Generative Information Retrieval

SIGIR 2024 tutorial – Section 2

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

Section 2:
Definitions & Preliminaries

SCAN ME

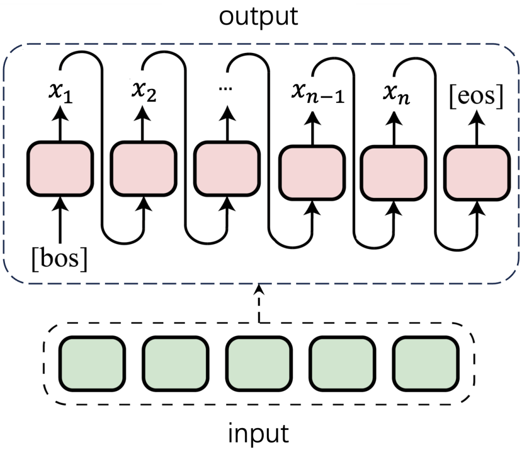
Generative retrieval: Definition

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

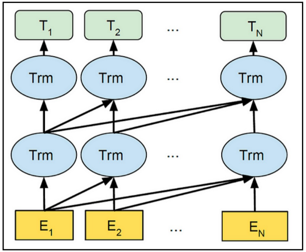
"Transformer Memory as a Differentiable Search Index". Tay et al. [2022]; "Autoregressive Entity Retrieval". De Cao et al. [2021]

Autoregressive formulation

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



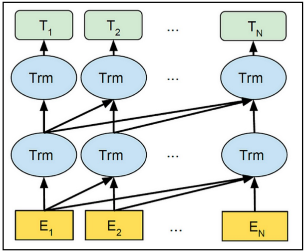
Autoregressive models



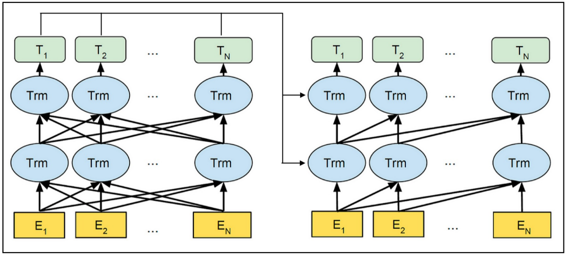
Decoder-only

Image source: Kenton and Toutanova [2019] and Retrieval-based Language Models and Applications

Autoregressive models



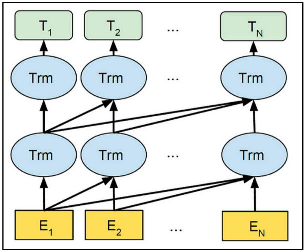
Decoder-only



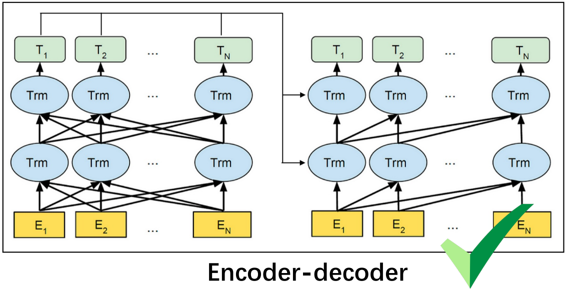
Encoder-decoder

Image source: Kenton and Toutanova [2019] and Retrieval-based Language Models and Applications

Autoregressive models



Decoder-only



Encoder-decoder

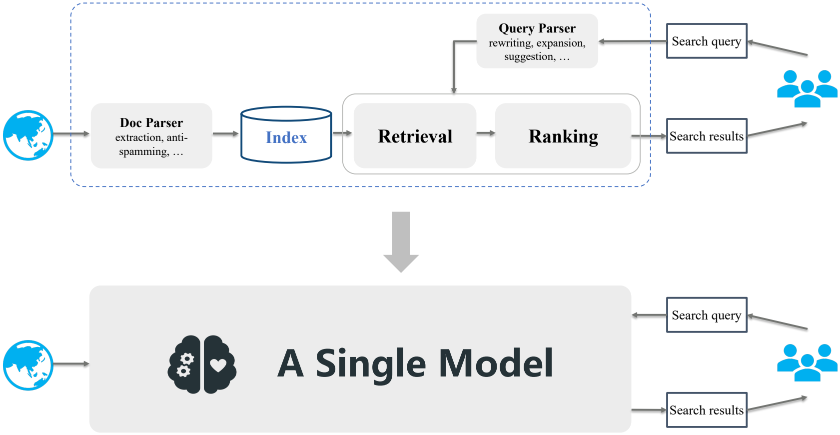
Image source: Kenton and Toutanova [2019] and Retrieval-based Language Models and Applications

