Recent Advances in Generative Information Retrieval

Yubao Tang
Postdoc at IRLab, University of Amsterdam
y.tang3@uva.nl

June 9, 2025

Slides

Acknowledgements

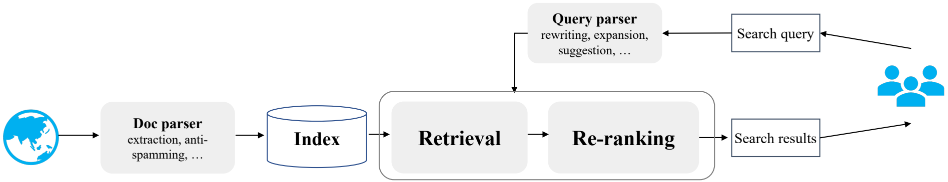
This slide deck is partially based on the GenIR tutorial, presented at SIGIR 2024 [Tang et al., 2024].

Information retrieval

Information retrieval (IR) is the activity of obtaining information resources that are relevant to an information need from a collection of those resources.

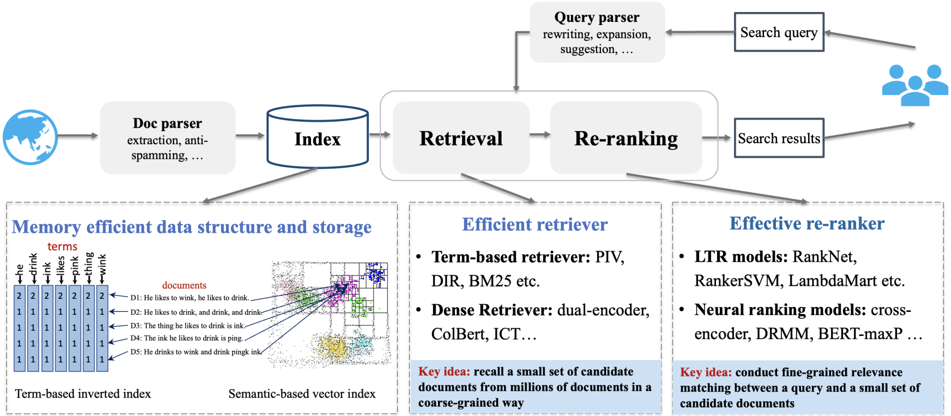
Given: User query (keywords, question, image, . . .)
Rank: Information objects (passages, documents, images, products, . . .)
Ordered by: Relevance scores

Core pipelined paradigm: Index-Retrieval-Ranking



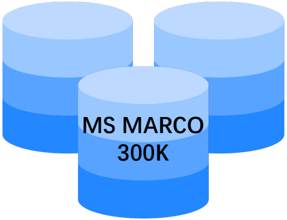
- Index: Build an index for each document in the entire corpus
- Retriever: Find an initial set of candidate documents for a query
- Re-ranker: Determine the relevance degree of each candidate

Index-Retrieval-Ranking: Disadvantages



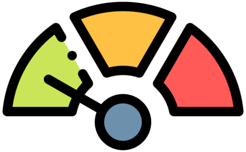
- **Effectiveness**: Heterogeneous ranking components are usually difficult to be optimized in an end-to-end way towards the global objective

Index-Retrieval-Ranking: Disadvantages



Big storage

GTR (Dense retrieval)
Memory size 1430MB



Slow inference speed

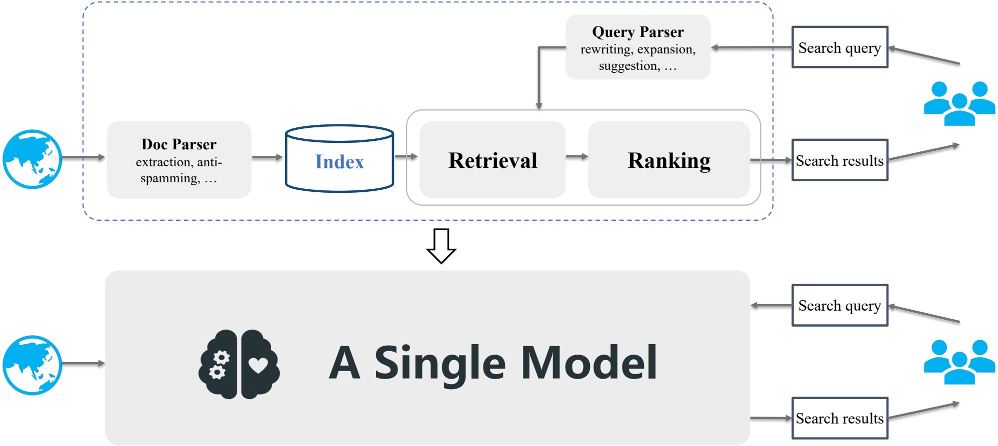
GTR (Dense retrieval)
Online latency 1.97s

- **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]

What if we replaced the pipelined architecture with a single consolidated model that efficiently and effectively encodes all of the information contained in the corpus?

