


Generative Information Retrieval

SIGIR 2024 tutorial – Section 1

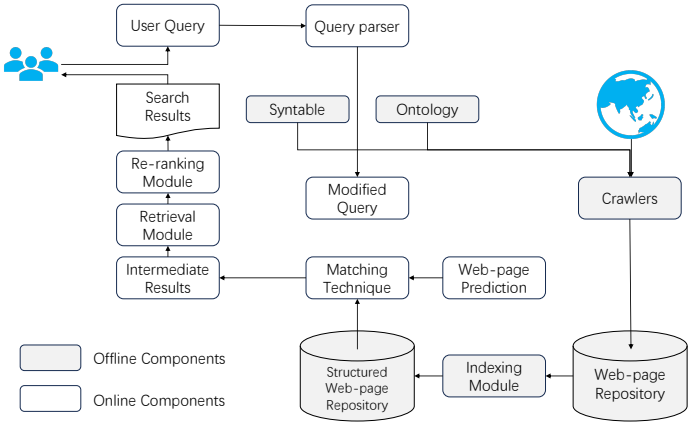
Yubao Tang[a], Ruqing Zhang[a], **Zhaochun Ren**[b], Jiafeng Guo[a] and **Maarten de Rijke**[c]

https://generative-ir.github.io/

July 14, 2024

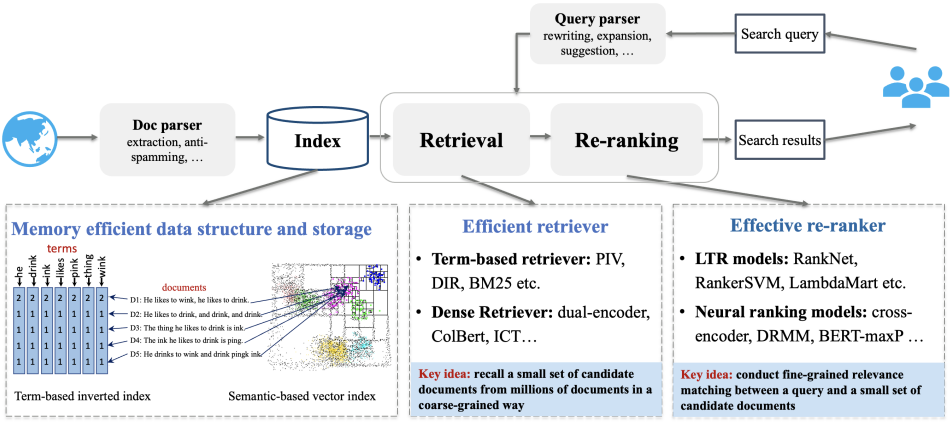
[a] Institute of Computing Technology, Chinese Academy of Sciences & UCAS
[b] Leiden University
[c] University of Amsterdam

Complex architecture design behind search engines



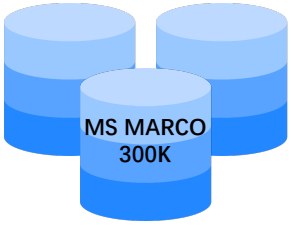
- **Advantages**:
  - Pipelined paradigm has withstood the test of time
  - Advanced machine learning and deep learning approaches applied to many components of modern systems

Index-Retrieval-Ranking: Disadvantages



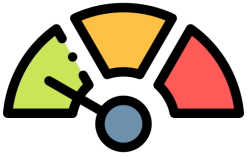
- **Effectiveness**: Heterogeneous ranking components are usually difficult to be optimized in an end-to-end way towards the global objective

Index-Retrieval-Ranking: Disadvantages



Big storage

GTR (Dense retrieval)
Memory size 1430MB



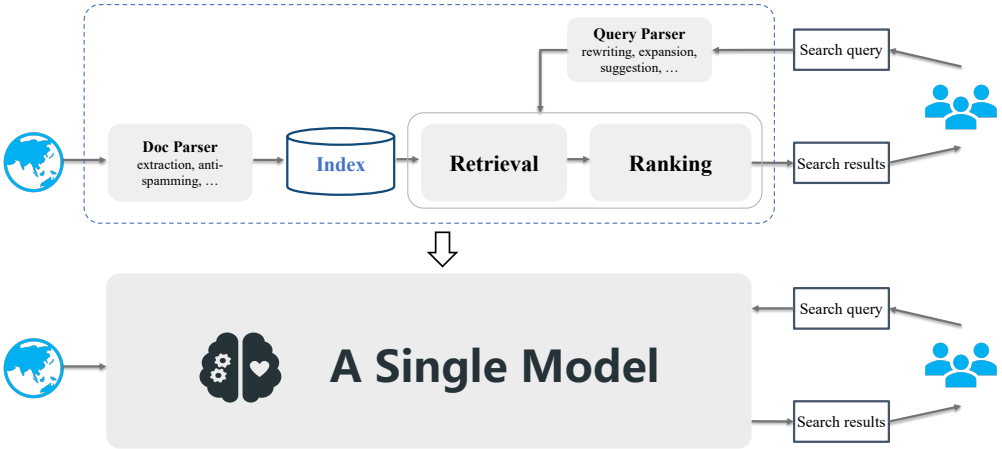
Slow inference speed

GTR (Dense retrieval)
Online latency 1.97s

- **Efficiency**: A large document index is needed to search over the corpus, leading to significant memory consumption and computational overhead

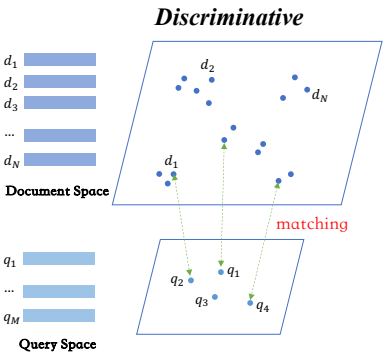
Source: [Sun et al., 2023]

