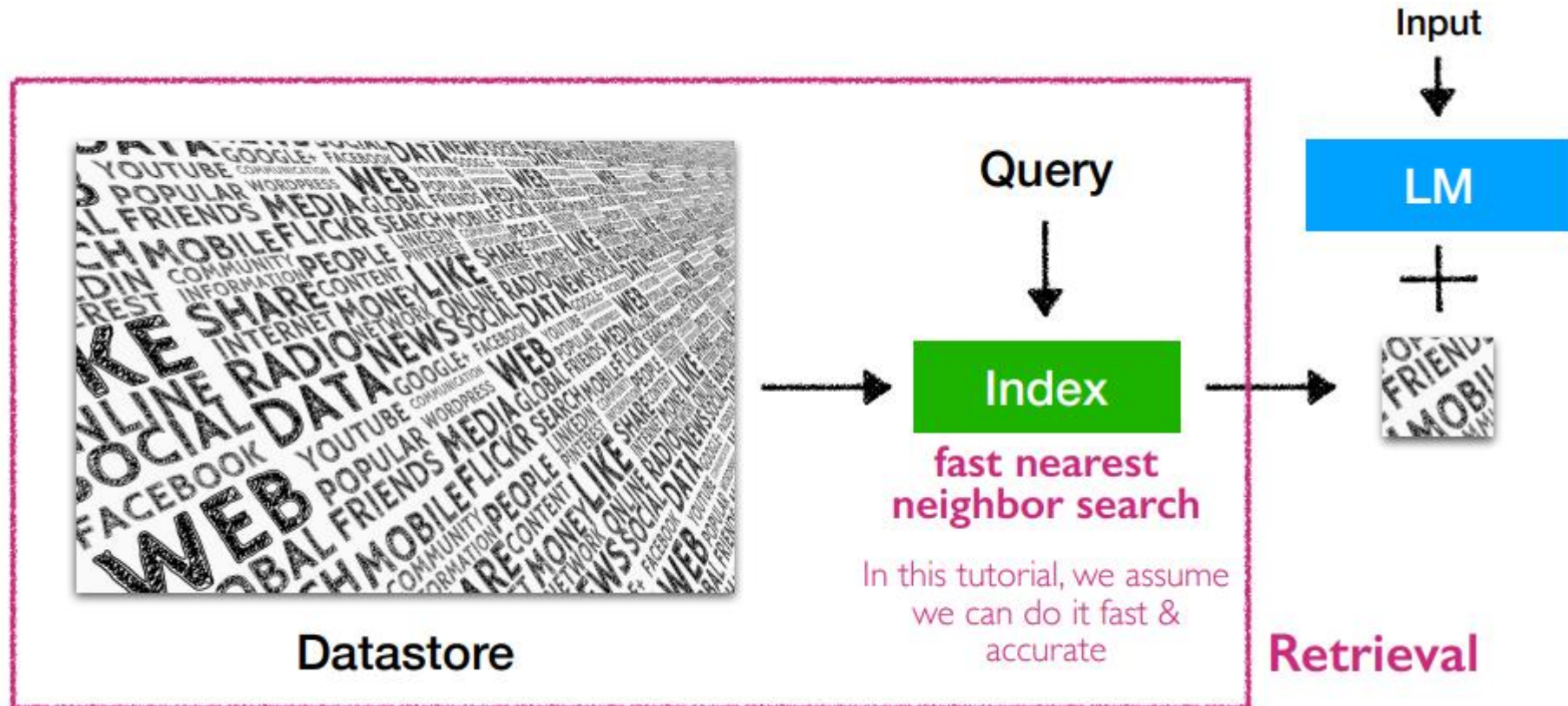
Exact Search

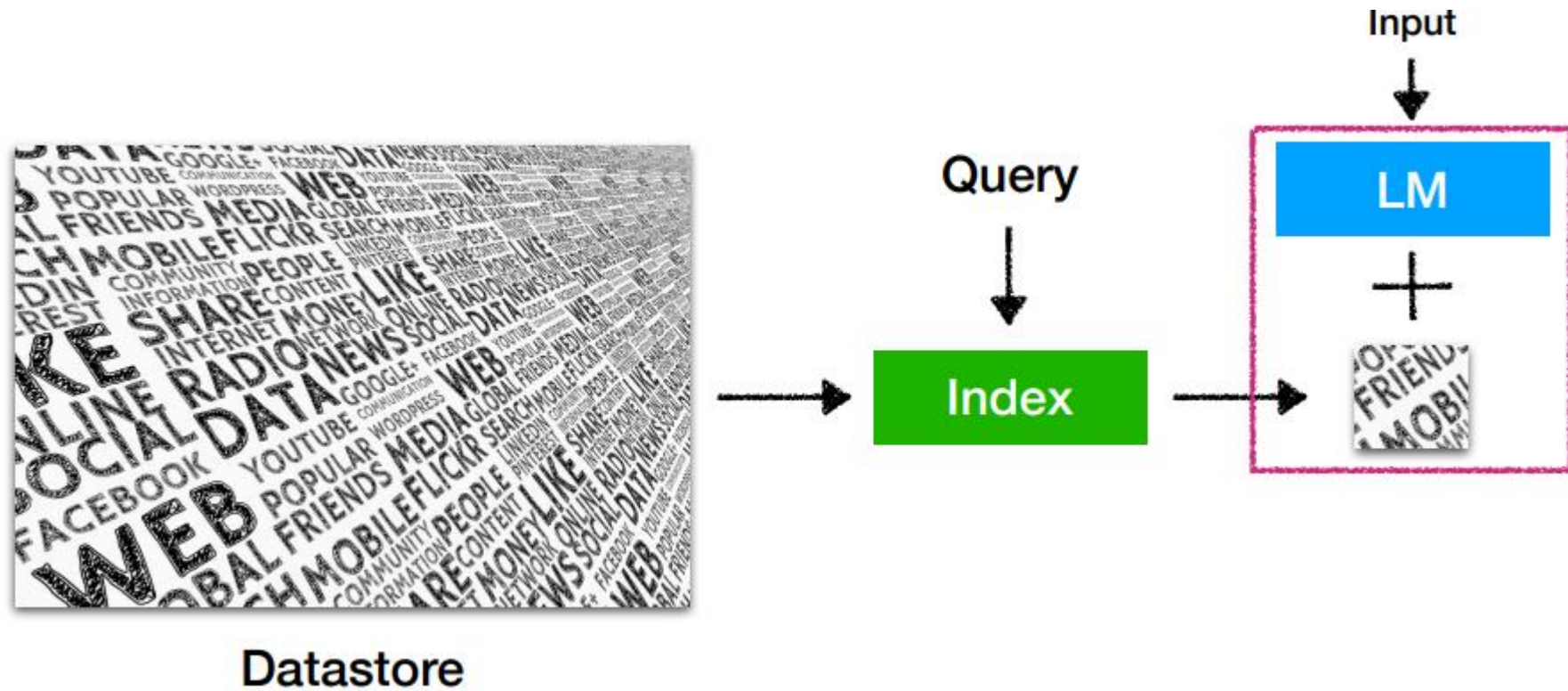
Approximate Search
(Relatively easy to scale to ~1B elements)

Inference

3. Search

- Retrieval system is a combination of datastore and the index





Thank You

감사합니다.