Opinion paper: A single model for IR



Source: [Metzler et al., 2021]

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

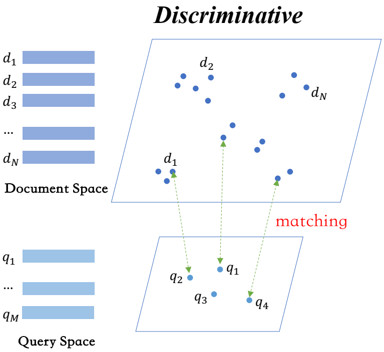
Source: [Najork, 2023]

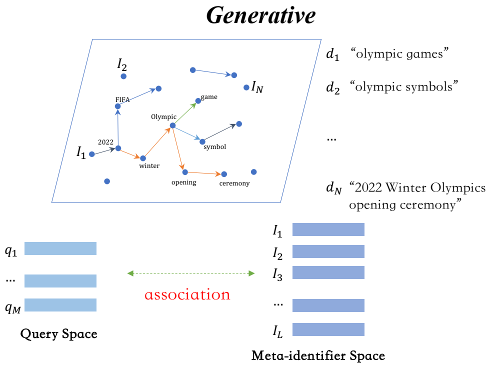
Closed-book generative retrieval

The IR task can be formulated as a sequence-to-sequence (Seq2Seq) generation problem

- **Input**: A sequence of query words
- **Output**: A sequence of document identifiers

Neural IR models: Discriminative vs. Generative





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

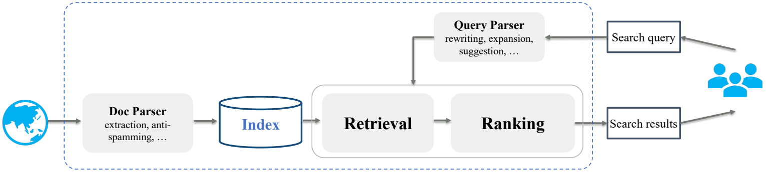
(probabilistic ranking principle)

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

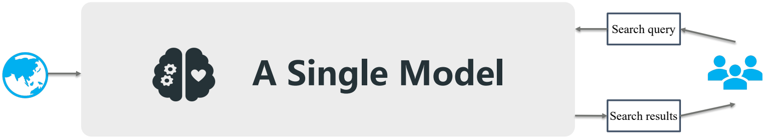
(query likelihood)

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?

	Dense retrieval	Generative retrieval
Memory size (MS MARCO 300K)	**GTR** **1430MB**	**GenRet** **860MB**
Online latency	**GTR** **1.97s**	**GenRet** **0.16s**

- **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]

Contents

- Definitions & preliminaries
- Docid design
- Inference strategies
- Training approaches
- Applications
- Challenges & opportunities

Definitions & preliminaries

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 (e.g., transformer-based encoder-decoder architecture)

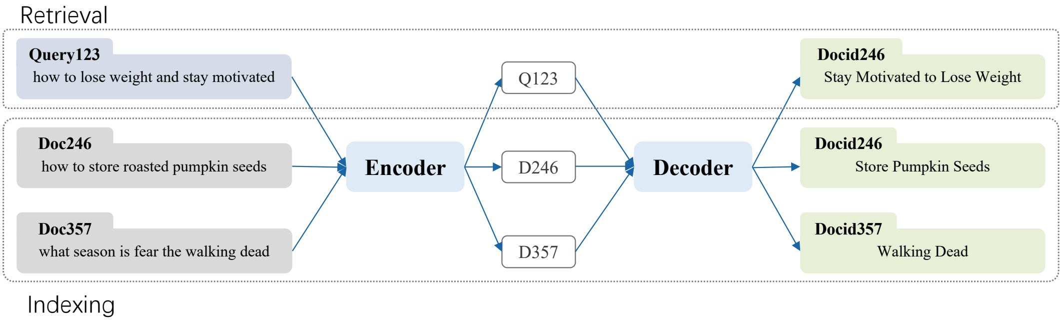
"Transformer Memory as a Differentiable Search Index". Tay et al. [2022]; "Autoregressive Entity Retrieval". De Cao et al. [2021]

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]

Training



Joint learning the indexing and retrieval tasks

Inference

- Once such a GR model is learned, it can be used to generate candidate docids for a test query q_t, all within a single, unified model,

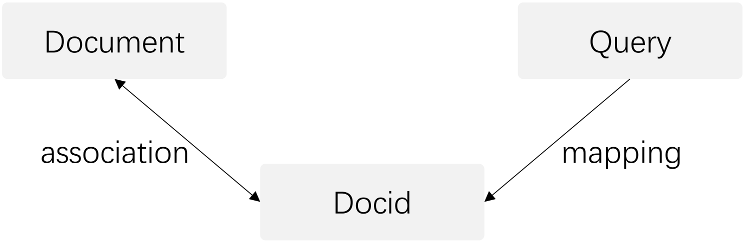
$$w_t = GR_\theta(q_t, w_0, w_1, \ldots, w_{t-1}),$$

where w_t is the t-th token in the docid string and the generation stops when decoding a special EOS token

- The docids generated with the top-K highest likelihood (joint probability of generated tokens within a docid) form a ranking list in descending order

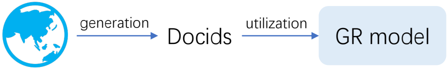
Docid design

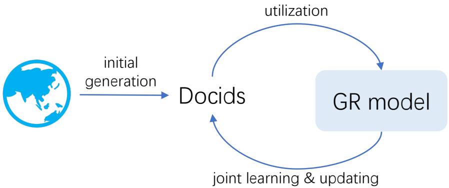
Research questions (1): Docid design



How to design docids for documents?

