



# Generative Information Retrieval

## SIGIR 2024 tutorial – Section 4

---

Yubao Tang<sup>a</sup>, Ruqing Zhang<sup>a</sup>, Zhaochun Ren<sup>b</sup>, Jiafeng Guo<sup>a</sup> and Maarten de Rijke<sup>c</sup>  
<https://generative-ir.github.io/>

July 14, 2024

<sup>a</sup> Institute of Computing Technology, Chinese Academy of Sciences & UCAS

<sup>b</sup> Leiden University

<sup>c</sup> University of Amsterdam

## **Section 4: Training approaches**



## Revisit the definition of generative retrieval

GR usually exploits a Seq2Seq encoder-decoder architecture to generate a ranked list of docids for an input query, in an autoregressive fashion



## Standard training objective

The commonly used training objective for both indexing and retrieval is **maximum likelihood estimation** (MLE):

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

---



## Different learning scenarios based on the corpus

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



## Different learning scenarios based on the corpus

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

- **Stationary scenarios:** The document collection is fixed
- **Dynamic scenarios:** Information changes and new documents emerge incrementally over time



## Stationary scenarios

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

According to the **availability of labeled data**, the training approaches in stationary scenarios can be generally classified into:

- **Supervised learning methods**
- **Pre-training methods**



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



## Supervised learning: Basic training method

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

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



## Supervised learning: Basic training method

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



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



## Limitation (1): Single document granularity

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



## Limitation (1): Single document granularity

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

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.



## Supervised learning: Multi-granularity enhanced

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



## Supervised learning: Multi-granularity enhanced

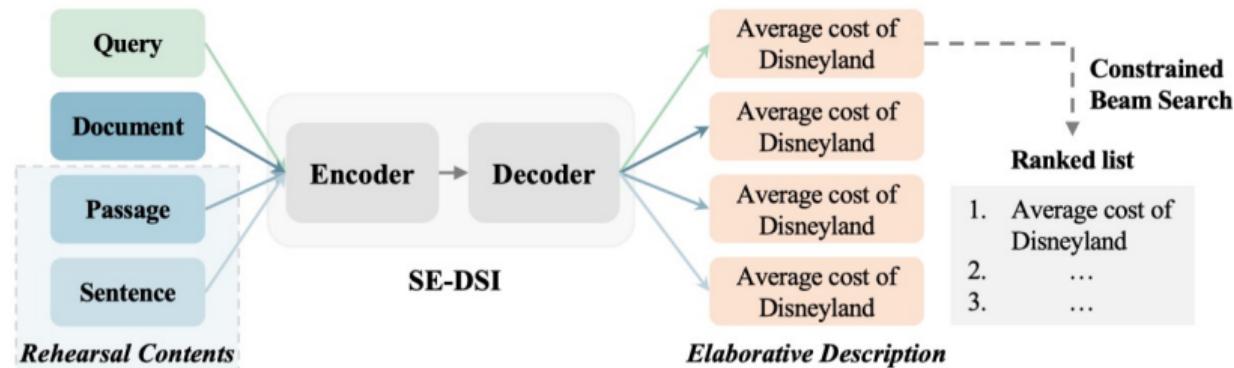
- Given a document, the **important passages  $p$**  and **sentences  $s$**  are selected to augment the indexing data

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



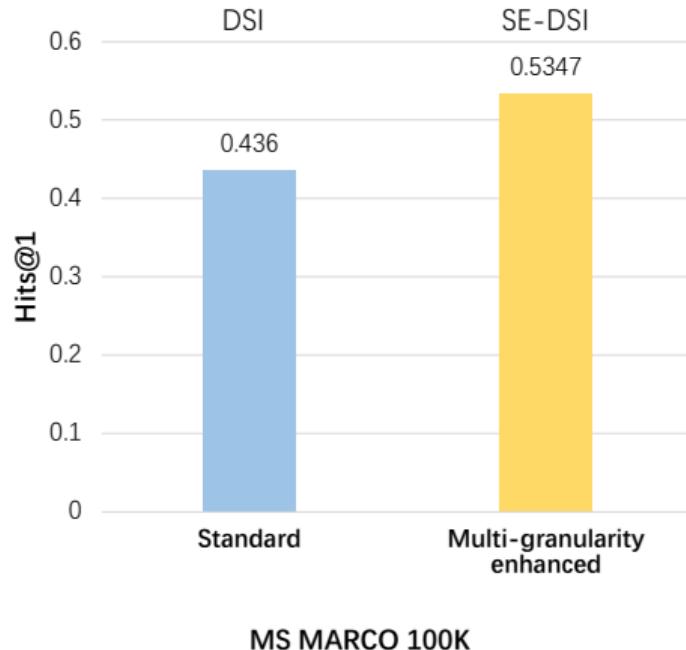
# Supervised learning: Multi-granularity enhanced

- Leading-style: Directly use the leading passages and sentences
- Summarization-style: Leverage the document summarization technique, e.g., TextRank, to highlight important parts



# 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

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



## Limitation (2): The gap between indexing and retrieval

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

Long document in indexing vs. Short query in retrieval



## Limitation (2): The gap between indexing and retrieval

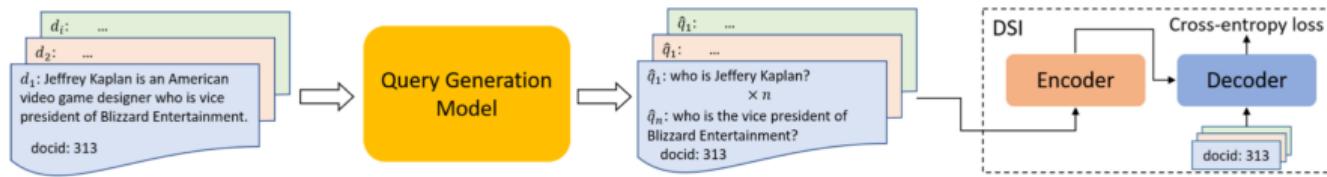
$$\begin{aligned}\mathcal{L}_{Global}(Q, D, I_D, I_Q; \theta) &= \mathcal{L}_{Indexing}(\underline{D}, I_D; \theta) + \mathcal{L}_{Retrieval}(\underline{Q}, I_Q; \theta) \\ &= - \sum_{d \in D} \log P(id \mid \underline{d}; \theta) - \sum_{q \in Q} \sum_{id^q \in I_Q} \log P(id^q \mid \underline{q}; \theta)\end{aligned}$$

Long document in indexing vs. Short query in retrieval

The data distribution mismatch that occurs between the indexing and retrieval



## Supervised learning: Pseudo query enhanced



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



## Supervised learning: Pseudo query enhanced

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


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



## Supervised learning: Pseudo query enhanced

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

↓

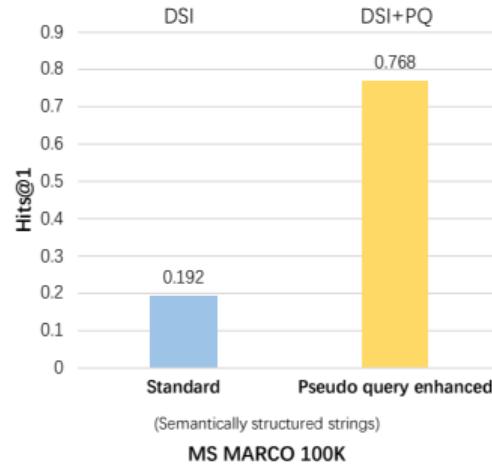
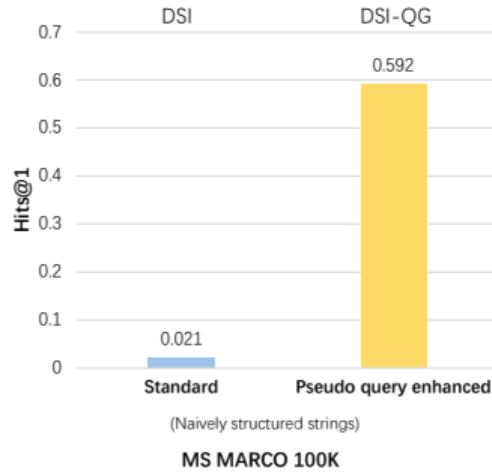
$$\mathcal{L}_{Indexing}(D, I_D; \theta) = - \sum_{pq \in D} \log P(id | \underline{pq}; \theta)$$

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



# Comparisons

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



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



## Limitation (3): Limited labeled data

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



## Limitation (3): Limited labeled data

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

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

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



# CorpusBrain [Chen et al., 2022]: Pre-training

INPUT:  
Apple Inc. is an American multinational  
[...] software and online services.

OUTPUT:  
Apple Inc. ISS

INPUT:  
Apple was founded as Apple Computer  
Company on April 1, 1976 [...] while  
Jobs resigned to found NeXT, taking  
some Apple employees with him.

OUTPUT:  
Apple Inc. [SEP] Tim Cook LPS

## Apple Inc.

Apple Inc. is an American multinational technology company that specializes in consumer electronics, software and online services. Apple is the largest information technology company by revenue [...]

Apple was founded as Apple Computer Company on April 1, 1976, by Steve Jobs, Steve Wozniak and Ronald Wayne to develop and sell Wozniak's Apple I personal computer [...] Apple went public in 1980, to instant financial success. The company developed computers featuring innovative graphical user interfaces, including the original Macintosh, announced in a critically acclaimed advertisement, "1984", directed by Ridley Scott. By 1985, the high cost of its products and power struggles between executives caused problems. Wozniak stepped back from Apple amicably, while Jobs resigned to found NeXT, taking some Apple employees with him.

[...] Apple became the first publicly traded U.S. company to be valued at over \$1 trillion in August 2018, then \$2 trillion in August 2020, and most recently \$3 trillion in January 2022. The company sometimes receives criticism regarding the labor practices of its contractors, its environmental practices, and its business ethics, including anti-competitive practices and materials sourcing. [...]