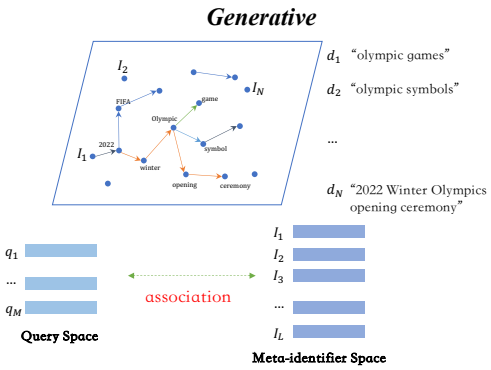
Opinion paper: A single model for IR



Source: [Metzler et al., 2021]

Neural IR models: Discriminative vs. Generative





$$p(R=1|q,d) \approx \ \ldots \ \approx argmax\ s(\vec{q},\vec{d})$$

(probabilistic ranking principle)

$$p(q|d) \approx p(docID|q) = argmax\ \mathrm{p}((I_1,\ldots,I_k)|q)$$

(query likelihood)

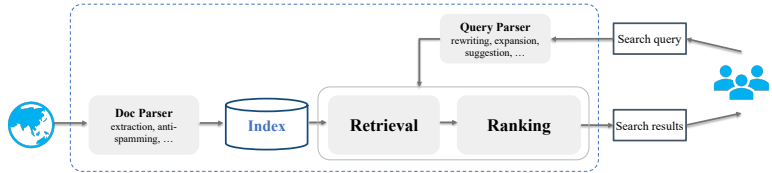
Two families of generative retrieval

- Closed-book: The language model is the **only source** of knowledge leveraged during generation, e.g.,
  - Capturing document ids in the language models
  - Language models as retrieval agents via prompting
- Open-book: The language model can draw on **external memory** prior to, during, and after generation, e.g.,
  - Retrieval augmented generation of answers
  - Tool-augmented generation of answers

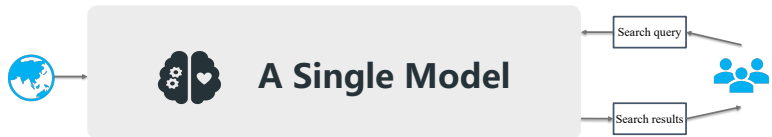
Source: [Najork, 2023]

Why generative retrieval?

Heterogeneous objectives

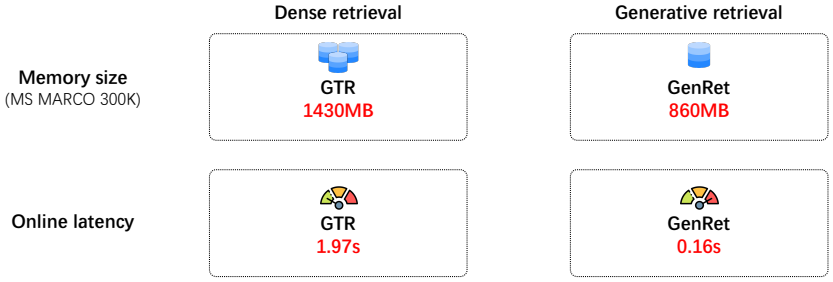


A global objective



- **Effectiveness**: Knowledge of all documents in corpus is encoded into model parameters, which can be optimized directly in an end-to-end manner

Why generative retrieval?



- **Efficiency**: Main memory computation of GR is the storage of document identifiers and model parameters
- Heavy retrieval process is replaced with a light generative process over the vocabulary of identifiers

Data source: [Sun et al., 2023]

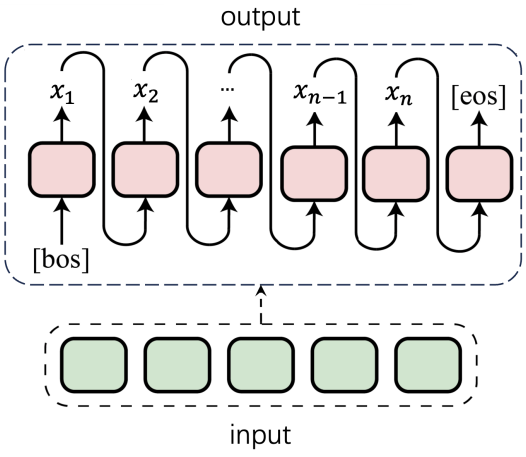
Generative retrieval: Definition

Generative retrieval (GR) aims to directly generate the identifiers of information resources (e.g., docids) that are relevant to an information need (e.g., an input query) in an autoregressive fashion

"Transformer Memory as a Differentiable Search Index". Tay et al. [2022]; "Autoregressive Entity Retrieval". De Cao et al. [2021]

Autoregressive formulation

$$P(x_n|x_1, x_2, \ldots, x_{n-1})$$

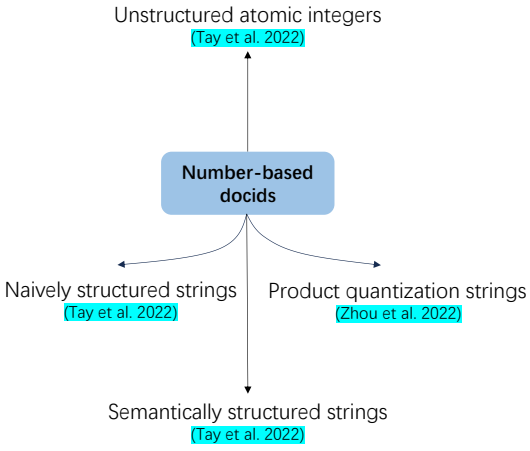


Two basic operations in GR

- Indexing: To **memorize information about each document**, a GR model should learn to associate the content of each document with its corresponding docid
- Retrieval: Given an input query, a GR model should **return a ranked list of candidate docids** by autoregressively generating the docid string

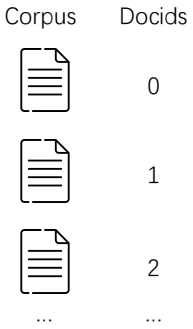
Source: [Tay et al., 2022]