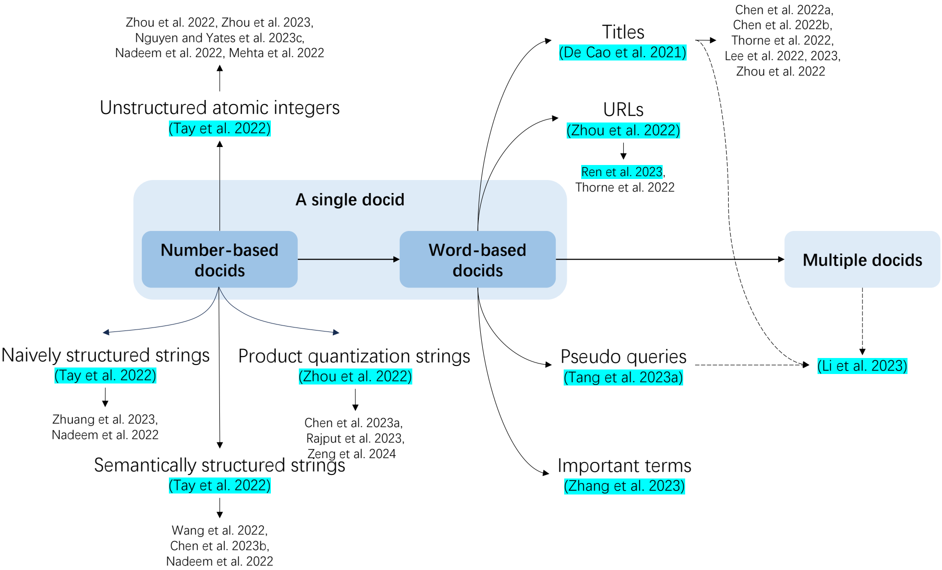
Categorization of docids





- Pre-defined static docids
- Learnable docids

Pre-defined static docids



Pre-defined static docids: Summary

Docid type		Construction	Uniqueness	The degree of semantic connection to the document	Relying on labeled data	Relying on metadata
A single docid: Number-based	Unstructured atomic integers (Tay et al. 2022)	Easy	Yes	None	No	No
	Naively structured strings (Tay et al. 2022)	Easy	Yes	None	No	No
	Semantically structured strings (Tay et al. 2022)	Moderate	Yes	Weak	No	No
	Product quantization strings (Zhou et al. 2022)	Moderate	No	Moderate	No	No
A single docid: Word-based	Titles (De Cao et al. 2021)	Easy	No	Strong	No	Yes
	URLs (Zhou et al. 2022, Ren et al. 2023)	Easy	Yes	Strong	No	Yes
	Pseudo queries (Tang et al. 2023a)	Moderate	No	Strong	Yes	No
	Important terms (Zhang et al. 2023)	Hard	Yes	Strong	Yes	No
Multiple docids	Single type: N-grams (Bevilacqua et al. 2022)	Easy	No	Moderate	No	No
	Diverse types (Li et al. 2023)	Moderate	No	Strong	Yes	Yes

How to design learnable docids tailored for retrieval tasks?

Learnable docids

- **Repeatable docids**:
  - GenRet [Sun et al., 2023] learns to tokenize documents into short discrete representations via a discrete auto-encoding, jointly training with the retrieval task
  - ASI [Yang et al., 2023] combines both the end-to-end learning of docids for existing and new documents and the end-to-end document retrieval based joint optimization
- **Unique docids**:
  - NOVO [Wang et al., 2023] uses unique n-gram sets identifying each document and can be generated in any order and can be optimized through retrieval tasks

Docid design: Summary

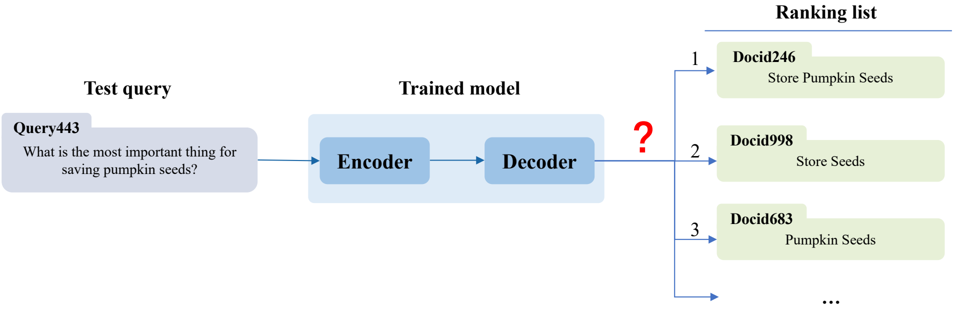
Docid type				
Pre-defined	Single	Number-based	- Simplified construction	- Low interpretability - Moderate performance
		Word-based	- High interpretability - Good performance	- Single-perspective representation of documents
	Multiple		- Comprehensive document representations - Better performance	- Slightly more complex construction
Learnable			- Adapting to GR objectives - Best performance	- Complex learning process

Docid design: Summary

Docid type				
Pre-defined	Single	Number-based	- Simplified construction	- Low interpretability - Moderate performance
		Word-based	- High interpretability - Good performance	- Single-perspective representation of documents
	Multiple		- Comprehensive document representations - Better performance	- Slightly more complex construction
Learnable			- Adapting to GR objectives - Best performance	- Complex learning process

Inference strategies

Research questions (2): Inference strategies



The generation process is different from general language generation

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

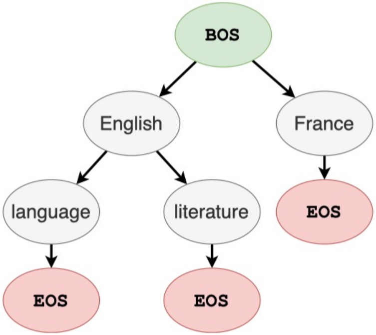


Image source: De Cao et al. [2021]

- For docids considering order of tokens
- **Applicable docids**: Naively structured strings, semantically structured strings, product quantization strings, titles, n-grams, URLs and pseudo queries