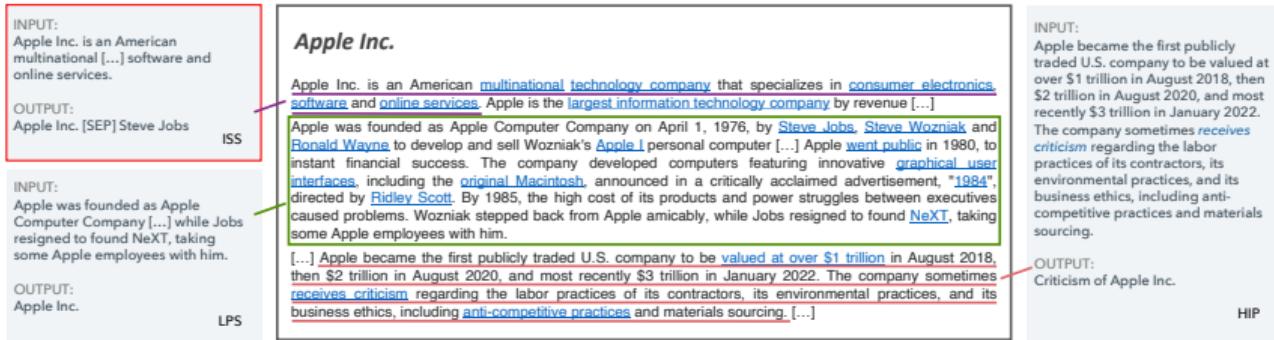
INPUT:  
Apple became the first publicly traded U.S. company to be valued at over \$1 trillion in August 2018, then \$2 trillion in August 2020, and most recently \$3 trillion in January 2022. The company sometimes receives criticism regarding the labor practices of its contractors, its environmental practices, and its business ethics, including anti-competitive practices and materials sourcing.

OUTPUT:  
Criticism of Apple Inc. HIP

Based on Wikipedia, three pre-training retrieval tasks are constructed



# CorpusBrain [Chen et al., 2022]: Pre-training

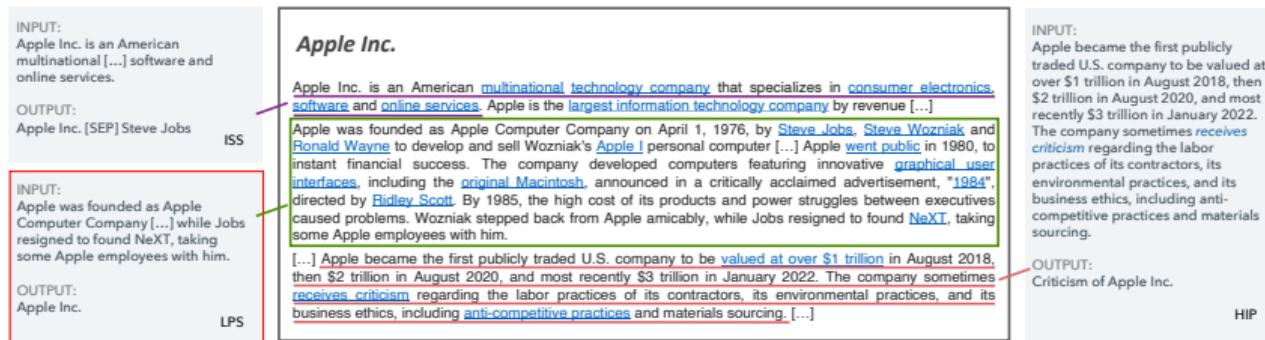


## Inner Sentence Selection (ISS):

- Pseudo query ( $PQ$ ): Randomly selected **inner sentence** from its document
- Docid ( $I_Q^P$ ): Concatenated relevant **document titles**, i.e., “title [SEP] title [SEP] title”



# CorpusBrain [Chen et al., 2022]: Pre-training

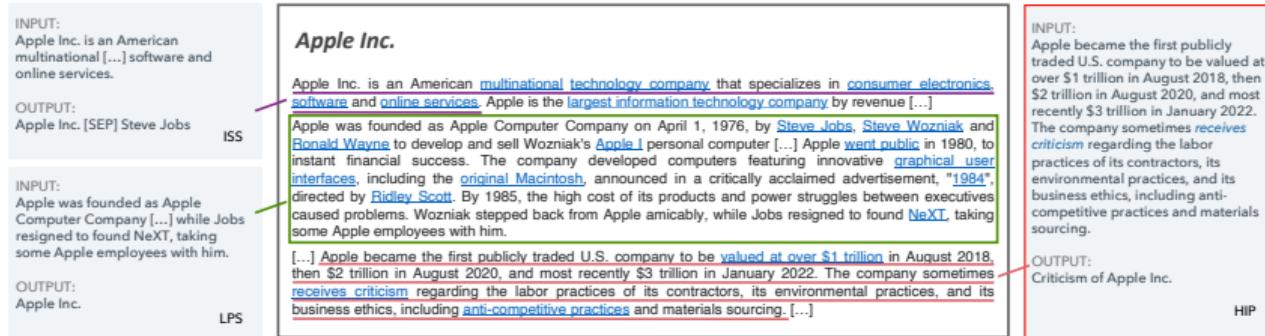


## Lead Paragraph Selection (LPS):

- Pseudo query ( $PQ$ ): A (lead) **paragraph** is sampled from the document
- Docid ( $I_Q^P$ ): Concatenated relevant **document titles**



# CorpusBrain [Chen et al., 2022]: Pre-training



## Hyperlink Identifier Prediction (HIP):

- Pseudo query ( $PQ$ ): The **anchor context**, i.e., the surrounding contextual information in the anchor's corresponding sentence
- Docid ( $I_Q^P$ ): The **document title** of the destination page



## CorpusBrain [Chen et al., 2022]: Training and inference

- **Pre-training:** Based on the three pre-training tasks, a large number of pseudo pairs of query and document identifiers are constructed. All the tasks are formulated by a standard seq2seq objective for the pre-training



## CorpusBrain [Chen et al., 2022]: Training and inference

- **Pre-training:** Based on the three pre-training tasks, a large number of pseudo pairs of query and document identifiers are constructed. All the tasks are formulated by a standard seq2seq objective for the pre-training
- **Fine-tuning:** CorpusBrain is fine-tuned using the processed data (in a Seq2Seq pair format) in downstream tasks

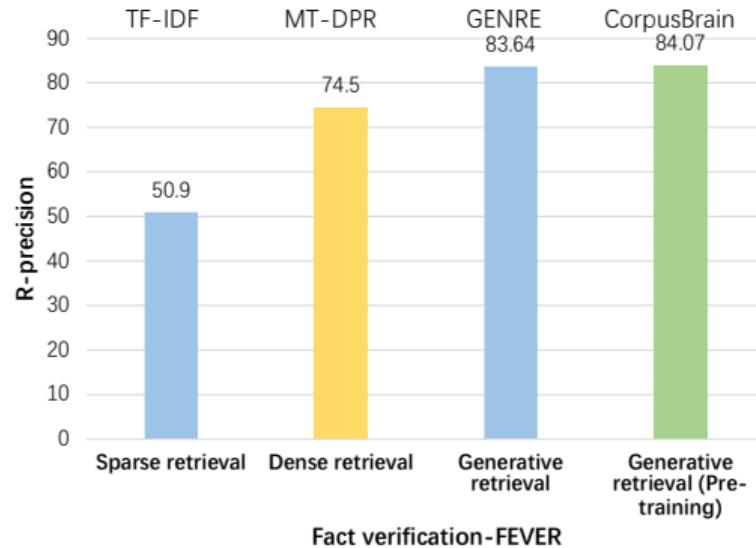


## CorpusBrain [Chen et al., 2022]: Training and inference

- **Pre-training:** Based on the three pre-training tasks, a large number of pseudo pairs of query and document identifiers are constructed. All the tasks are formulated by a standard seq2seq objective for the pre-training
- **Fine-tuning:** CorpusBrain is fine-tuned using the processed data (in a Seq2Seq pair format) in downstream tasks
- **Test:** Given a test query, the fine-tuned CorpusBrain utilizes constrained beam search to decode relevant docids



# 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



## Limitation (4): Pointwise optimization for GR

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

---



## Limitation (4): Pointwise optimization for GR

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

---

- 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**



## Limitation (4): Pointwise optimization for GR

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

---

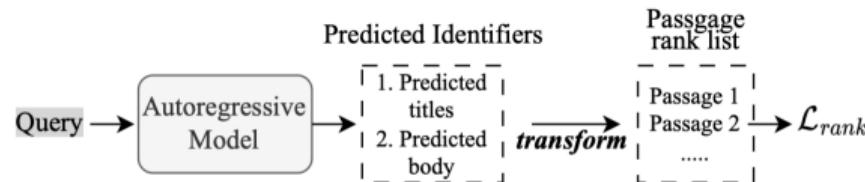
- It assumes the likelihood for each relevant docid is **independent** of the other docids in the list for a query
- Ranking is a prediction task on **list of objects**