A single docid: Number-based



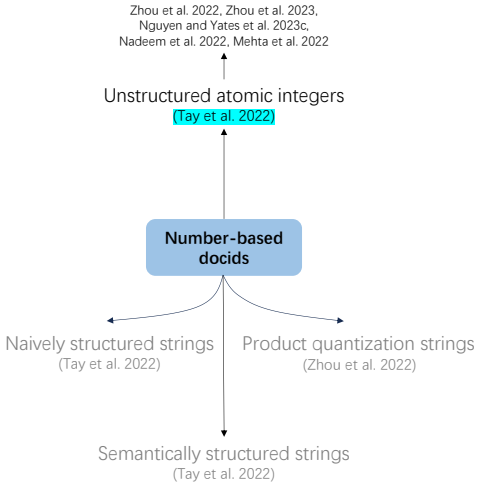
Number-based: Unstructured atomic integers

- An arbitrary (and possibly random) unique integer identifier



Source: [Tay et al., 2022]

Unstructured atomic integers and subsequent work



Easy to build: analogous to the output layer in standard language model

Unstructured atomic integers: obvious constraints

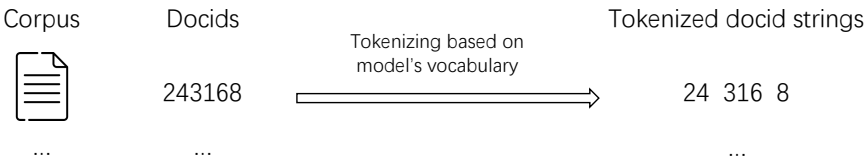
The need to learn embeddings for each individual docid

The need for the large softmax output space

It is challenging to be used on large corpora!

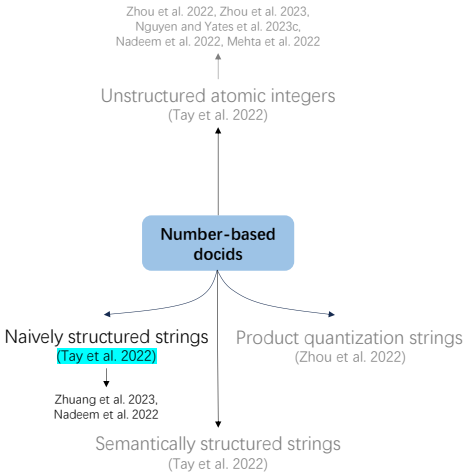
Number-based: Naively structured strings

- Treat arbitrary unique integers as tokenizable strings



Source: [Tay et al., 2022]

Naively structured strings and subsequent work



Such a way frees the limitation for the **corpus size** that comes with unstructured atomic docid

Naively structured strings: obvious constraints

Identifiers are assigned in an arbitrary manner

The docid space lacks semantic structure

Number-based: Semantically structured strings

Properties:

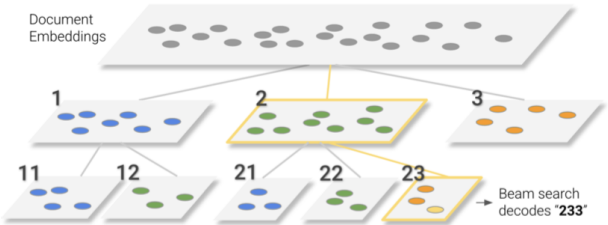
- The docid should capture some information about the semantics of its associated document
- The docid should be structured in a way that the search space is effectively reduced after each decoding step

Semantically similar documents share docid prefixes

Source: [Tay et al., 2022]

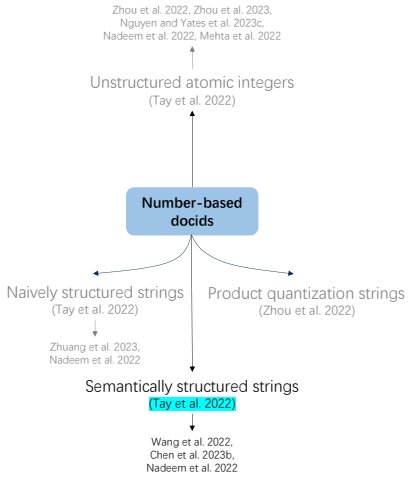
Number-based: Semantically structured strings

- A hierarchical clustering algorithm over document embeddings to induce a decimal tree



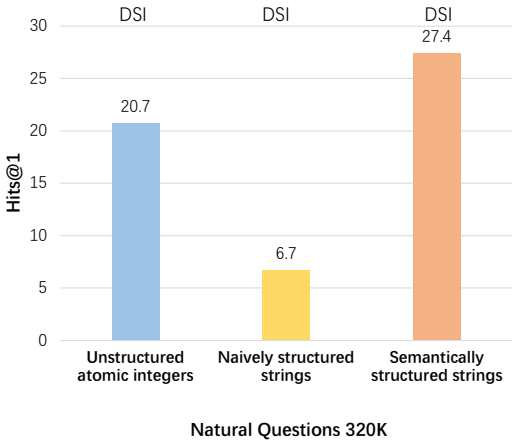
Source: [Tay et al., 2022]

Semantically structured strings and subsequent work



- The document semantics can be incorporated in the decoding process
- It is not limited by the size of the corpus

Performance comparisons [Tay et al., 2022]



Data source: Tay et al. [2022]

- Backbone: T5-base
- Observations: imbuing the docid space with semantic structure can lead to better retrieval capabilities