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

Single identifier: Constrained beam search with a prefix tree

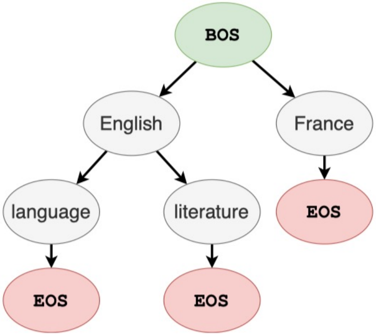
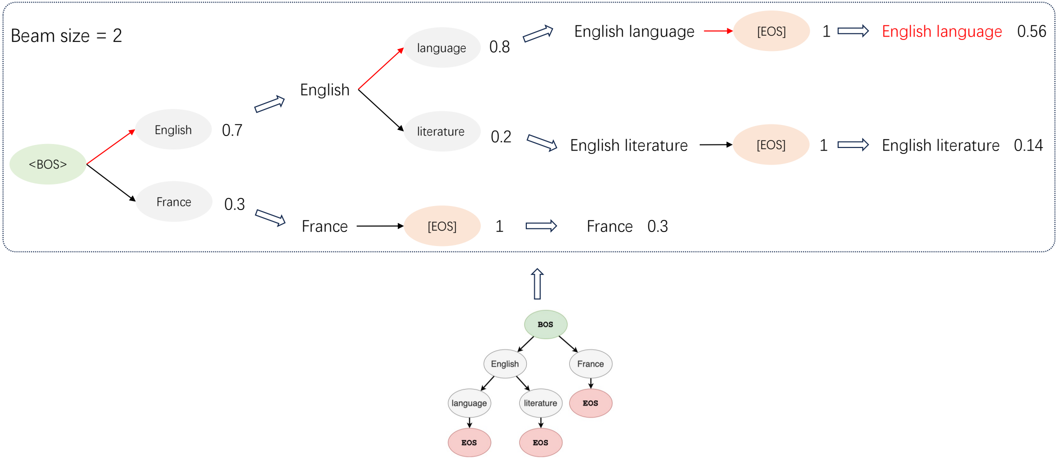


Image source: De Cao et al. [2021]

"Autoregressive Entity Retrieval". 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

Example

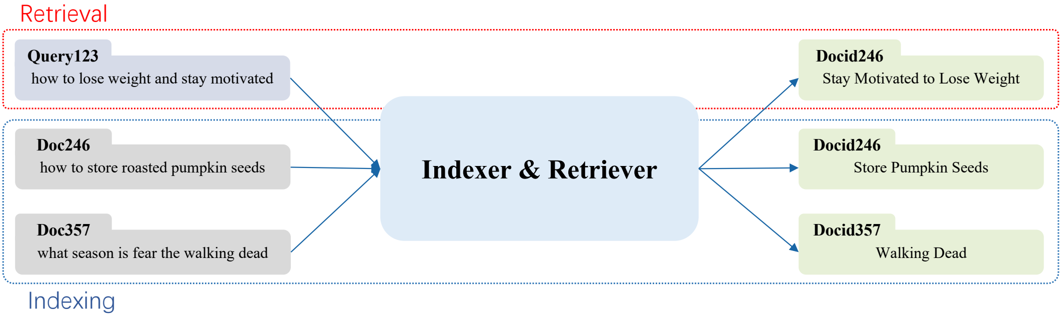

Beam size = 2
<BOS>
English
0.7
France
0.3
English
language
0.8
literature
0.2
English language
[EOS]
1
English language 0.56
English literature
[EOS]
1
English literature 0.14
France
[EOS]
1
France 0.3
BOS
English
France
language
literature
EOS
EOS
EOS

Inference: Summary

Inference strategies			
A single docid	Constrained beam search with prefix tree (De Cao et al. 2021)	- Simple	- It cannot generate in an unordered manner
	Constrained greedy search with inverted index (Zhang et al. 2023)	- It can generate in any permutations of docids	- It may require handling a significant amount of duplicate terms
Multiple docids	Constrained beam search with FM-index (Bevilacqua et al. 2022)	- It can store all the information of documents - The contributions of multiple docids comprehensively are considered	- It cannot generate in an unordered manner - Complex construction - Complex aggregation functions
	Scoring functions (Li et al. 2023)	- The contributions of multiple docids comprehensively are considered - Simple aggregation functions	- Depending on design

Training approaches

Research questions (3): Training approaches



Joint learning process of the indexing and retrieval tasks

Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Rich information in documents
  - Limited labeled data

Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Rich information in documents
  - Limited labeled data
- **How to learn heterogeneous tasks well within a single model?**
  - Different data distributions
  - Different optimization objectives

Challenges of training approaches

- **How to memorize the whole corpus effectively and efficiently?**
  - Rich information in documents
  - Limited labeled data
- **How to learn heterogeneous tasks well within a single model?**
  - Different data distributions
  - Different optimization objectives
- **How to handle a dynamically evolving document collection?**
  - Internal index: model parameters
  - High computational costs: re-training from scratch every time the underlying corpus is updated

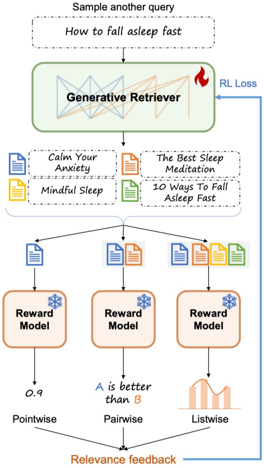
Multiple optimization: GenRRL [Zhou et al., 2023]

Based on reinforce learning framework

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

"Enhancing Generative Retrieval with Reinforcement Learning from Relevance Feedback". Zhou et al. [2023]

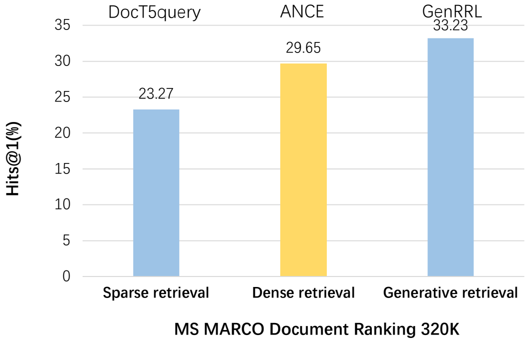
Multiple optimization: GenRRL [Zhou et al., 2023]