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

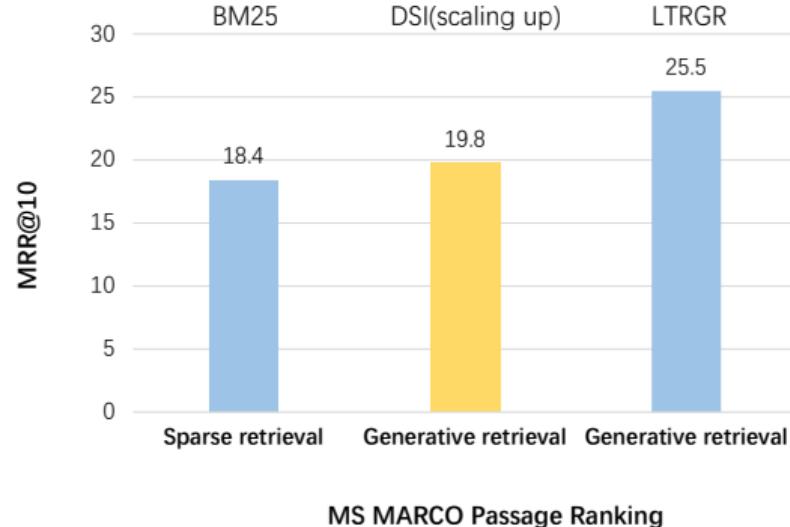
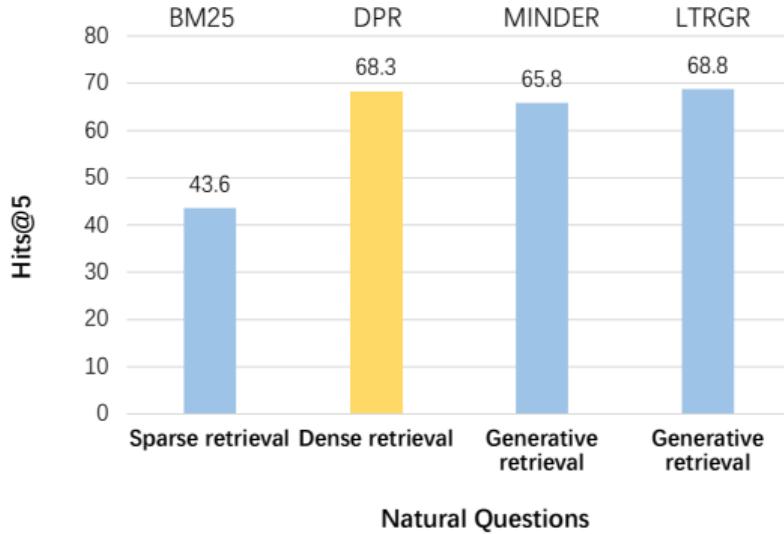


$$\max(0, s(q, d_-) - s(q, d_+) + m),$$

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



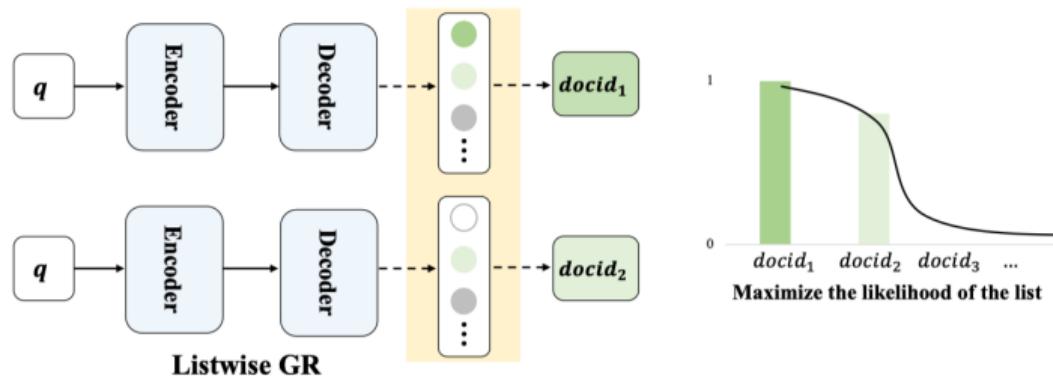
# LTRGR [Li et al., 2023c]: Performance



# Listwise optimization: [Tang et al., 2023b]

Training with position-aware ListMLE

- View the docid ranking problem as a **sequential learning process**, with each step targeting to maximize the corresponding stepwise probability distribution



## Listwise optimization: [Tang et al., 2023b]

Given:

- A query  $q$
- Its ground-truth docid list  $\pi_q = [id^{(1)}, id^{(2)}, \dots]$ , in descending order of relevance, where  $id^{(1)}$  is the docid ranked at the first position, and  $id^{(2)}$  is the docid ranked at the second position, and so on



## Sequential learning process

**Step 1:** Maximize the following top-1 positional conditional probability:

$$P(id^{(1)} | q; \theta) = \frac{\exp(\tilde{P}(id^{(1)} | q; \theta))}{\sum_{j=1}^n \exp(\tilde{P}(id^{(j)} | q; \theta))},$$

where  $\tilde{P}(id^{(i)} | q; \theta) = \frac{\log \prod_{t \in [1, |id^{(i)}|]} P(w_t | q, w_{<t}; \theta)}{|id^{(i)}|}$  (without considering the ranking order information), and  $P(id^{(i)} | q; \theta)$  is the generated likelihood of the  $i$ -th relevant docid  $id^{(i)}$  for  $q$



## Sequential learning process

**Step 2:** For  $i = 2, \dots, n$ , maximize the following  $i$ -th positional conditional probability given the preceding top  $i - 1$  docids,

$$P(id^{(i)} | q, id^{(1)}, \dots, id^{(i-1)}; \theta) = \frac{\exp(\tilde{P}(id^{(i)} | q; \theta))}{\sum_{j=i}^n \exp(\tilde{P}(id^{(j)} | q; \theta))}$$

The learning process ends at step  $n + 1$



## Listwise loss with position importance

- Listwise probability with position importance

$$\min_{\theta} - \log P(\pi_q | q; \theta)$$

$$= -\alpha(1) \log P(id^{(1)} | q; \theta) - \sum_{i=2}^n \alpha(i) \log P(id^{(i)} | q, id^{(1)}, \dots, id^{(i-1)}; \theta),$$

where the weight  $\alpha(\cdot)$  is a decreasing function

- Listwise loss function incorporating the probability based on Plackett-Luce model

$$\mathcal{L}_{List}(q, \pi_q; \theta) = \sum_{i=1}^n \alpha(i) \left( -\tilde{P}(id^{(i)} | q; \theta) + \log \left( \sum_{k=i}^n \exp(\tilde{P}(id^{(k)} | q; \theta)) \right) \right)$$



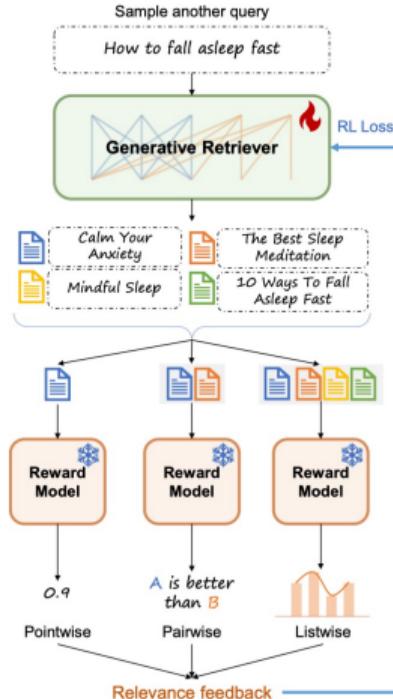
## Multiple optimization: GenRRL [Zhou et al., 2023]

Based on reinforce learning framework

- train a linear reward model
- train a GR model with **pointwise, pairwise and listwise** optimization strategies



# Multiple optimization: GenRRL [Zhou et al., 2023]

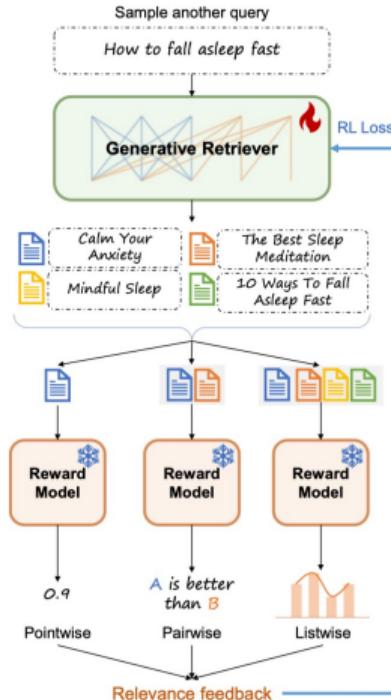


- Pointwise optimization:

$-\sum_i(R(q, id_i) - b) \sum_t \log P(w_t^i | w_{<t}, q)$ ,  
where  $R$  is a reward model, and  $b$  is a baseline



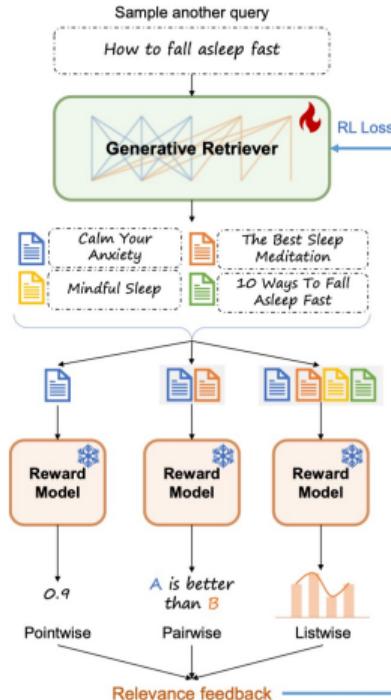
# Multiple optimization: GenRRL [Zhou et al., 2023]



- Pointwise optimization:  
–  $\sum_i (R(q, id_i) - b) \sum_t \log P(w_t^i | w_{<t}, q)$ ,  
where  $R$  is a reward model, and  $b$  is a baseline
- Pairwise optimization:  
–  $\sum_{(id_i, id_j)} (R(q, id_i) \log p_{ij} + R(q, id_j) \log p_{ji})$ ,  
where  $p_{ij} = |P(w_t^i | q) - P(w_t^j | q)|$