Generative retrieval: Definition

GR usually exploits a Seq2Seq encoder-decoder architecture to generate a ranked list of docids for 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]

Revisit the key idea



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

Source: [Tay et al., 2022]

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]

Indexing: Formulation

Given:

- A corpus of documents D;
- A corresponding docid set I_D;

Indexing: Formulation

Given:

- A corpus of documents D;
- A corresponding docid set I_D;

The indexing task directly takes each original document $d \in D$ as input and generates its docid $id \in I_D$ as output in a straightforward Seq2Seq fashion, i.e.,

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

where θ denotes the model parameters, and $P(id \mid d; \theta)$ is the likelihood of each docid id given the document d

Retrieval: Formulation

Given:

- A query set Q;
- A set of relevant docids I_Q;

Retrieval: Formulation

Given:

- A query set Q;
- A set of relevant docids I_Q;

The retrieval task aims to generate a ranked list of relevant docids $id^q \in I_Q$ in response to a query $q \in Q$ with the indexed information, i.e.,

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

where $P(id^q \mid q; \theta)$ is the likelihood of each relevant docid id^q given the query q

Training

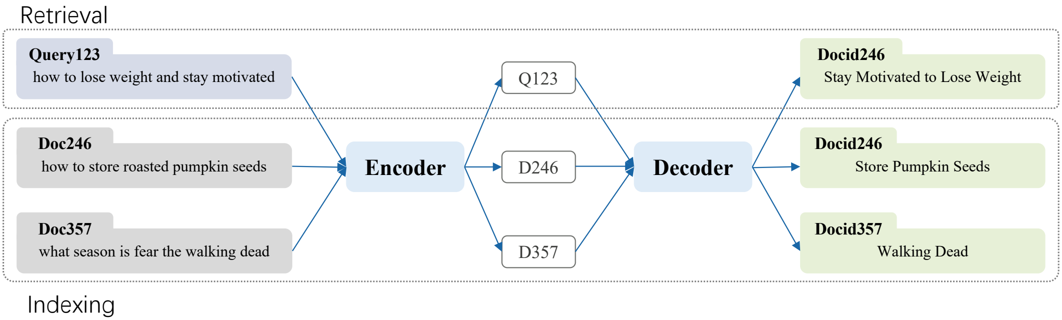
Following the above two basic operations, i.e., indexing and retrieval, a single model can be optimized directly in an end-to-end manner towards a global objective,

Training

Following the above two basic operations, i.e., indexing and retrieval, a single model can be optimized directly in an end-to-end manner towards a global objective,

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

Training: An example



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,

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

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

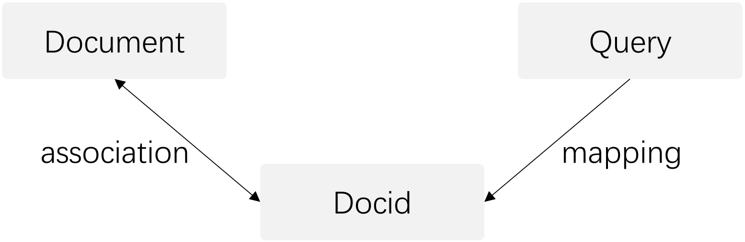
Inference

- 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

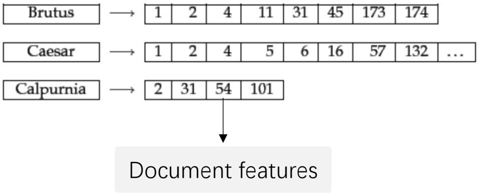
Research questions (1): Docid design



Unfortunately, there is no natural identifier for each document!

