Generative Information Retrieval

SIGIR 2024 tutorial – Section 5

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 5:
Inference strategies

SCAN
ME

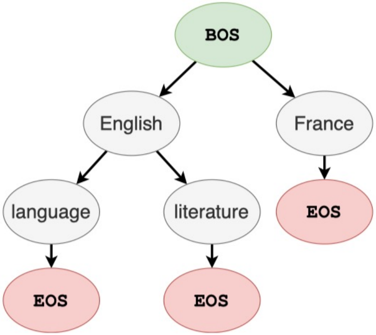
Roadmap of inference strategies

- A **single identifier** to represent a document:
  - Constrained beam search with a prefix tree
  - Constrained greedy search with the inverted index

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



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

Image source: De Cao et al. [2021]

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

Single identifier: Constrained beam search with a prefix tree

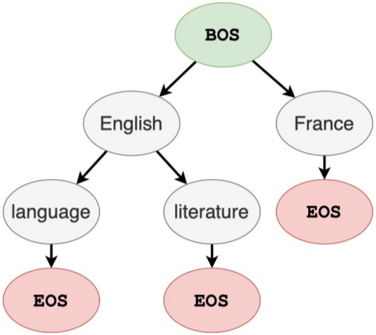
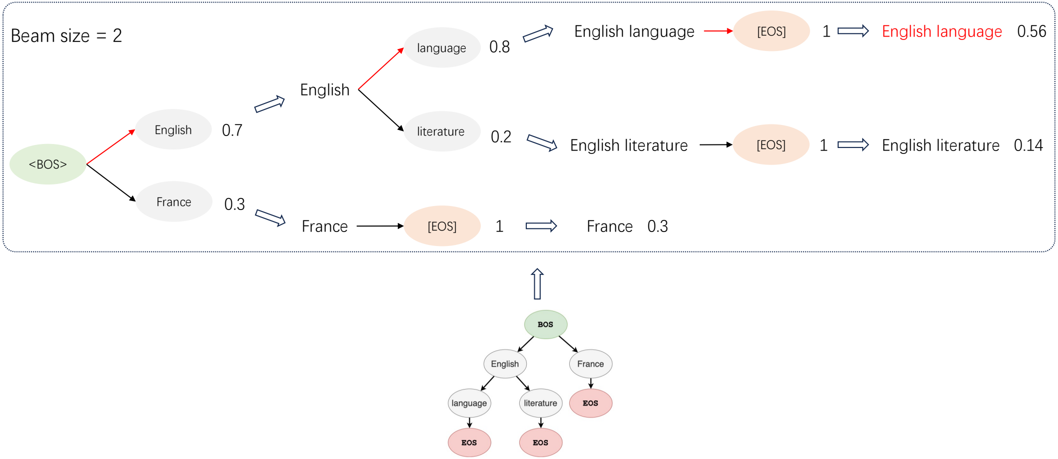


Image source: De Cao et al. [2021]

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

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

Example


Beam size = 2
<BOS>
English
0.7
France
0.3
English
language
0.8
literature
0.2
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

Single identifier: Constrained greedy search with the inverted index

- Applicable docids: Important terms

"Term-Sets Can Be Strong Document Identifiers For Auto-Regressive Search Engines". Zhang et al. [2023]

Single identifier: Constrained greedy search with the inverted index

- Applicable docids: Important terms
- Inverted index table: Enable the generation in any permutations (unordered docids) are constructed

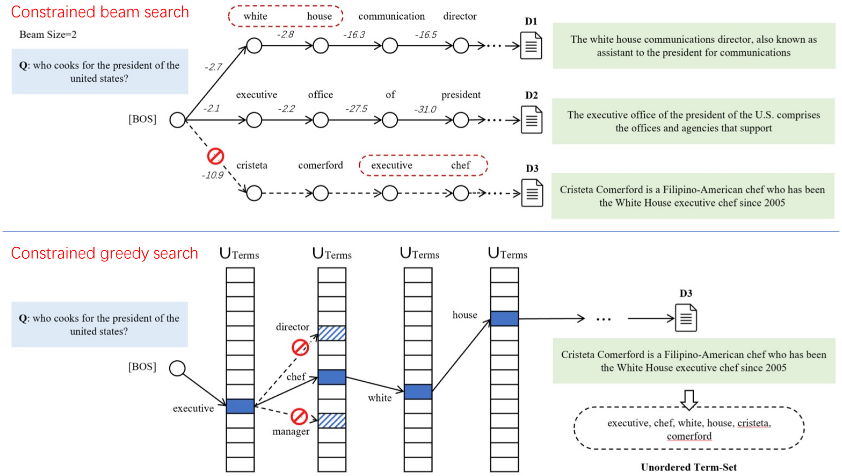
"Term-Sets Can Be Strong Document Identifiers For Auto-Regressive Search Engines". Zhang et al. [2023]