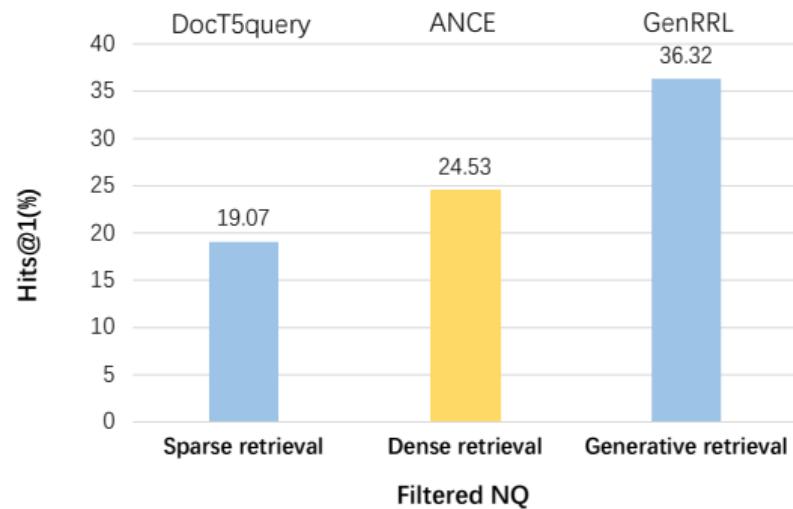
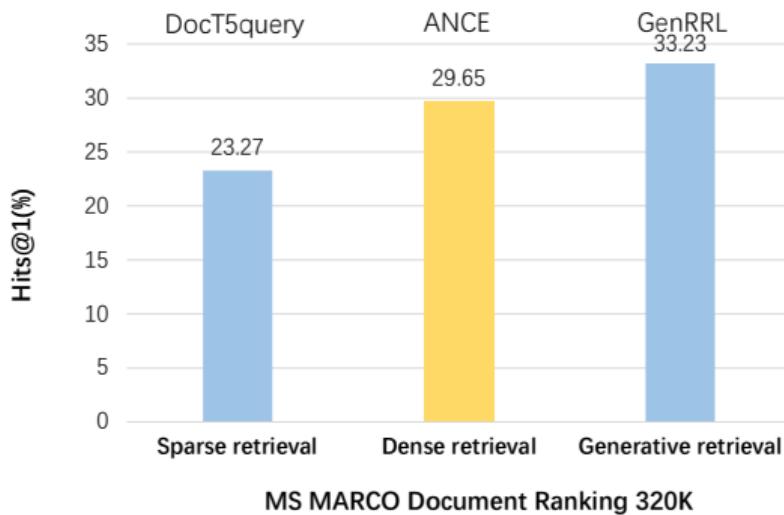
# Multiple optimization: GenRRL [Zhou et al., 2023]



- Pointwise optimization:  
–  $\sum_i (R(q, id_i) - b) \sum_t \log P(w_t^i | w_{<t}, q)$ ,  
where  $R$  is a reward model, and  $b$  is a baseline
- Pairwise optimization:  
–  $\sum_{(id_i, id_j)} (R(q, id_i) \log p_{ij} + R(q, id_j) \log p_{ji})$ ,  
where  $p_{ij} = |P(w_t^i | q) - P(w_t^j | q)|$
- Listwise optimization:  
–  $\sum_{id_i \in C} R(q, id_i) \log \frac{\exp(P(id_i|q))}{\sum_j \exp(P(id_j|q))}$



# GenRRL [Zhou et al., 2023]: Performance



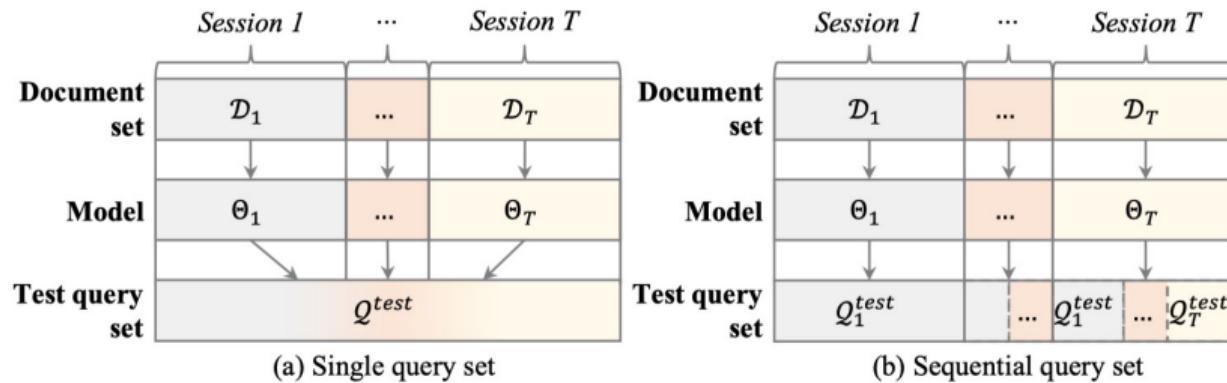
## Dynamic scenarios

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

Information changes and new documents emerge incrementally over time



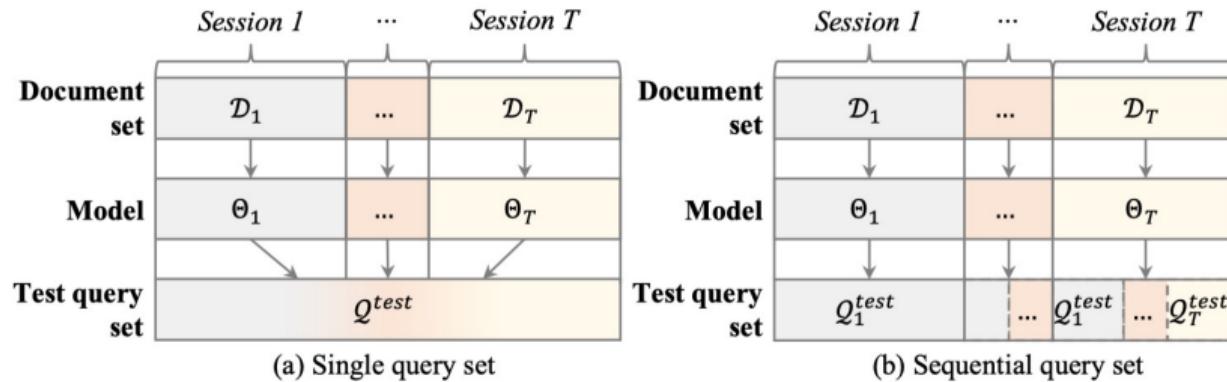
# Continual learning task: Formulation



- **Initial model:** A large-scale base document set  $D_0$  and sufficiently many labeled query-document pairs



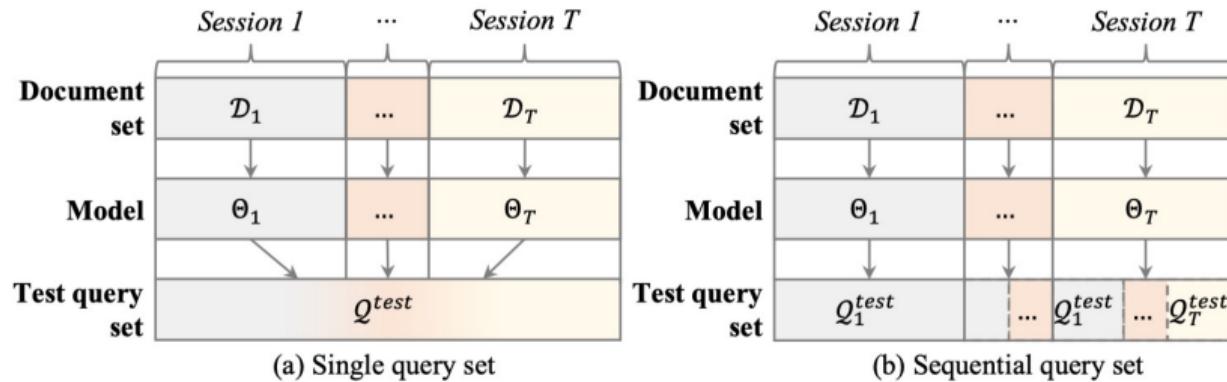
# Continual learning task: Formulation



- **Initial model:** A large-scale base document set  $D_0$  and sufficiently many labeled query-document pairs
- **New datasets:**  $T$  new datasets  $D_1, \dots, D_T$ , from  $T$  sessions arriving in a sequential manner, which are only composed of newly encountered documents without queries related to these documents



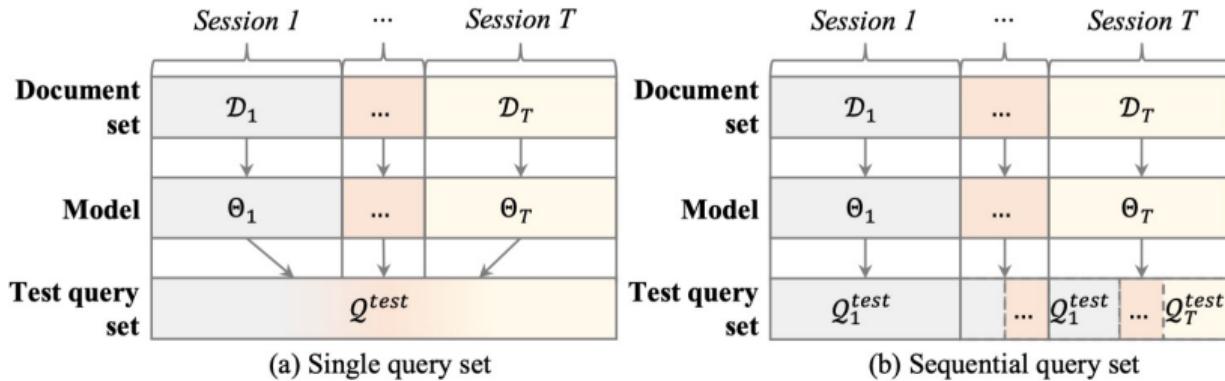
# Continual learning task: Formulation



- **Initial model:** A large-scale base document set  $D_0$  and sufficiently many labeled query-document pairs
- **New datasets:**  $T$  new datasets  $D_1, \dots, D_T$ , from  $T$  sessions arriving in a sequential manner, which are only composed of newly encountered documents without queries related to these documents
- **Model update:** The new dataset  $D_t$  and previous datasets  $D_0, \dots, D_{t-1}$