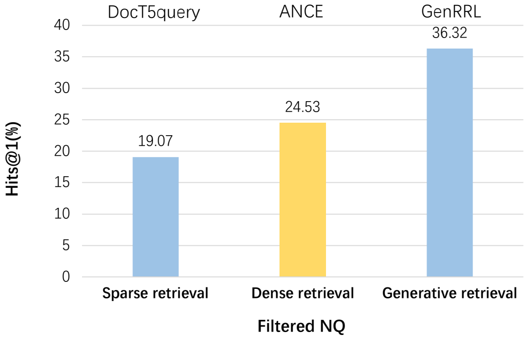


- Pointwise optimization:
  $-\sum_i (R(q, id_i) - b) \sum_t \log P(w_t^i \mid w_{<t}, q)$,
  where R is a reward model, and b is a baseline
- Pairwise optimization:
  $-\sum_{(id_i, id_j)} (R(q, id_i) \log p_{ij} + R(q, id_j) \log p_{ji}$,
  where $p_{ij} = |P(w_t^i \mid q) - P(w_t^j \mid q)|$
- Listwise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

"Enhancing Generative Retrieval with Reinforcement Learning from Relevance Feedback". Zhou et al. [2023]

GenRRL [Zhou et al., 2023]: Performance

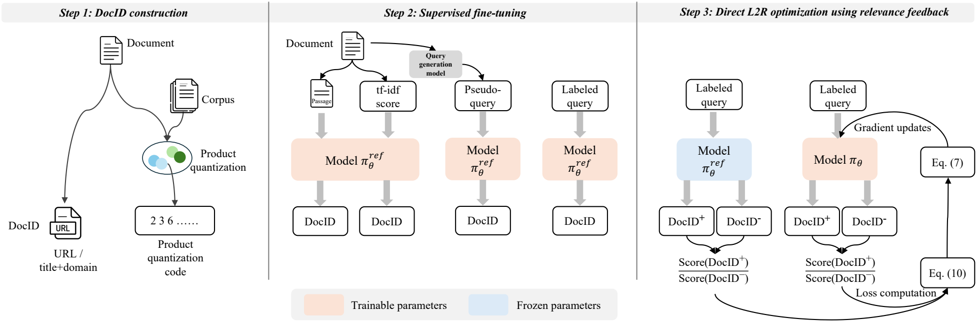




GenRRL introduce significant complexity, requiring the optimization of an auxiliary reward function followed by reinforcement fine-tuning, which is computationally expensive and often unstable

"Enhancing Generative Retrieval with Reinforcement Learning from Relevance Feedback". Zhou et al. [2023]

DDRO [Mekonnen et al., 2025]



- DDRO: aligns token-level docid generation with direct document-level relevance optimization via pairwise ranking, eliminating the need for explicit reward modeling and reinforcement learning

"Lightweight and Direct Document Relevance Optimization for Generative Information Retrieval". Mekonnen et al. [2025]

DDRO [Mekonnen et al., 2025]

Model	R@1	R@5	R@10	MRR@10
GenRRL (TU) [76]	33.01	63.62	74.91	45.93
GenRRL (Sum) [76]	33.23	64.48	**75.80**	46.62
DDRO (PQ)	32.92	64.36	73.02	45.76
DDRO (TU)	**38.24**	**66.46**	74.01	**50.07**

- Experimental results on MS 300K show that DDRO's potential to enhance retrieval effectiveness with a simplified optimization approach

"Lightweight and Direct Document Relevance Optimization for Generative Information Retrieval". Mekonnen et al. [2025]

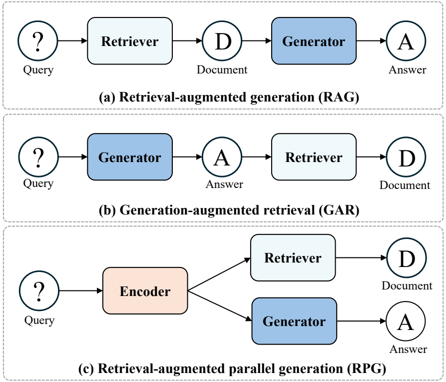
Combination of GR and retrieval-agumented generation (RAG)

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 ψ is the model parameters

MINT [Tang et al., 2025a]

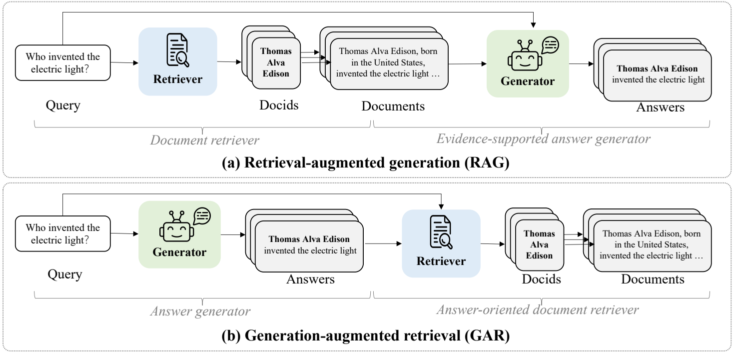


RAG faces challenges:

- the structural gap between traditional dense retrievers and autoregressive generators
- limited generation performance due to insufficient contextual guidance returned by the retriever

"Boosting Retrieval-Augmented Generation with Generation-Augmented Retrieval: A Co-Training Approach". Tang et al. [2025a]

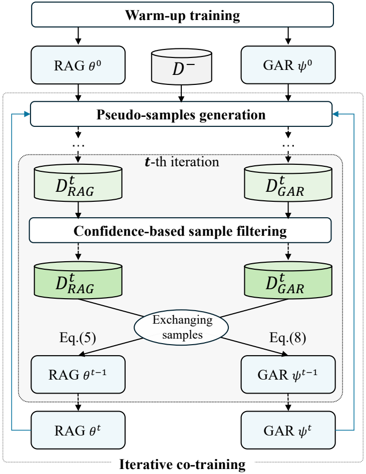
MINT [Tang et al., 2025a]



- MINT: a framework that enhances RAG by co-training Retrieval-augMented generatIon and geNeration-augmented reTrieval