Single identifier: Constrained greedy search with the inverted index

- Applicable docids: Important terms
- Inverted index table: Enable the generation in any permutations (unordered docids) are constructed
- Generation process: The model is expected to produce docids of the highest generation likelihood. At each step of generation, the terms from the inverted index table which give rise to the top-K generation likelihood are greedily selected

"Term-Sets Can Be Strong Document Identifiers For Auto-Regressive Search Engines". Zhang et al. [2023]

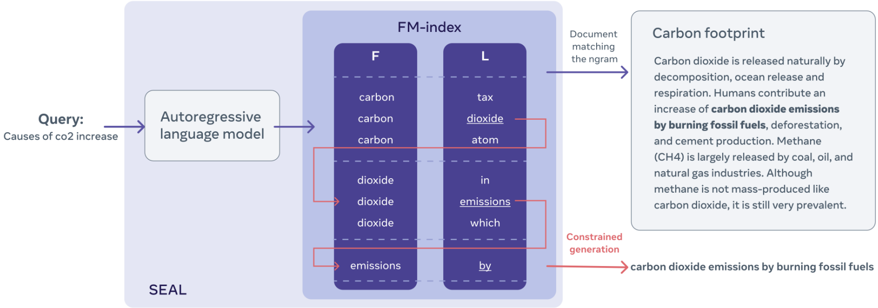
Constrained beam search vs. Constrained greedy search



"Term-Sets Can Be Strong Document Identifiers For Auto-Regressive Search Engines". Zhang et al. [2023]

Image source: Zhang et al. [2023]

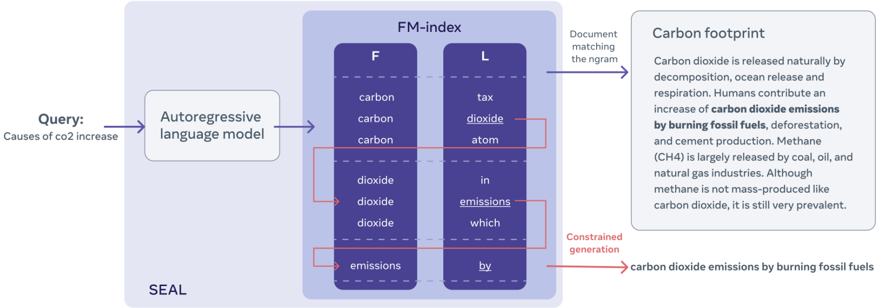
Multiple identifiers: Constrained beam search with the FM-index



- Applicable docids: N-grams based docids

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

Multiple identifiers: Constrained beam search with the FM-index



- Applicable docids: N-grams based docids
- FM-index: An index combining the Burrows-Wheeler Transform (BWT) with a few small auxiliary data structures

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

FM-index: N-gram level scores

Given an input query q, we obtain the weight of each predicted n-gram n:

$$score(n, q) = \max\left(0, \log \frac{P(n|q)(1 - P(n))}{P(n)(1 - P(n|q))}\right),$$

where $P(n|q)$ is the probability of the generative model decoding n conditioned on q, and $p(n)$ denotes the unconditional n-gram probability.

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

N-gram level to document level scores

How to aggregate the contribution of multiple generated n-gram identifiers to its corresponding documents?

Aggregation functions: SEAL [Bevilacqua et al., 2022]

The document-level rank score combines the n-gram level rank score $score(n, q)$ and coverage weight $cover(n, K)$:

$$score(d, q) = \sum_{n \in K^d} score(n, q)^{\alpha} \times cover(n, K),$$

where K denotes all the generated n-grams, K^d is the subset of n-grams in K that appear in d , α is a hyperparameter

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

Aggregation functions: SEAL [Bevilacqua et al., 2022]

For docid repetition problem

- Coverage weight $cover(n, K)$: Avoid the overscoring of very repetitive documents, where many similar n-grams are matched

$$cover(n, K) = 1 - \beta + \beta \frac{|set(n) \backslash C(n, K)|}{|set(n)|},$$

where β is a hyperparameter, $set(n)$ is the set of tokens in n, and $C(n, K)$ is the union of all tokens in K with top-g highest scores

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

Aggregation functions: MINDER [Li et al., 2023]