# Continual learning task: Evaluation

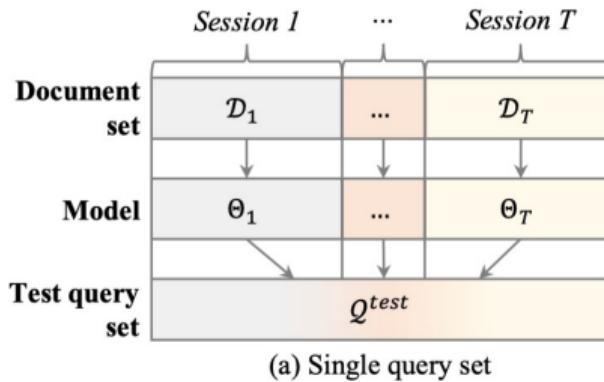


Two types of test query set for performance evaluation:

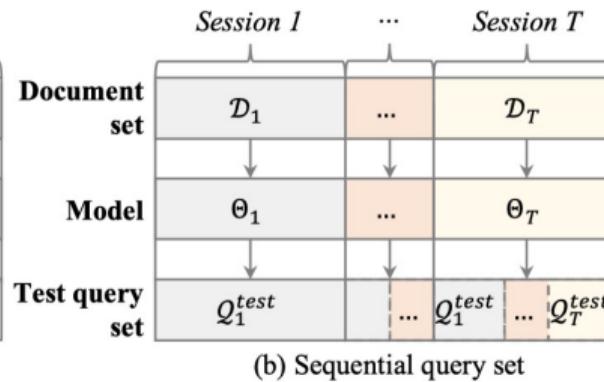
- **Single query set:** There is only one test query set, and their relevant documents arrive in different sessions
- **Sequential query set:** The test query set is specific for each session, and the relevant documents appear in existing sessions



# Continual learning task: Evaluation



(a) Single query set



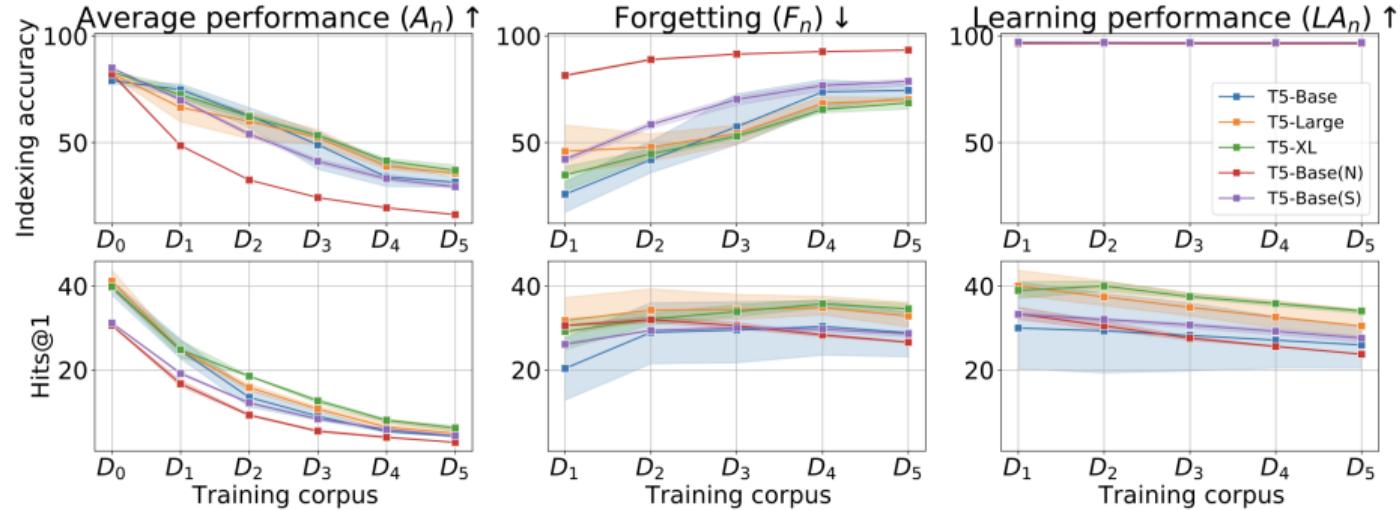
(b) Sequential query set

Two types of test query set for performance evaluation:

- **Single query set:** There is only one test query set, and their relevant documents arrive in different sessions
- **Sequential query set:** The test query set is specific for each session, and the relevant documents appear in existing sessions



# Catastrophic forgetting



The GR model undergoes **severe forgetting under continual indexing of new documents**



## Challenges of continual learning for GR

- How to incrementally index new documents with low computational and memory costs?



## Challenges of continual learning for GR

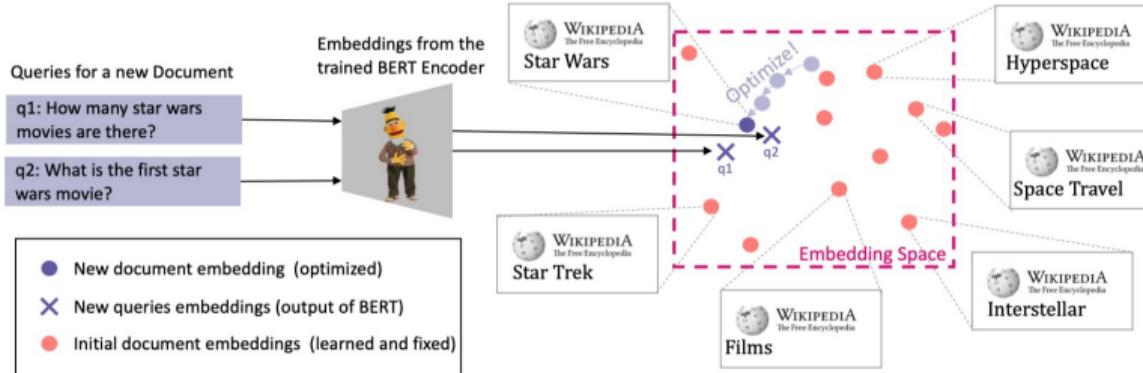
- How to incrementally index new documents with low computational and memory costs?
- How to prevent catastrophic forgetting for previously indexed documents and maintain the retrieval ability?



- Docid: unique atomic integers
- Constrained optimization problem: find the optimal document vector for a new document, do not modify any other existing document vectors and do not require broader updates to the query encoder



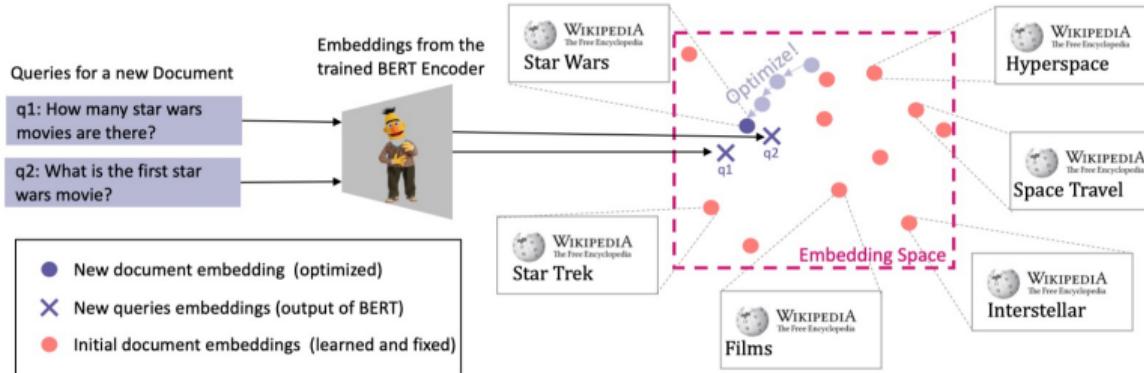
# IncDSI [Kishore et al., 2023]: Incrementally indexing new documents



- Constrained optimization:
  - The new document is scored higher than all the existing documents for its representative query embedding



# IncDSI [Kishore et al., 2023]: Incrementally indexing new documents



- Constrained optimization:
  - The new document is scored higher than all the existing documents for its representative query embedding
  - The new document is scored lower than all the existing documents for other representative query embedding



## DSI++ [Mehta et al., 2022]: Incrementally indexing new documents

- Docids: The new documents are assigned **unstructured atomic integers** as docids, and the GR model learns new embeddings for each of them

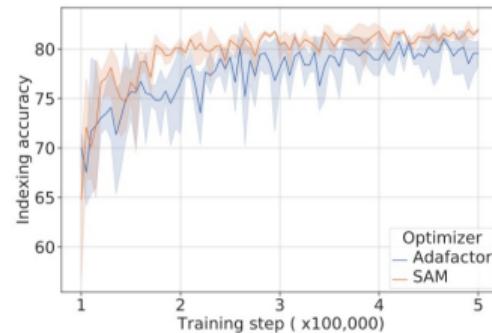


## DSI++ [Mehta et al., 2022]: Incrementally indexing new documents

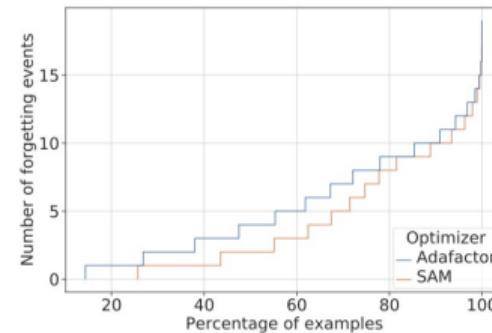
- Docids: The new documents are assigned **unstructured atomic integers** as docids, and the GR model learns new embeddings for each of them
- **Modifying the training dynamics:** Since flatter minima implicitly alleviate forgetting, optimizing for flatter loss basins using Sharpness-Aware Minimization (SAM) as an objective allows the model to stably memorize more documents