Number-based docids: Summary

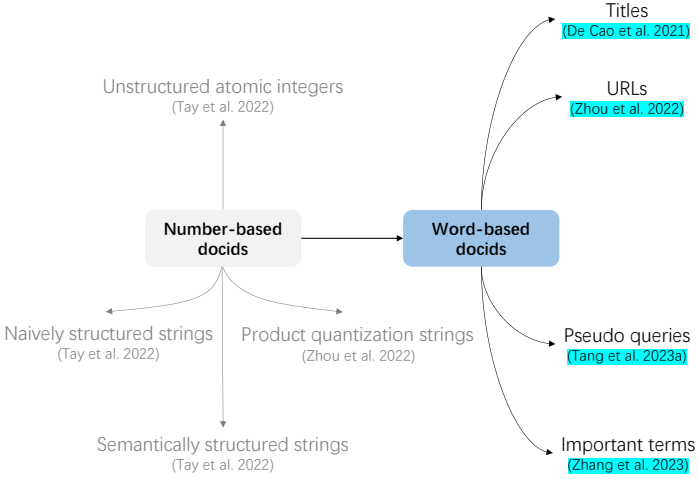
Docids based on integers are easy to build

Unstructured atomic integers and naively/semantically structured strings can maintain uniqueness

They are composed of unreadable numbers

It is challenging to interpret the model's understanding of the corpus

A single docid: Word-based



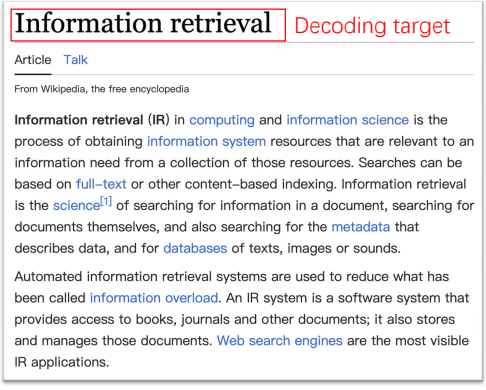
A single docid: Word-based

The fundamental inspiration

- The query is usually keyword-based natural language, which can be challenging to map into a numeric string, while mapping it to words would be more intuitive

Word-based: Titles

- Document titles: be able to summarize the main content



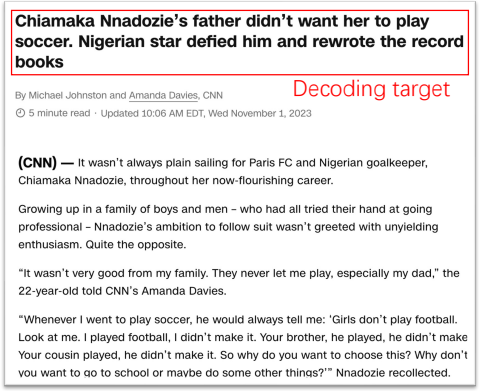


Image source: Wikipedia and CNN

"Autoregressive Entity Retrieval". De Cao et al. [2021]

Word-based: URLs

- The URL of a document contains certain semantic information and can uniquely correspond to this document

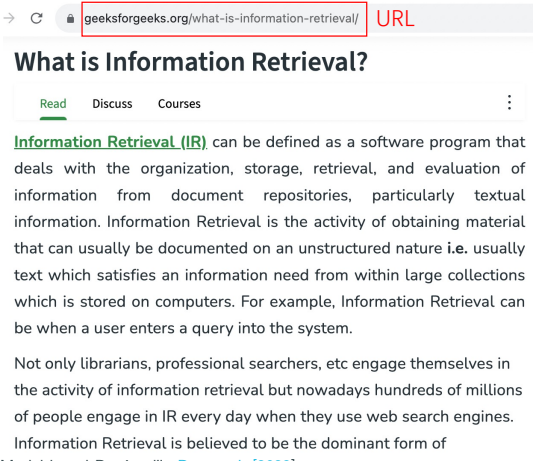


Image source: Geeksforgeeks

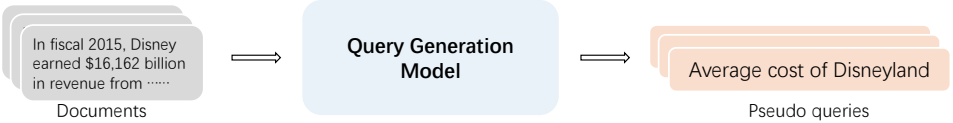
"TOME: A Two-stage Approach for Model-based Retrieval". Ren et al. [2023]

If the special document metadata is not available

It is necessary to design automatic docid generation techniques

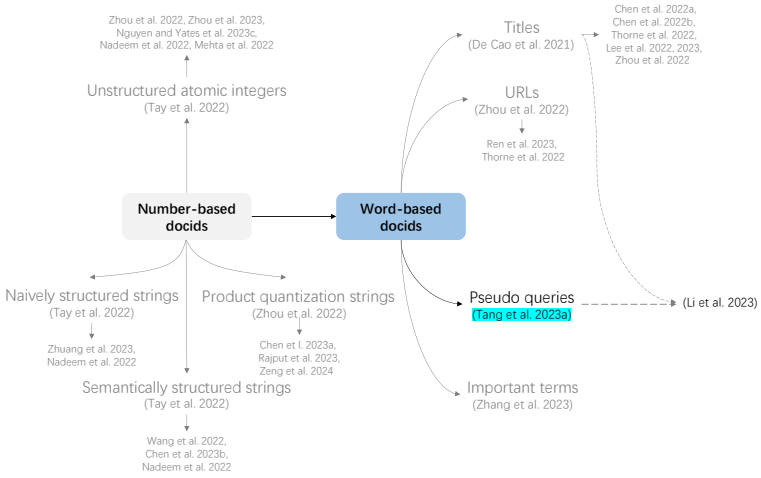
Word-based: Pseudo queries

- Doc2Query technique: pseudo queries are likely to be representative or related to the contents of documents



"Semantic-Enhanced Differentiable Search Index Inspired by Learning Strategies". Tang et al. [2023]