"Boosting Retrieval-Augmented Generation with Generation-Augmented Retrieval: A Co-Training Approach". Tang et al. [2025a]

MINT [Tang et al., 2025a]



- Bridge the gap between the retriever and generator using a unified encoder-decoder structure
- Incorporate an iterative co-training strategy between RAG and GAR, enabling mutual enhancement through pseudo-samples generation

"Boosting Retrieval-Augmented Generation with Generation-Augmented Retrieval: A Co-Training Approach". Tang et al. [2025a]

MINT [Tang et al., 2025a]

Methods	FC	Entity linking		Slot filling		Open domain QA				Dial
	FEV	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
Sparse & dense retrieval											
BM25†	30.29	2.82	1.38	3.84	32.04	43.37	12.34	31.31	14.40	1.20	17.20
DPR†	59.10	79.51	-	-	60.61	70.91	31.13	39.47	35.48	-	37.66
MT-DPR†	64.05	81.69	49.20	46.95	57.64	73.81	32.80	38.42	36.29	10.86	38.00
RAG†	66.04	76.40	48.28	46.01	53.57	67.97	38.25	34.61	41.38	10.70	38.04
E5†	68.52	79.72	50.47	48.10	54.48	70.01	39.40	37.35	42.62	11.02	39.16
SimLM†	68.06	80.11	51.98	49.54	55.42	72.11	38.58	36.11	41.80	10.36	38.31
Generative retrieval											
T5†	71.63	86.71	67.34	62.20	64.87	78.51	38.69	38.09	45.73	10.35	42.51
BART†	69.90	87.43	67.22	60.71	61.57	76.13	39.84	38.44	47.26	10.09	40.19
SEAL†	70.55	82.05	57.09	58.70	55.91	74.89	39.67	40.54	44.16	9.32	41.59
CorpusBrain†	72.23	88.79	69.40	63.23	63.42	79.05	40.09	39.45	47.97	10.68	42.19
GenRet	72.45	88.92	69.57	63.56	63.77	79.52	40.25	39.78	48.21	10.67	42.26
Llama2†	74.39	85.53	66.55	61.45	66.12	77.90	40.59	40.37	48.43	10.66	42.69
UniGen	72.57	89.12	69.77	63.74	63.86	79.74	40.43	39.84	48.41	10.73	42.53
GCoQA	70.83	87.43	67.12	61.03	61.35	77.23	38.77	37.25	46.93	10.20	40.78
CorpusLM (T5)†	75.64	90.96	70.35	65.43	68.89	81.08	41.46	39.31	48.80	10.90	44.96
CorpusLM (Llama2)†	76.21	88.59	69.39	64.18	69.17	80.79	44.10	42.06	50.62	10.88	43.92
Co-training RAG and GAR (Ours)											
MINT_R (T5)	75.69	90.99	70.42	65.52	68.91	81.00	41.52	39.37	48.84	10.91	44.97
MINT_G (T5)	78.84	92.24	73.47	67.83	70.03	83.24	43.89	42.01	50.13	10.93	45.73
MINT_{RG} (T5)	78.87	**92.26***	**73.51***	**67.85***	70.08	**83.27***	43.95	42.16	50.22	**10.95**	**45.81***
MINT_R (Llama2)	76.51	88.84	69.85	64.82	69.25	81.32	44.78	42.76	51.21	10.90	43.98
MINT_G (Llama2)	77.56	89.38	70.13	65.47	70.56	81.84	45.83	43.81	51.78	10.91	44.24
MINT_{RG} (Llama2)	**79.13***	90.67	71.46	66.32	**71.48***	82.47	**46.52***	**44.15***	**52.81***	10.92	44.87

Retrieval performance:

- Compared to most GR baselines, MINT_{RG} consistently achieves superior performance. For example, on the HoPo dataset, MINT_{RG} (Llama2) performs better than the SOTA CorpusLM (Llama2) by 5% in terms of R-precision

"Boosting Retrieval-Augmented Generation with Generation-Augmented Retrieval: A Co-Training Approach". Tang et al. [2025a]

Training: Summary

Training approaches				
Standard approach (Tay et al. 2022)			- Simple	- Moderate performance
Stationary	Multi-granularity enhanced (Tang et al. 2023a)		- Enhancing the memorization ability	- Requiring extra tools for selecting important paragraphs or sentences
	Pseudo query enhanced (Zhuang et al. 2023)		- Reducing the gap between training and inference	- Depending on labeled data
	Pre-training based (Chen et al. 2022b)		- Addressing the issue of no or limited labeled data	- Depending on the quality of pre-training corpora and task design
	Pairwise optimization (Li et al., 2023c)		- Enhancing the relevance signals	- Multi-stage training
	Multiple optimization (Zhou et al., 2023)		- Fully utilize relevance signals	- Rely on various external algorithms
Dynamic	IncDSI (Kishore et al.2023)	Unstructured atomic integers & constrained optimization	- Simple design & high efficiency	- Moderate performance - Ignore catastrophic forgetting
	DSI++ (Mehta et al. 2022)	Unstructured atomic integers & experience replay	- Simple design - Good performance	- High computational cost - Difficult to capture the relationship between old and new corpora
	CLEVER (Chen et al. 2023a)	Incremental product quantization & memory-augmented learning	- High efficiency - Better at capturing the relationship between old and new corpora - Better performance	- More complicated docid implementation - Extra memory bank
Apply GR in QA	GCoQA (Li et al. 2023a)		- Large model knowledge is introduced	- Data preprocessing incurs high costs - Separate retrieval and reader module
	Re3val (Song et al., 2024)		- A cross-encoder is introduced	As above
	UniGen (Li et al., 2023a)		- Joint optimization for GR and QA	- The generation process of the retriever and reader is not explicitly connected
Large-scale corpora	PIPOR & PAG (Zeng et al. 2024a & Zeng et al. 2024b)		- The optimization considers the docid structure	- The optimization process is complex

Applications

Research questions (4): Applications

How to employ generative retrieval models in different downstream tasks?