# DSI++ [Mehta et al., 2022]: Incrementally indexing new documents



(a) Indexing accuracy during memorization



(b) Cumulative histogram of forgetting events

- SAM **outperforms Adafactor** in terms of the overall indexing accuracy
- SAM **undergoes less severe fluctuations** during the course of training



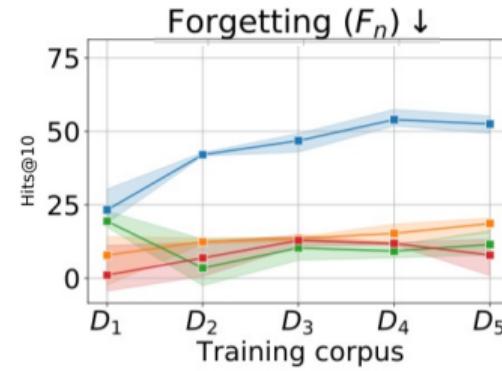
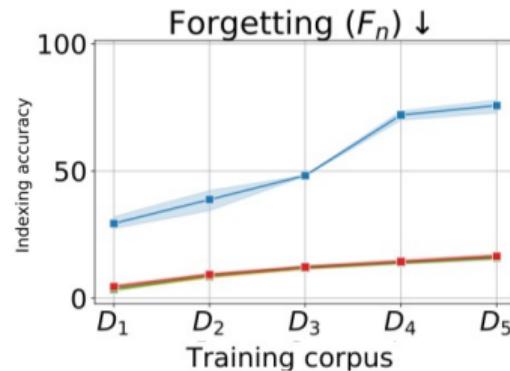
## DSI++ [Mehta et al., 2022]: Preventing catastrophic forgetting

- **Generative memory:** Train a query generator model to sample pseudo-queries for previously seen documents and supplement the query-docid pairs during continual indexing



# DSI++ [Mehta et al., 2022]: Preventing catastrophic forgetting

- **Generative memory:** Train a query generator model to sample pseudo-queries for previously seen documents and supplement the query-docid pairs during continual indexing
- It **reduces the forgetting**, and **improves average Hits@10 by +21.1%** over baselines



## Limitations of DSI++

- Learning embeddings for each individual new docid from scratch incurs prohibitively **high computational costs**



## Limitations of DSI++

- Learning embeddings for each individual new docid from scratch incurs prohibitively **high computational costs**
- The relationships between new and old documents may not be easily obtained from **randomly-selected exemplars**



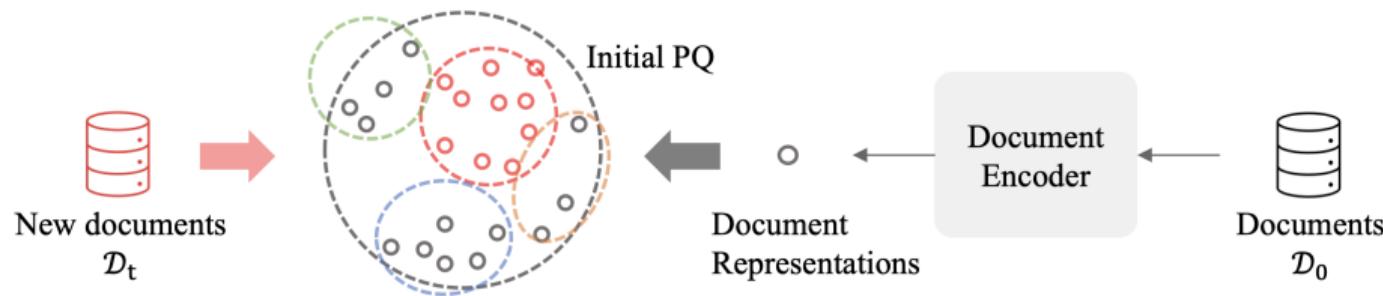
## CLEVER [Chen et al., 2023]: Incrementally indexing new documents

Incremental product quantization (PQ) codes as identifiers: Update a partial quantization codebook according to two adaptive thresholds



## CLEVER [Chen et al., 2023]: Incrementally indexing new documents

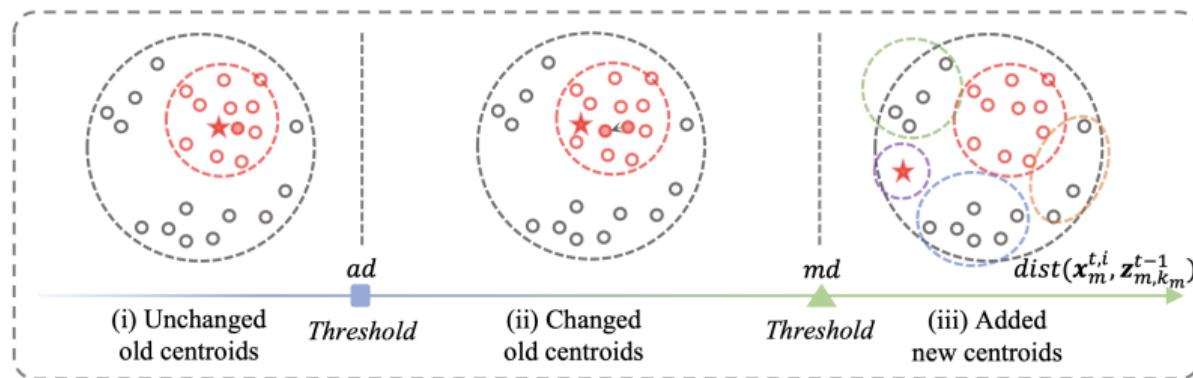
Incremental product quantization (PQ) codes as identifiers: Update a partial quantization codebook according to two adaptive thresholds



- Build base PQ
  - Centroids are obtained via clustering over document representations
  - Document representations are learned with a bootstrapped training process



## CLEVER [Chen et al., 2023]: Incremental product quantization



- Update adaptively
  - Dynamic thresholds: Average distance ( $ad$ ); maximum distance ( $md$ )
  - Three types of update for centroid representation: Depend on contributions to centroid update



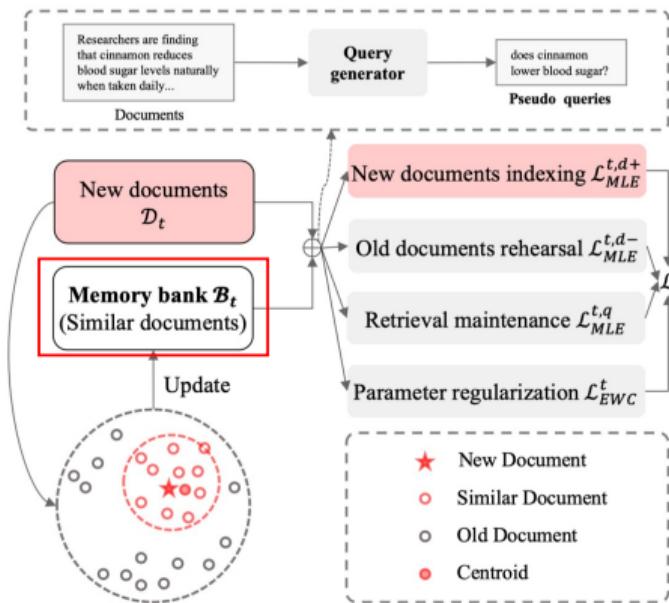
## CLEVER [Chen et al., 2023]: Preventing catastrophic forgetting

Memory-augmented learning mechanism: Form **meaningful connections** between old and new documents



# CLEVER [Chen et al., 2023]: Preventing catastrophic forgetting

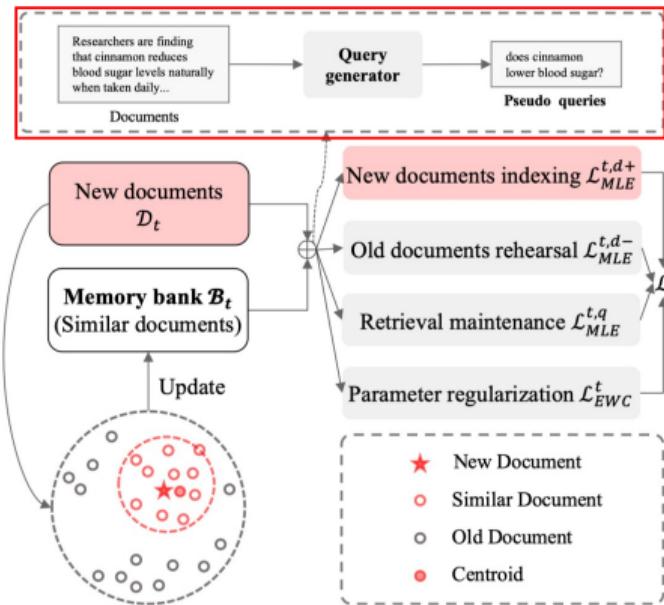
Memory-augmented learning mechanism: Form meaningful connections between old and new documents



- **Dynamic memory bank:** Construct a memory bank with **similar documents** for each new session and replay the process of indexing them alongside the indexing of new documents



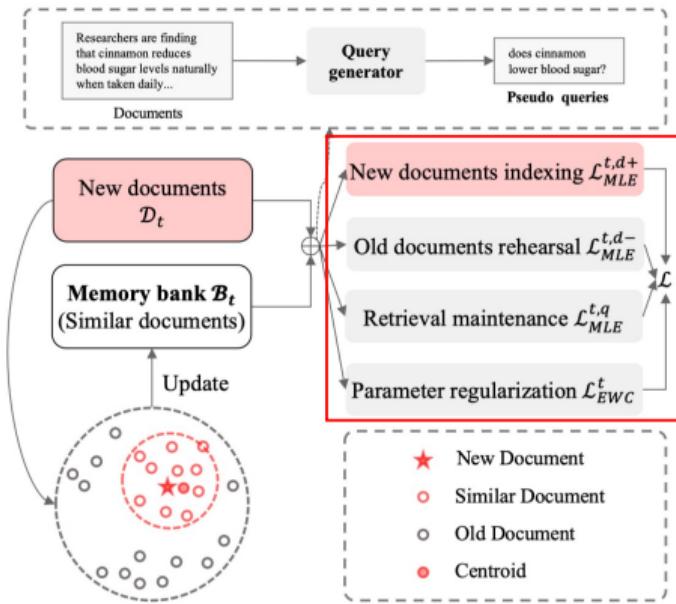
# CLEVER [Chen et al., 2023]: Memory-augmented learning mechanism