Pseudo queries and subsequent work



Without the requirements of certain document metadata, e.g., titles and URLs

Word-based docids: Summary

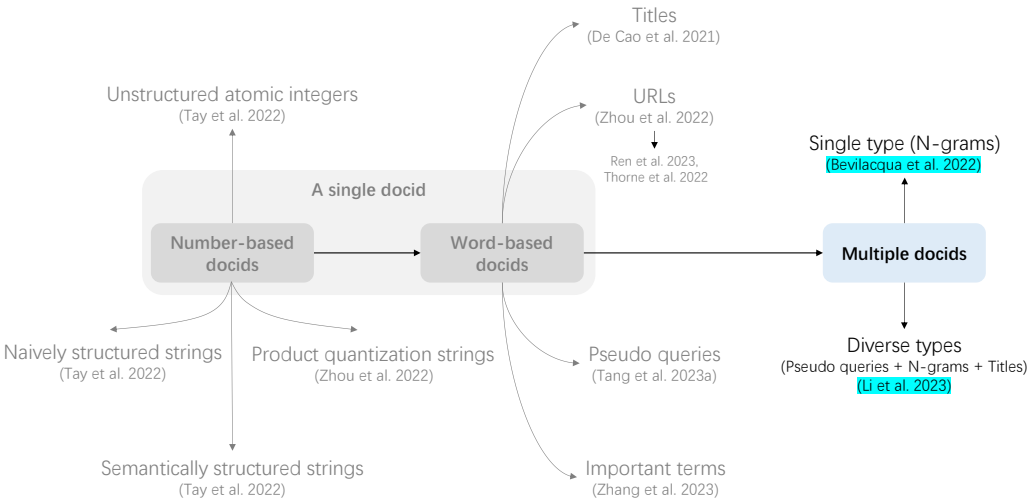
Semantically related to the content of the document

Good interpretability

Rely on metadata or labeled data

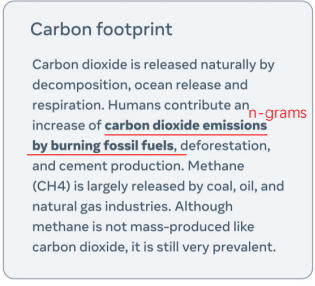
May lead to duplication

Multiple docids



Multiple docids: Single type (N-grams) [Bevilacqua et al., 2022]

- All n-grams (i.e., substrings) in a document are treated as its possible docids
- Part of n-grams as docids during training: Only the terms from the document that have **a high overlap with the query** are chosen as the target docids



"Autoregressive Search Engines: Generating Substrings as Document Identifiers". Bevilacqua et al. [2022]

Multiple docids: Diverse types (MINDER) [Li et al., 2023]

Query: Who is the singer of *does he love you?*

↑**Relevant**

Passage (*https://en.wikipedia.org/wiki/Does_He_Love_You*)

"Does He Love You" is a song written by Sandy Knox and Billy Stritch, and recorded as a duet by American country music artists Reba McEntire and Linda Davis. It was released in August 1993 as the first single from Reba's album "Greatest Hits Volume Two". It is one of country music's several songs about a love triangle. "Does He Love You" was written in 1982 by Billy Stritch. .....

Multiview Identifiers

Title: Does He Love You

Substrings: "Does He Love You" is a song ..., recorded as a duet by American country music artists Reba McEntire and Linda Davis, ...

Pseudo-queries:

Who wrote the song does he love you?

Who sings does he love you?

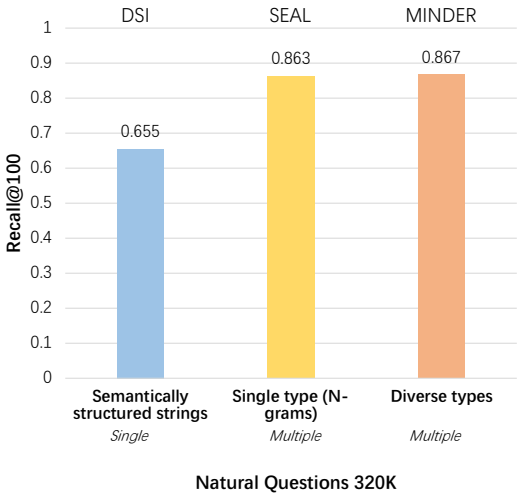
When was does he love you released by reba?

What is the first song in the album "Greatest Hits Volume Two" about?

- Three views of docids
  - Title: Indicate the subject of a document
  - Substrings (N-grams): Be also semantically related
  - Pseudo-queries: Integrate multiple segments and contextualized information

"Multiview Identifiers Enhanced Generative Retrieval". Li et al. [2023]

Performance comparisons



- Backbone: BART-large
- Results: Using multiple docids for a document yields better results than using a single docid

Data source: Li et al. [2023]

Multiple docids: Summary

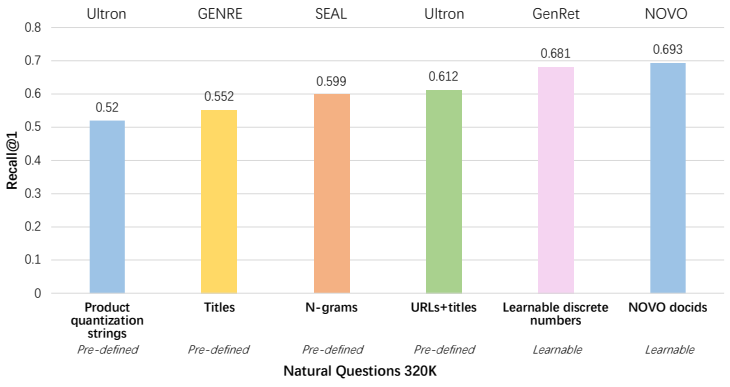
Multiple docids can provide a more comprehensive representation of the document, assisting the model in gaining a multifaceted understanding

Similar docids across different documents can reflect the similarity between the documents

GR models with the increased docid numbers demand more memory usage and inference time

It is challenging to design discriminative multiple docids for a document

Performance comparisons



- Backbone: T5-base
- Results: Two learnable docids yields better results than partial pre-defined static docids

Data source: Wang et al. [2023]

Learnable docids: Summary

It can be optimized together with the ultimate goal of GR to better adapt to retrieval

A learnable approach can enable number-based docids like those in GenRet [Sun et al., 2023] to perform well