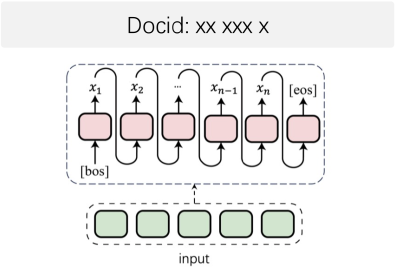
Research questions (1): Docid design

Traditional information retrieval



As an entry

Generative retrieval

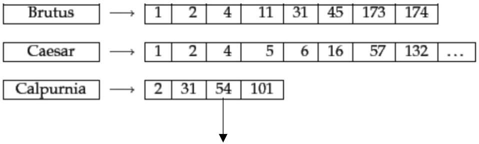


For generation

Image source: Information retrieval book

Research questions (1): Docid design

Traditional information retrieval

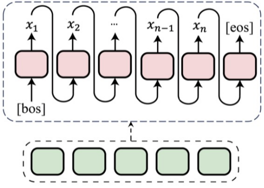


Document features

As an entry

Generative retrieval

Docid: xx xxx x



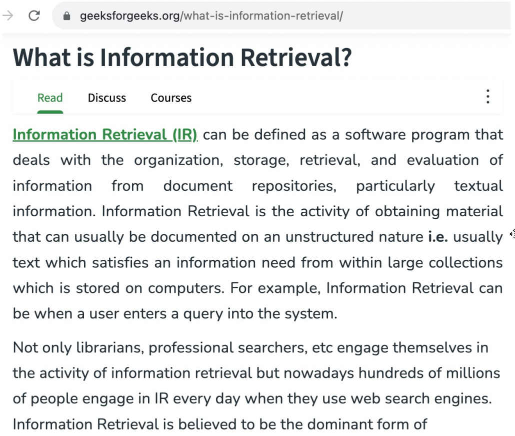
For generation

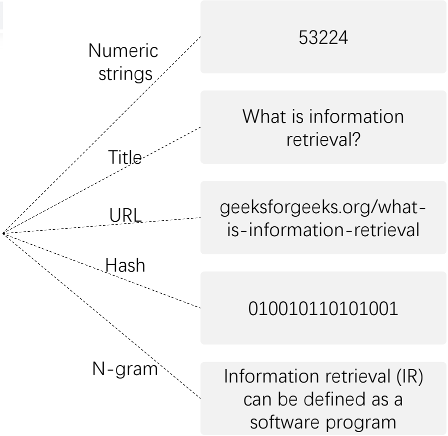
How to design docids for documents?

Image source: Information retrieval book

Research questions (1): Docid design

- Possible design choices





Document source: Geeksforgeeks

Challenges of docid design

- **Shall we use randomized numbers or codes as docids?**

Challenges of docid design

- **Shall we use randomized numbers or codes as docids?**
- **If not, how to obtain proper identifiers for documents?**
  - Titles, URLs or ?

Challenges of docid design

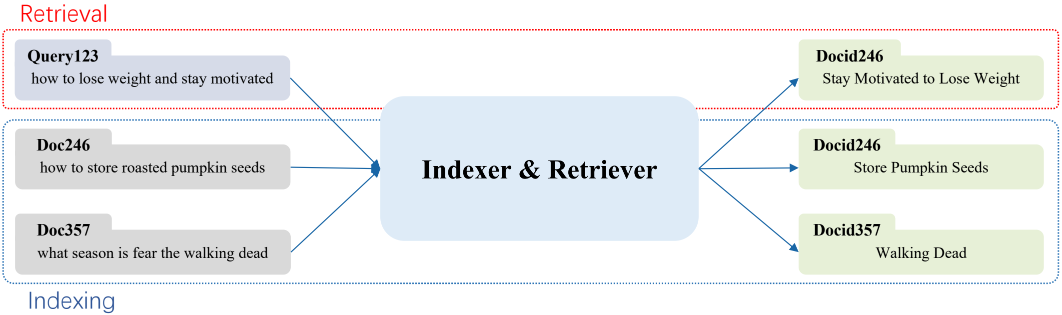
- **Shall we use randomized numbers or codes as docids?**
- **If not, how to obtain proper identifiers for documents?**
  - Titles, URLs or ?
- **Would the choices of different docids affect the model performance (e.g., effectiveness, capacity, etc.)?**
  - Long (e.g., 728 hash code) vs. Short docids (e.g., n-grams)
  - Single (e.g., title or URL) vs. Multiple docids (e.g., multiple keywords)

Challenges of docid design

- **Shall we use randomized numbers or codes as docids?**
- **If not, how to obtain proper identifiers for documents?**
  - Titles, URLs or ?
- **Would the choices of different docids affect the model performance (e.g., effectiveness, capacity, etc.)?**
  - Long (e.g., 728 hash code) vs. Short docids (e.g., n-grams)
  - Single (e.g., title or URL) vs. Multiple docids (e.g., multiple keywords)

We will tackle these questions in Section 3!