- **Pseudo query-docid pairs:** Train a query generator model to sample pseudo-queries for documents and **supplement the query-docid pairs** during indexing



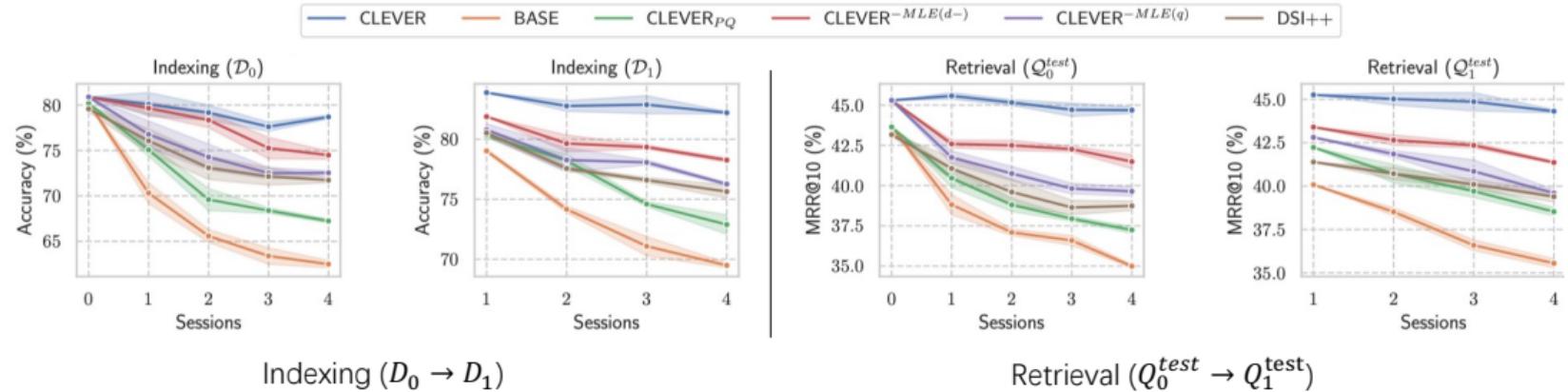
# CLEVER [Chen et al., 2023]: Memory-augmented learning mechanism



- **Sequentially training:** new documents indexing, old document rehearsal, retrieval maintenance losses and an **elastic weight consolidation** (EWC) loss as a regularization term



# CLEVER [Chen et al., 2023]: Performance



- CLEVER almost avoids catastrophic forgetting on both indexing and retrieval tasks, showing its effectiveness in a dynamic setting



## Combination of GR and retrieval-augmented generation (RAG)

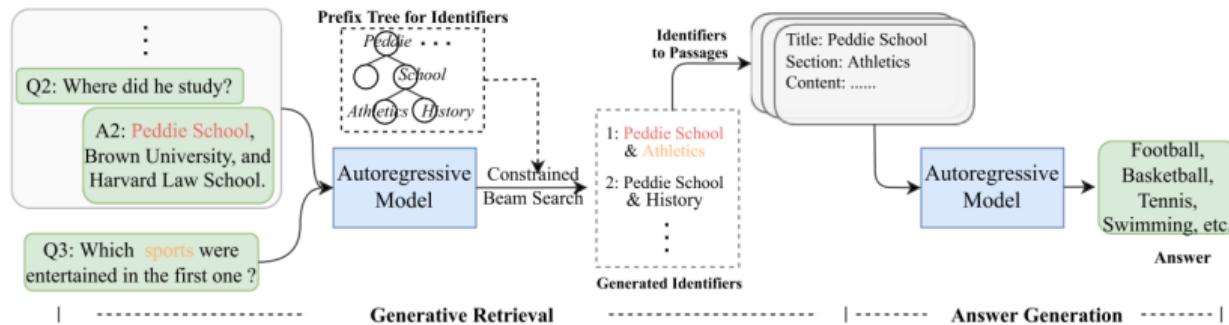
How to jointly train the GR model and QA model?

$$\mathcal{L}_{QA}(Q^*, I_D^*, D^*, A; \psi) = - \sum_{q^* \in Q^*, id \in I_D, d \in D, a \in A} \log f(a|q^*, id, d; \psi),$$

where  $Q^*$  is the query set of the downstream task,  $I_D^*$  are the docids retrieved by a GR model,  $D^*$  are the corresponding documents,  $a$  is an answer in the answer set  $A$ ,  $f$  is the QA function and  $\psi$  is the model parameters



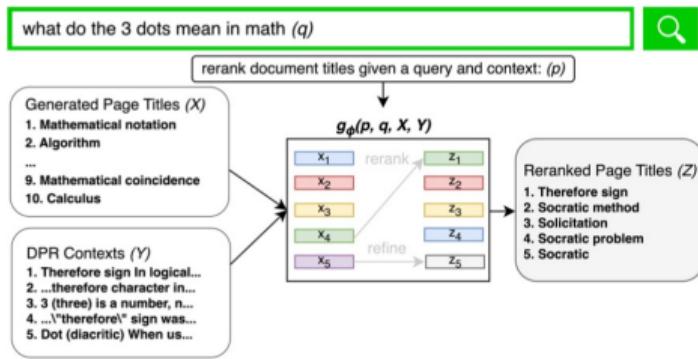
# GCoQA [Li et al., 2023b]



- Step 1: **Document retrieval** with a GR model
- Step 2: **Answer generation** with another autoregressive model



# Re3val [Song et al., 2024]



- Step 1: Relevant titles generation using a GR model
- Step 2: Retrieved titles **reranking** using a cross-encoder
- Step 3: **Context retrieval** for titles using BM25
- Step 4: Answer generation using an generative model

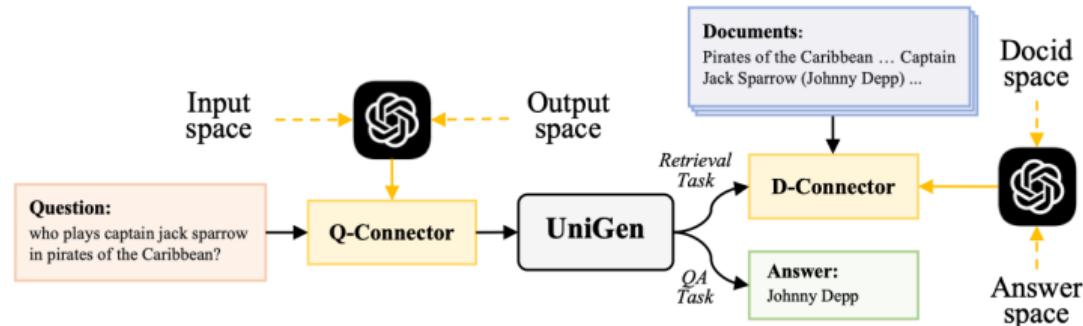


## Limitation

Generative document retrieval and grounded answer generation rely on **separate retrieval and reader module**, which may hinder simultaneous optimization



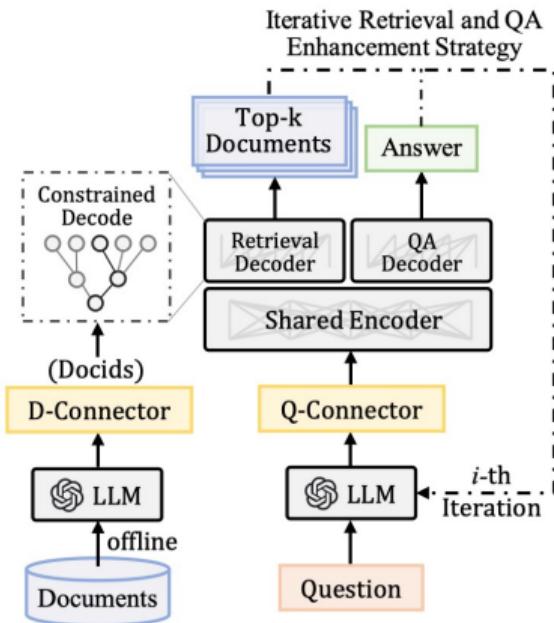
# UniGen [Li et al., 2023a]



- Joint learning for GR and QA



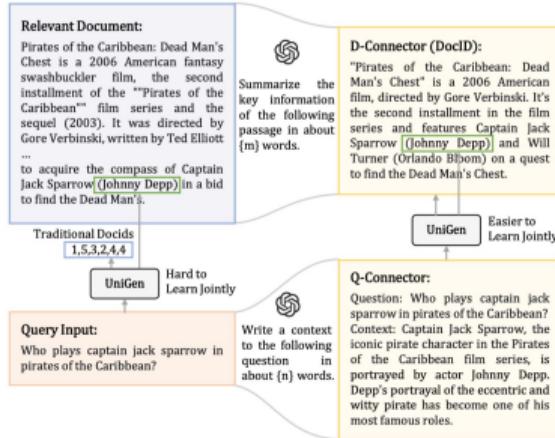
# UniGen [Li et al., 2023a]: Architecture