It relies on complex task design for learning

Supervised learning: Basic training method

- Learn the indexing task first, and then learn retrieval tasks
  - Step 1: $\mathcal{L}_{Indexing}(D, I_D; \theta) = -\sum_{d \in D} \log P(id \mid d; \theta)$
  - Step 2: $\mathcal{L}_{Retrieval}(Q, I_Q; \theta) = -\sum_{q \in Q} \sum_{id^q \in I_Q} \log P(id^q \mid q; \theta)$
- Learn indexing and retrieval tasks simultaneously in a multitask fashion

$$\begin{aligned}\mathcal{L}_{Global}(Q, D, I_D, I_Q; \theta) &= \mathcal{L}_{Indexing}(D, I_D; \theta) + \mathcal{L}_{Retrieval}(Q, I_Q; \theta) \\ &= -\sum_{d \in D} \log P(id \mid d; \theta) - \sum_{q \in Q} \sum_{id^q \in I_Q} \log P(id^q \mid q; \theta)\end{aligned}$$

"Transformer Memory as a Differentiable Search Index". Tay et al. [2022]

Limitation (1): Single document granularity

$$\mathcal{L}_{Global}(Q, D, I_D, I_Q; \theta) = \mathcal{L}_{Indexing}(D, I_D; \theta) + \mathcal{L}_{Retrieval}(Q, I_Q; \theta)$$
$$= -\sum_{d \in D} \log P(id \mid d; \theta) - \sum_{q \in Q} \sum_{id^q \in I_Q} \log P(id^q \mid q; \theta)$$

When indexing, memorizing each document at a single granularity, e.g., first L tokens or the full text, is insufficient, especially for long documents with rich semantics.

"Semantic-Enhanced Differentiable Search Index Inspired by Learning Strategies". Tang et al. [2023a]

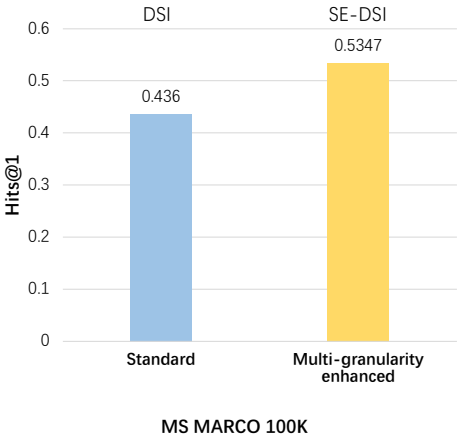
Supervised learning: Multi-granularity enhanced

- Given a document, the important passages p and sentences s are selected to augment the indexing data

$$\mathcal{L}_{Indexing}(D, I_D; \theta) = -(\sum_{d \in D} \log P(id \mid d; \theta) + \sum_{p \in d} \log P(id \mid p; \theta) + \sum_{s \in d} \log P(id \mid s; \theta))$$

"Semantic-Enhanced Differentiable Search Index Inspired by Learning Strategies". Tang et al. [2023a]

Comparisons



Data source: Tang et. al. [2023a]

- Backbone: T5-base
- Multi-granularity representations of documents can comprehensively encode the documents, and further contribute to the retrieval

Limitation (2): The gap between indexing and retrieval

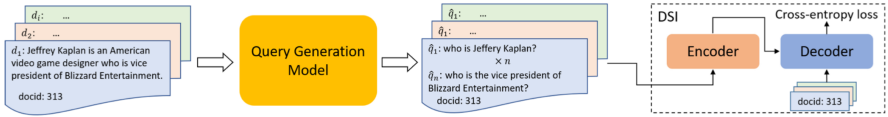
$$\mathcal{L}_{Global}(Q, D, I_D, I_Q; \theta) = \mathcal{L}_{Indexing}(D, I_D; \theta) + \mathcal{L}_{Retrieval}(Q, I_Q; \theta)$$
$$= -\sum_{d \in D} \log P(id \mid d; \theta) - \sum_{q \in Q} \sum_{id^q \in I_Q} \log P(id^q \mid q; \theta)$$

Long document in indexing vs. Short query in retrieval

The data distribution mismatch that occurs between the indexing and retrieval

"Bridging the Gap Between Indexing and Retrieval for Differentiable Search Index with Query Generation". Zhuang et al. [2023]

Supervised learning: Pseudo query enhanced



Using a set of pseudo queries *pq* generated from the document as the inputs of the indexing task

"Bridging the Gap Between Indexing and Retrieval for Differentiable Search Index with Query Generation". Zhuang et al. [2023]

Supervised learning: Pseudo query enhanced

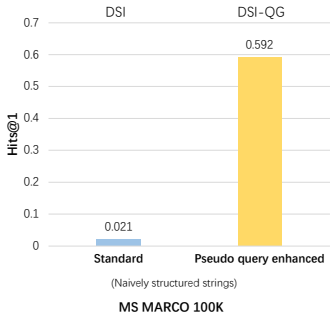
$$\mathcal{L}_{Indexing}(D, I_D; \theta) = -\sum_{d \in D} \log P(id \mid d; \theta)$$

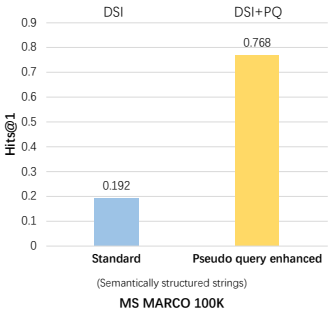
$$\mathcal{L}_{Indexing}(D, I_D; \theta) = -\sum_{pq \in D} \log P(id \mid pq; \theta)$$

$$\mathcal{L}_{Retrieval}(Q, I_Q; \theta) = -\sum_{q \in Q} \sum_{id^q \in I_Q} \log P(id^q \mid q; \theta)$$

"Bridging the Gap Between Indexing and Retrieval for Differentiable Search Index with Query Generation". Zhuang et al. [2023]

Comparisons





- Backbone: T5-base
- Using only pseudo synthetic queries to docid during indexing is an effective training strategy on MS MARCO [Pradeep et al., 2023]

Data source: Pradeep et al. [2023], Zhuang et al. [2023]