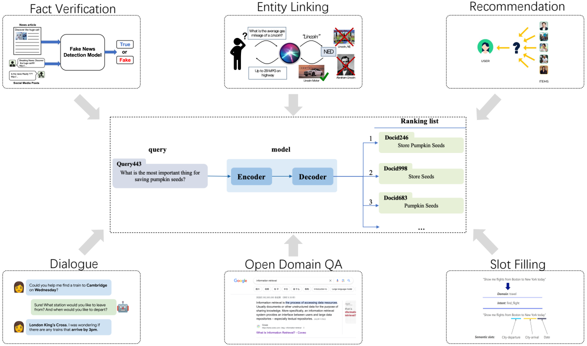


Image source: Murayama [2021], Stanford blog, Froomle AI, Google, Zero-shot blog, and Heck and Heck [2020]

A range of target tasks

Fact Verification
De Cao et al. 2021, Chen et al. 2022b, Chen et al. 2022a, Thorne et al. 2022, Lee et al. 2023

Open Domain QA
De Cao et al. 2021, Chen et al. 2022b, Zhou et al. 2022, Lee et al. 2023

Entity Linking
De Cao et al. 2021, Chen et al. 2022b, Lee et al. 2023

Knowledge-intensive language tasks

A range of target tasks

Fact Verification
De Cao et al. 2021, Chen et al. 2022b, Chen et al. 2022a, Thorne et al. 2022, Lee et al. 2023

Open Domain QA
De Cao et al. 2021, Chen et al. 2022b, Zhou et al. 2022, Lee et al. 2023

Entity Linking
De Cao et al. 2021, Chen et al. 2022b, Lee et al. 2023

Multi-hop retrieval
Lee et al. 2022

Recommendation
Si et al. 2023, Rajput et al. 2023

Code retrieval
Naddem et al. 2022

More retrieval tasks

A range of target tasks

Fact Verification
De Cao et al. 2021, Chen et al. 2022b, Chen et al. 2022a, Thorne et al. 2022, Lee et al. 2023

Open Domain QA
De Cao et al. 2021, Chen et al. 2022b, Zhou et al. 2022, Lee et al. 2023

Entity Linking
De Cao et al. 2021, Chen et al. 2022b, Lee et al. 2023

Multi-hop retrieval
Lee et al. 2022

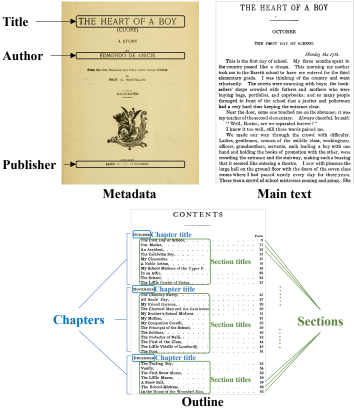
Recommendation
Si et al. 2023, Rajput et al. 2023

Code retrieval
Naddem et al. 2022

Official site retrieval
Tang et al. 2023a

Industry retrieval tasks

GBS [Tang et al., 2025b]

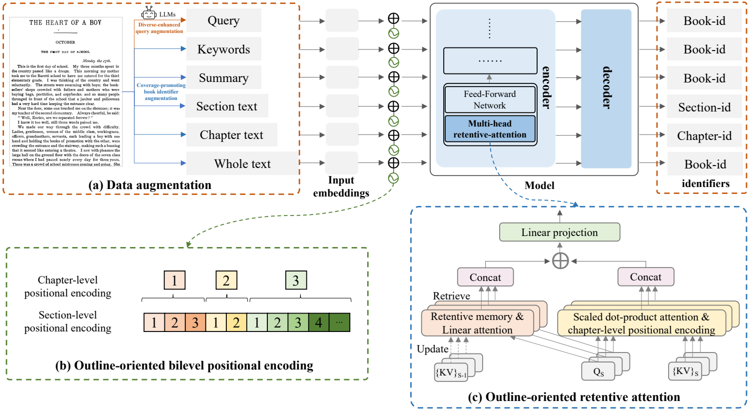


- Books contain complex, multi-faceted information, where the outline provides hierarchical information
- How can GR be applied to book search?

"Generative Retrieval for Book Search". Tang et al. [2025b]

GBS [Tang et al., 2025b]

- Generative retrieval framework for Book Search (GBS)



"Generative Retrieval for Book Search". Tang et al. [2025b]

GBS [Tang et al., 2025b]

	Method	BBS 10K		BBS 20K		BBS 40K		WhatsThatBook	
		Hits@10	MRR@20	Hits@10	MRR@20	Hits@10	MRR@20	Hits@10	MRR@20
Sparse	BM25	41.8	30.5	40.6	30.1	40.1	29.8	45.3	41.6
	DocT5query	46.6	39.3	42.5	35.4	37.5	30.7	48.2	42.8
Dense	RepBERT	53.1	46.4	48.3	41.9	45.6	37.3	56.8	48.1
	DPR	51.3	43.6	46.4	40.3	42.1	35.9	54.2	45.7
Generative	DSI	20.7	13.4	18.5	11.6	13.5	7.2	22.6	15.4
	GENRE	26.5	22.1	24.5	19.3	19.3	15.6	29.1	24.7
	SEAL	27.8	23.6	25.7	20.6	20.5	16.2	30.5	24.8
	DSI-QG	40.3	35.7	36.8	28.9	32.1	23.7	44.6	25.9
	NCI	42.8	36.8	37.2	29.4	32.8	24.2	45.2	39.4
	Corpusbrain	48.9	43.1	42.4	37.5	36.3	31.5	52.3	45.7
	Ultron	48.6	44.5	41.3	36.3	35.7	30.1	51.9	45.2
	GenRet	51.3	46.6	47.2	42.1	41.4	37.8	55.8	49.3
	NOVO	52.7	47.3	47.6	42.8	41.8	38.4	56.5	49.7
	ASI	58.4	53.6	53.5	44.6	46.6	40.1	56.2	49.5
	RIPOR	62.5	52.8	56.8	46.2	52.9	42.7	66.7	55.4
Ours	GBS^S	66.4	54.1	61.3	49.5	56.3	46.5	70.4	58.1
	GBS^P	**66.7**†	**54.4**†	**61.6**†	**49.8**†	**56.7**†	**46.9**†	**70.7**†	**58.6**†