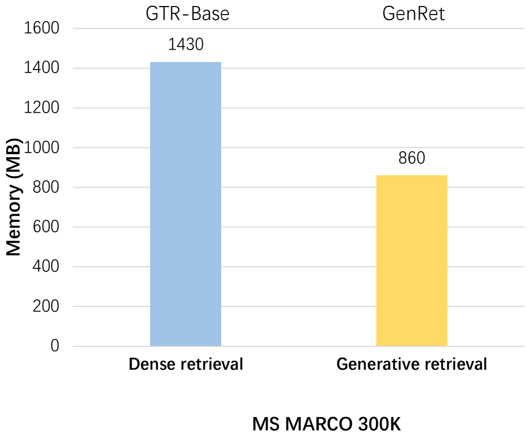
The document-level rank score: Sum of the scores of its covered docid

$$score(q, d) = \sum_{i_d \in I_d} P(i_d|q),$$

where $P(i_d|q)$ is the generated likelihood score of the docid i_d of the document d. And I_d denotes the docids generated for d

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

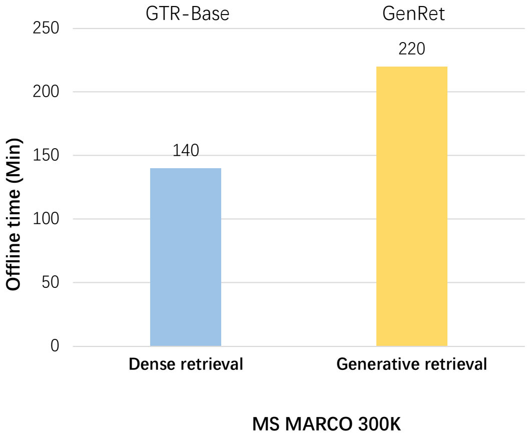
Inference efficiency: Memeory footprint



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

Data source: Sun et al. [2023]

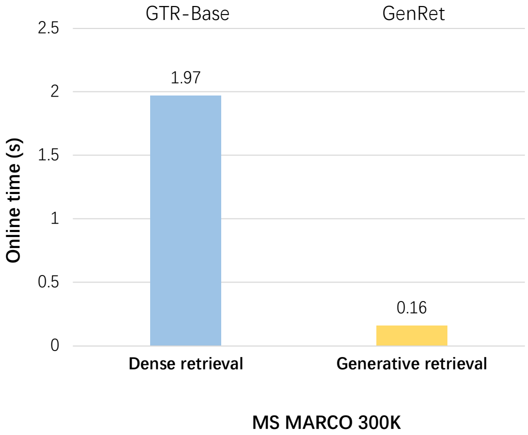
Inference efficiency: Offline latency



Data source: Sun et al. [2023]

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

Inference efficiency: Online latency



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

Data source: Sun et al. [2023]

A look back

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

Revisit: Challenges of model inference

- **How to generate valid docids?**

Revisit: Challenges of model inference

- **How to generate valid docids?**
  - Constrained generation mechanism based on prefix tree, inverted index table or FM-index

Revisit: Challenges of model inference

- **How to generate valid docids?**
  - Constrained generation mechanism based on prefix tree, inverted index table or FM-index
- **How to organize the docids for large scale corpus?**

Revisit: Challenges of model inference

- **How to generate valid docids?**
  - Constrained generation mechanism based on prefix tree, inverted index table or FM-index
- **How to organize the docids for large scale corpus?**
  - Exploiting the structured docid space

Revisit: Challenges of model inference

- **How to generate valid docids?**
  - Constrained generation mechanism based on prefix tree, inverted index table or FM-index
- **How to organize the docids for large scale corpus?**
  - Exploiting the structured docid space
- **How to generate a ranked list of docids for a query?**

Revisit: Challenges of model inference

- **How to generate valid docids?**
  - Constrained generation mechanism based on prefix tree, inverted index table or FM-index
- **How to organize the docids for large scale corpus?**
  - Exploiting the structured docid space
- **How to generate a ranked list of docids for a query?**
  - One-by-one generation based on likelihood probabilities

References

References i

M. Bevilacqua, G. Ottaviano, P. Lewis, W.-t. Yih, S. Riedel, and F. Petroni. Autoregressive search engines: Generating substrings as document identifiers. In *Advances in Neural Information Processing Systems*, pages 31668–31683, 2022.

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

Y. Li, N. Yang, L. Wang, F. Wei, and W. Li. Multiview identifiers enhanced generative retrieval. In *61st Annual Meeting of the Association for Computational Linguistics*, pages 6636–6648, 2023.

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.

P. Zhang, Z. Liu, Y. Zhou, Z. Dou, and Z. Cao. Term-sets can be strong document identifiers for auto-regressive search engines. *arXiv preprint arXiv:2305.13859*, 2023.