Limitation (3): Limited labeled data

$$\mathcal{L}_{Global}(Q, D, I_D, I_Q; \theta) = \mathcal{L}_{Indexing}(D, I_D; \theta) + \mathcal{L}_{Retrieval}(Q, I_Q; \theta)$$

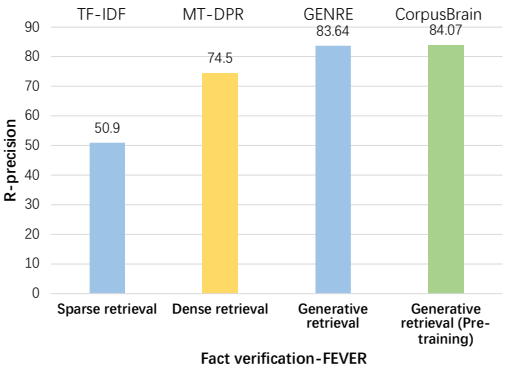
What should we do if there is no or few labeled query-docid pairs?

Pre-training methods

Constructing pseudo query-docid pairs (PQ, I_Q^P) for the pre-training retrieval task

$$\mathcal{L}_{Pre-train}(PQ, D, I_D, I_Q^P; \theta) = \mathcal{L}_{Indexing}(D, I_D; \theta) + \mathcal{L}_{Retrieval}(PQ, I_Q^P; \theta)$$

CorpusBrain [Chen et al., 2022]: Performance



- In the KILT leaderboard, Corpusbrain achieved first place in 5 of them, second place in 1 task, and third place in 4 tasks, outperforming traditional pipelined approaches

"CorpusBrain: Pre-train a Generative Retrieval Model for Knowledge-Intensive Language Tasks". Chen et al. [2022]

Limitation (4): Pointwise optimization for GR

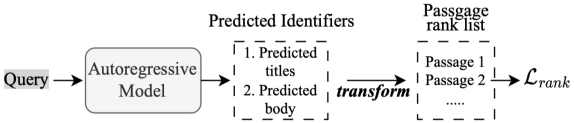
$$\mathcal{L}_{Global}(Q, D, I_D, I_Q; \theta) = \mathcal{L}_{Indexing}(D, I_D; \theta) + \mathcal{L}_{Retrieval}(Q, I_Q; \theta)$$
$$= -\sum_{d \in D} \log P(id \mid d; \theta) - \sum_{q \in Q} \sum_{id^q \in I_Q} \log P(id^q \mid q; \theta)$$

- It assumes the likelihood for each relevant docid is independent of the other docids in the list for a query
- Ranking is a prediction task on list of objects

Pairwise and listwise optimization strategies for GR are necessary!

Pairwise optimization: LTRGR [Li et al., 2023c]

- Step 1: Initial training with pointwise optimization
- Step 2: Based on the trained initial model, perform pairwise optimization

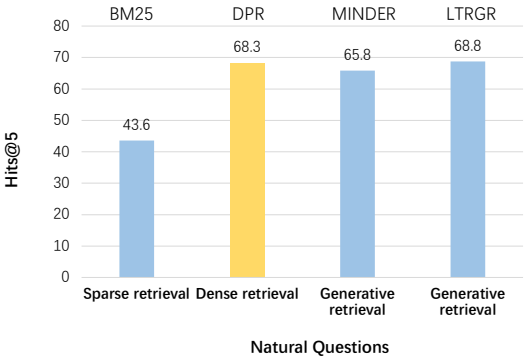


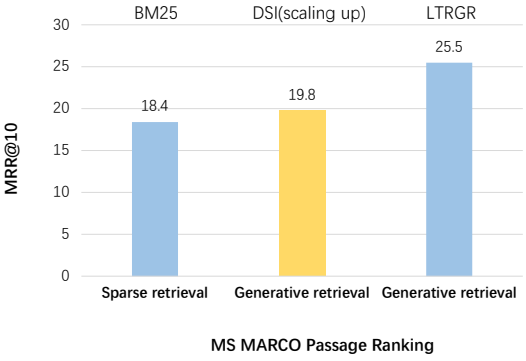
$$max(0, s(q, d_-) - s(q, d_+) + m),$$

where d_- and d_+ are negative and positive documents, and m is the margin

"Learning to Rank in Generative Retrieval". Li et al. [2023c]

LTRGR [Li et al., 2023c]: Performance





"Learning to Rank in Generative Retrieval". Li et al. [2023c]

Roadmap of inference strategies

- A **single identifier** to represent a document:
  - Constrained beam search with a prefix tree
  - Constrained greedy search with the inverted index
- **Multiple identifiers** to represent a document
  - Constrained beam search with the FM-index
  - Scoring functions to aggregate the contributions of several identifiers

Single identifier: Constrained beam search with a prefix tree

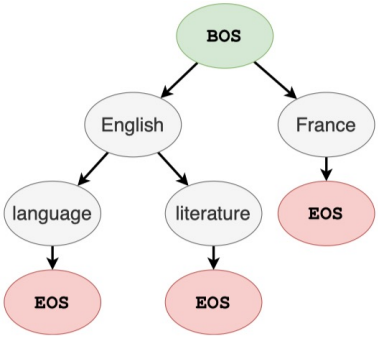
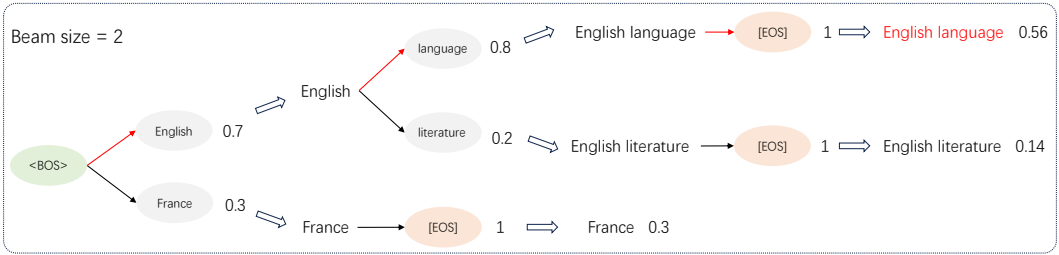