"Generative Retrieval for Book Search". Tang et al. [2025b]

Overall performance

Tasks (Datasets)	GR method & DR baseline	Retrieval performance	Memory cost	Inference time
KILT (Wikipedia)	GENRE	83.6 RP ✓	2.1 GB ✓	
	DPR+BERT	72.9 RP	70.9GB	
Fact Verification-Document retrieval (FEVER)	GERE	84.3 P ✓	-	5.35ms ✓
	RAG	62.17 P	-	13.89ms
Multi-hop retrieval (EntailTree & HotpotQA)	GMR	52.5 F1 ✓	2.95 GB ✓	
	ST5	16.9 F1	15.81GB	
Sequential recommendation (Sports and Outdoors)	TIGER	1.81 nDCG@5 ✓	-	-
	S^3-Rec	1.61 nDCG@5	-	-
Code retrieval (CodeSearchNet)	CodeDSI	90.4 Acc ✓	-	-
	CodeBERT	89.8 Acc	-	-
Official site retrieval (Industry online data)	SE-DSI	+42.4 R@20 ✓	-31 times ✓	-2.5 times ✓
	DualEnc			

Data source: papers mentioned above

Overall performance

Tasks (Datasets)	GR method & DR baseline	Retrieval performance	Memory cost	Inference time
KILT (Wikipedia)	GENRE	83.6 RP ✓	2.1 GB ✓	
	DPR+BERT	72.9 RP	70.9GB	
Fact Verification-Document retrieval (FEVER)	GERE	84.3 P ✓	-	5.35ms ✓
	RAG	62.17 P	-	13.89ms
Multi-hop retrieval (EntailTree & HotpotQA)	GMR	52.5 F1 ✓	2.95 GB ✓	
	ST5	16.9 F1	15.81GB	
Sequential recommendation (Sports and Outdoors)	TIGER	1.81 nDCG@5 ✓	-	-
	S^3-Rec	1.61 nDCG@5	-	-
Code retrieval (CodeSearchNet)	CodeDSI	90.4 Acc ✓	-	-
	CodeBERT	89.8 Acc	-	-
Official site retrieval (Industry online data)	SE-DSI	+42.4 R@20 ✓	-31 times ✓	-2.5 times ✓
	DualEnc			

Data source: papers mentioned above

The performance of current GR methods can only compete with part of dense retrieval baselines, but still falls short compared to full-ranking methods

Challenges & opportunities

Pros of generative retrieval

IR 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 real 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 an 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

Q & A

Thank you for joining us today!

Homepage

References

References i

N. De Cao, G. Izacard, S. Riedel, and F. Petroni. Autoregressive entity retrieval. In *International Conference on Learning Representations*, 2021.

R. Deffayet, T. Thonet, D. Hwang, V. Lehoux, J.-M. Renders, and M. de Rijke. Sardine: A simulator for automated recommendation in dynamic and interactive environments. *ACM Transactions on Recommender Systems*, To appear.

L. Heck and S. Heck. Zero-shot visual slot filling as question answering. *arXiv preprint arXiv:2011.12340*, 2020.

K. A. Mekonnen, Y. Tang, and M. de Rijke. Lightweight and direct document relevance optimization for generative information retrieval. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025.

D. Metzler, Y. Tay, D. Bahri, and M. Najork. Rethinking search: Making domain experts out of dilettantes. *SIGIR Forum*, 55(1):1–27, 2021.

T. Murayama. Dataset of fake news detection and fact verification: a survey. *arXiv preprint arXiv:2111.03299*, 2021.

M. Najork. Generative information retrieval (slides), 2023. URL https://docs.google.com/presentation/d/191AeVzPkh20Ly855tKDkz1uv-1pHV_9GxfntiTJPUug/.

C. Shah and E. M. Bender. Situating search. In *Proceedings of the 2022 Conference on Human Information Interaction and Retrieval*, pages 221–232, 2022.

W. Sun, L. Yan, Z. Chen, S. Wang, H. Zhu, P. Ren, Z. Chen, D. Yin, M. de Rijke, and Z. Ren. Learning to tokenize for generative retrieval. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Y. Tang, R. Zhang, Z. Ren, J. Guo, and M. de Rijke. Recent advances in generative information retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 3005–3008, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704314. doi: 10.1145/3626772.3661379. URL https://doi.org/10.1145/3626772.3661379.

Y. Tang, R. Zhang, J. Guo, M. de Rijke, Y. Fan, and X. Cheng. Boosting retrieval-augmented generation with generation-augmented retrieval: A co-training approach. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025a.

Y. Tang, R. Zhang, J. Guo, M. de Rijke, S. Liu, S. Wang, D. Yin, and X. Cheng. Generative retrieval for book search. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '25, page 2606–2617, New York, NY, USA, 2025b. Association for Computing Machinery. ISBN 9798400712456. doi: 10.1145/3690624.3709435. URL https://doi.org/10.1145/3690624.3709435.

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.

Z. Wang, Y. Zhou, Y. Tu, and Z. Dou. Novo: Learnable and interpretable document identifiers for model-based ir. In *Proceedings of the 32nd ACM Conference on Information and Knowledge Management*, 2023.

T. Yang, M. Song, Z. Zhang, H. Huang, W. Deng, F. Sun, and Q. Zhang. Auto search indexer for end-to-end document retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

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.