Research questions (2): 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

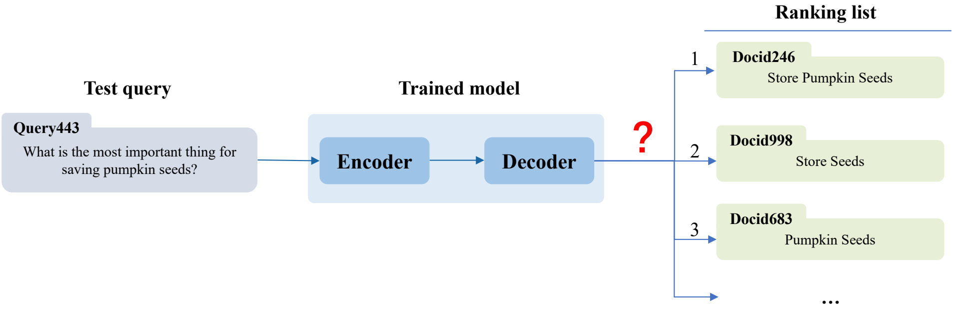
- **How to memorize the whole corpus effectively and efficiently?**
  - Rich information in documents
  - Limited labeled data
- **How to learn heterogeneous tasks well within a single model?**
  - Different data distributions
  - Different optimization objectives
- **How to handle a dynamically evolving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

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

We will tackle these questions in Section 4!

Research questions (3): Inference strategies



The generation process is different from general language generation

Challenges of model inference

- **How to generate valid docids?**
  - Limited docids vs. free generation

Challenges of model inference

- **How to generate valid docids?**
  - Limited docids vs. free generation
- **How to organize the docids for large scale corpus?**
  - Data structure for docids over millions of documents

Challenges of model inference

- **How to generate valid docids?**
  - Limited docids vs. free generation
- **How to organize the docids for large scale corpus?**
  - Data structure for docids over millions of documents
- **How to generate a ranked list of docids for a query?**
  - One-by-one generation: likelihood probabilities
  - One-time generation: directly decoding a sequence of docids

Challenges of model inference

- **How to generate valid docids?**
  - Limited docids vs. free generation
- **How to organize the docids for large scale corpus?**
  - Data structure for docids over millions of documents
- **How to generate a ranked list of docids for a query?**
  - One-by-one generation: likelihood probabilities
  - One-time generation: directly decoding a sequence of docids

We will tackle these questions in Section 5!

Research questions (4): Applications

How to employ generative retrieval models in different downstream tasks?

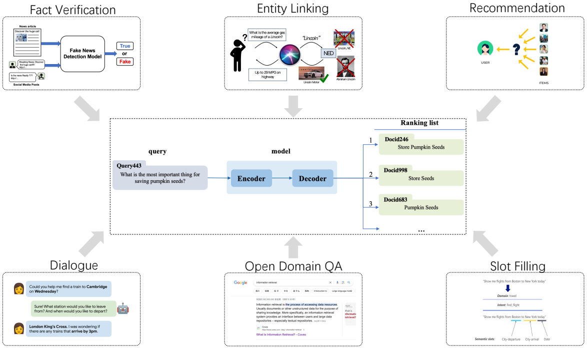
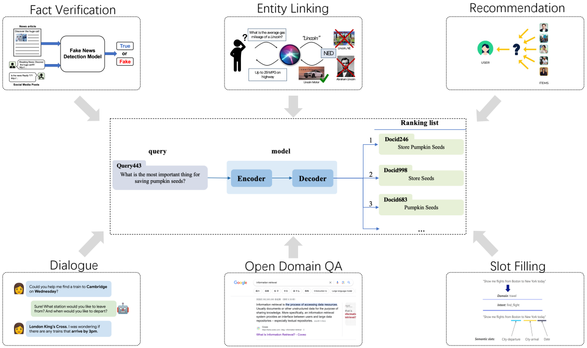


Image source: Murayama [2021], Stanford blog, Froomle AI, Google, Zero-shot blog, and Heck and Heck [2020]

Research questions (4): Applications

How to employ generative retrieval models in different downstream tasks?



We will tackle this question in Section 6!

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

References

References i

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

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

J. D. M.-W. C. Kenton and L. K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

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

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.