- A shared encoder and two distinct decoders for GR and QA



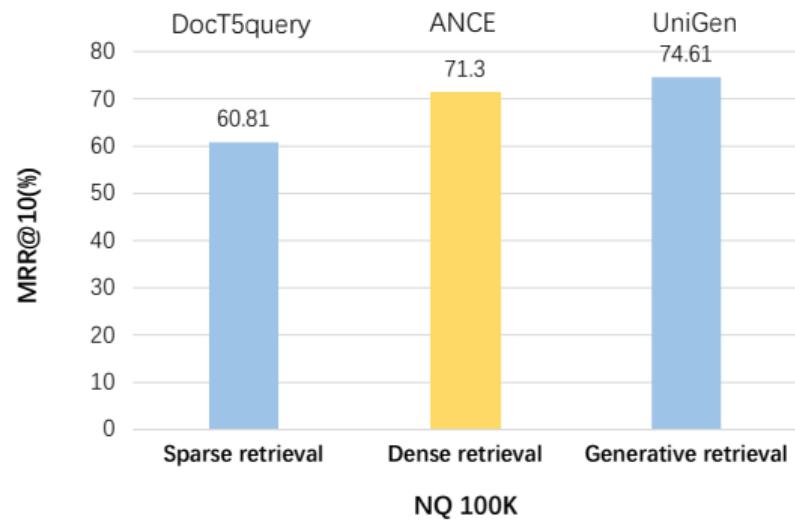
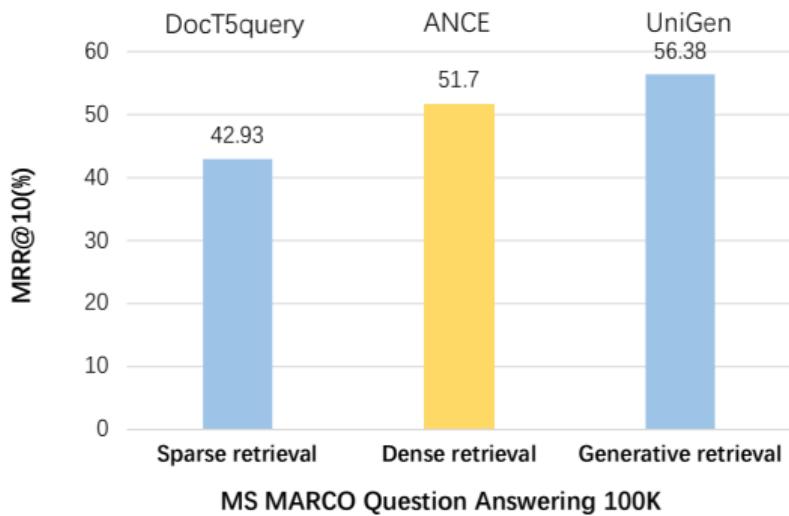
# UniGen [Li et al., 2023a]



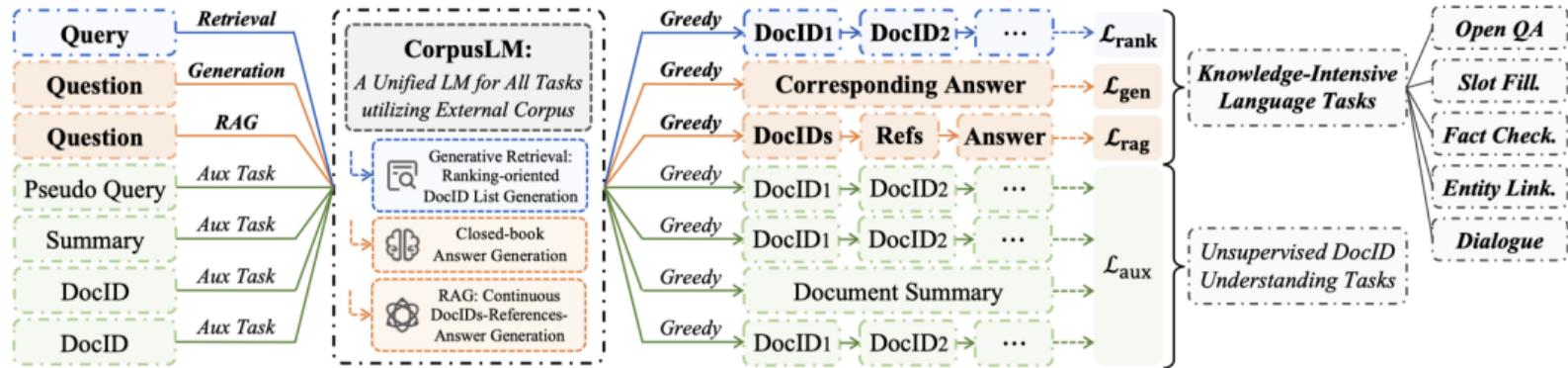
- Use LLMs to generate a query context and document summary, serving as bridges between query inputs, documents, and answer outputs



# UniGen [Li et al., 2023a]: Performance



# CorpusLM [Li et al., 2024]



- a **unified language model** that leverages external corpus to tackle various knowledge-intensive tasks by integrating GR, closed-book generation, and RAG through a unified greedy decoding process



## Limitations in large-scale corpus

- Existing GR models only perform well on artificially-constructed and **small-scale collections**
- Zeng et al. [2024a] and Zeng et al. [2024b] introduced RIPOR and PAG, designed to improve the performance of GR models for MS MARCO dataset, with 8.8M passages.

**It is necessary to explore the capacity of GR models to larger corpus**



## Revisit: Challenges of training approaches

- How to memorize the whole corpus effectively and efficiently?



## Revisit: Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Multi-granularity enhanced document content
  - Pre-training
  - Listwise optimization



## Revisit: Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Multi-granularity enhanced document content
  - Pre-training
  - Listwise optimization
- **How to learn heterogeneous tasks well within a single model?**



## Revisit: Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Multi-granularity enhanced document content
  - Pre-training
  - Listwise optimization
- **How to learn heterogeneous tasks well within a single model?**
  - Pseudo query enhanced input



## Revisit: Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Multi-granularity enhanced document content
  - Pre-training
  - Listwise optimization
- **How to learn heterogeneous tasks well within a single model?**
  - Pseudo query enhanced input
- **How to handle a dynamically evolving document collection?**



## Revisit: Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Multi-granularity enhanced document content
  - Pre-training
  - Listwise optimization
- **How to learn heterogeneous tasks well within a single model?**
  - Pseudo query enhanced input
- **How to handle a dynamically evolving document collection?**
  - Low computational and memory costs
  - Maintaining the retrieval ability



## References

## References i

- J. Chen, R. Zhang, J. Guo, Y. Liu, Y. Fan, and X. Cheng. Corpusbrain: Pre-train a generative retrieval model for knowledge-intensive language tasks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 191–200, 2022.
- J. Chen, R. Zhang, J. Guo, M. de Rijke, W. Chen, Y. Fan, and X. Cheng. Continual learning for generative retrieval over dynamic corpora. In *Proceedings of the 32nd ACM Conference on Information and Knowledge Management*, 2023.
- V. Kishore, C. Wan, J. Lovelace, Y. Artzi, and K. Q. Weinberger. Incdsi: incrementally updatable document retrieval. In *International Conference on Machine Learning*, pages 17122–17134. PMLR, 2023.
- X. Li, Y. Zhou, and Z. Dou. Unigen: A unified generative framework for retrieval and question answering with large language models. In *AAAI Conference on Artificial Intelligence*, 2023a.
- X. Li, Z. Dou, Y. Zhou, and F. Liu. Corpuslm: Towards a unified language model on corpus for knowledge-intensive tasks. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*, 2024.



## References ii

- Y. Li, N. Yang, L. Wang, F. Wei, and W. Li. Generative retrieval for conversational question answering. *Information Processing & Management*, 60(5):103475, 2023b.
- Y. Li, N. Yang, L. Wang, F. Wei, and W. Li. Learning to rank in generative retrieval. In *The 38th Annual AAAI Conference on Artificial Intelligence*, 2023c.
- S. V. Mehta, J. Gupta, Y. Tay, M. Dehghani, V. Q. Tran, J. Rao, M. Najork, E. Strubell, and D. Metzler. DSI++: Updating transformer memory with new documents. *arXiv preprint arXiv:2212.09744*, 2022.
- R. Pradeep, K. Hui, J. Gupta, A. D. Lelkes, H. Zhuang, J. Lin, D. Metzler, and V. Q. Tran. How does generative retrieval scale to millions of passages? In *Gen-IR@SIGIR 2023: The First Workshop on Generative Information Retrieval*, 2023.
- E. Song, S. Kim, H. Lee, J. Kim, and J. Thorne. Re3val: Reinforced and reranked generative retrieval. In *Findings of the Association for Computational Linguistics*, 2024.
- Y. Tang, R. Zhang, J. Guo, J. Chen, Z. Zhu, S. Wang, D. Yin, and X. Cheng. Semantic-enhanced differentiable search index inspired by learning strategies. In *29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023a.



## References iii

- Y. Tang, R. Zhang, J. Guo, M. de Rijke, W. Chen, and X. Cheng. Listwise generative retrieval models via a sequential learning process. *ACM Transactions on Information Systems*, 2023b.
- Y. Tay, V. Q. Tran, M. Dehghani, J. Ni, D. Bahri, H. Mehta, Z. Qin, K. Hui, Z. Zhao, J. Gupta, T. Schuster, W. W. Cohen, and D. Metzler. Transformer memory as a differentiable search index. In *Advances in Neural Information Processing Systems*, volume 35, pages 21831–21843, 2022.
- H. Zeng, C. Luo, B. Jin, S. M. Sarwar, T. Wei, and H. Zamani. Scalable and effective generative information retrieval. In *The 2024 ACM Web Conference*, 2024a.
- H. Zeng, C. Luo, and H. Zamani. Planning ahead in generative retrieval: Guiding autoregressive generation through simultaneous decoding. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*, 2024b.
- Y. Zhou, Z. Dou, and J.-R. Wen. Enhancing generative retrieval with reinforcement learning from relevance feedback. In *EMNLP 2023: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.



## References iv

S. Zhuang, H. Ren, L. Shou, J. Pei, M. Gong, G. Zuccon, and D. Jiang. Bridging the gap between indexing and retrieval for differentiable search index with query generation. In *Gen-IR@SIGIR 2023: The First Workshop on Generative Information Retrieval*, 2023.