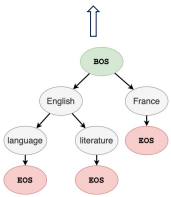
Image source: De Cao et al. [2021]

- For docids considering order of tokens
- **Applicable docids**: Naively structured strings, semantically structured strings, product quantization strings, titles, n-grams, URLs and pseudo queries
- Prefix tree: Nodes are annotated with tokens from the predefined candidate set. For each node, its children indicate all the allowed continuations from the prefix defined traversing the tree from the root to it

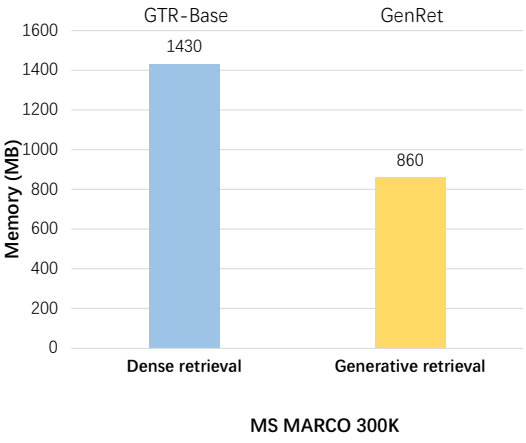
"Autoregressive Entity Retrieval". De Cao et al. [2021]

Example


Beam size = 2
<BOS>
English
0.7
France
0.3
English
language
0.8
literature
0.2
France
English language
[EOS]
1
English language
0.56
English literature
[EOS]
1
English literature
0.14
France
[EOS]
1
France
0.3


BOS
English
France
language
literature
EOS
EOS
EOS

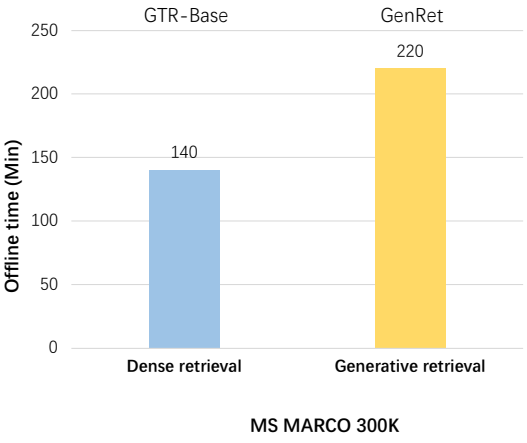
Inference efficiency: Memeory footprint



Data source: Sun et al. [2023]

- The memory footprint of the GR model GenRet is smaller than that of the traditional dense retrieval method GTR, e.g., 1.6 times

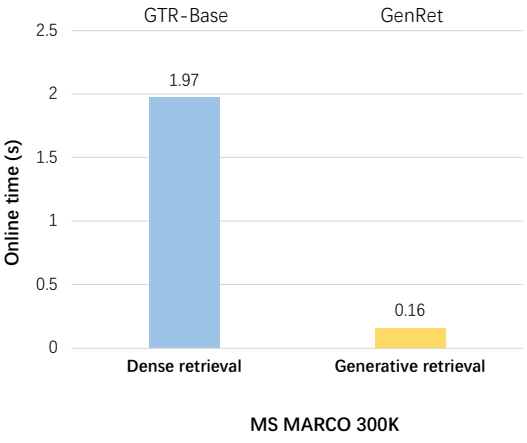
Inference efficiency: Offline latency



Data source: Sun et al. [2023]

- GenRet takes a longer time for offline indexing, as the use of auxiliary models. GTR's offline time consumption comes from document encoding

Inference efficiency: Online latency



- Compared with the traditional dense retrieval model GTR, the GR model GenRet is faster, e.g., 12 times

Data source: Sun et al. [2023]

Pros of generative retrieval

Information retrieval in the era of language models

- Encode the global information in corpus; optimize in an end-to-end way
- The semantic-level association extending beyond mere signal-level matching
- Constraint decoding over thousand-level vocabulary
- Internal index which eliminates large-scale external index

Cons of generative retrieval: Scalability

- Large-scale real-word corpus
  - Current research can generalize from corpora of hundreds of thousands to millions
  - How to accurately memorize vast amounts of complex data?
- Highly dynamic corpora
  - Document addition, removal and updates
  - How to keep such GR models up-to-date?
  - How to learn on new data without forgetting old ones?
- Multi-modal/granularity/language search tasks
  - Different search tasks leverage very different indexes
  - How to unify different search tasks into a single generative form?
  - How to capture task specifications while obtaining the shared knowledge?
- Combining GR with retrieval-augmented generation (RAG)
  - How to integrate GR with RAG to enhance the effectiveness of both?

Cons of generative retrieval: Controllability

For a failure issue, it is often unclear what modeling knobs one should turn to fix the model's behavior

- Interpretability
  - Black-box neural models
  - How to provide credible explanation for the retrieval process and results?
- Debuggable
  - Attribution analysis: how to conduct causal traceability analysis on the causes, key links and other factors of specific search results?
  - Model editing: how to accurately and conveniently modify training data or tune hyperparameters in the loss function?
- Robustness
  - When a new technique enters into the real-world application, it is critical to know not only how it works in average, but also how would it behave in abnormal situations

Cons of generative retrieval: User-centered

Searching is a socially and contextually situated activity with diverse set of goals and needs for support that must not be boiled down to a combination of text matching and text generating algorithms [Shah and Bender, 2022]

- Human information seeking behavior
- Transparency
- Provenance
- Accountability

Source: [Shah and Bender, 2022]

Cons of generative retrieval: Performance

The current performance of GR can only be compared to the index-retrieval stage of traditional methods, and it has not yet achieved the additional improvement provided by re-ranking

Q & A

Thank you for joining us today!

All materials are available at

https://generative-ir.github.io/