EMORY UNIVERSITY

Optimizing Search Efficiency: Exploring Differentiable Search Index Models for Information Retrieval

Information Retrieval

May 8, 2024

Navid Azimi, Alireza Rafiei

Department of Computer Science, Emory University

Contents

Introduction

Motivation

- The efficacy of IR systems in providing pertinent document rankings in response to user queries is paramount
- Traditional IR systems commonly employ an index-then-retrieve pipeline (not always the most efficient approach!)
- Alternative approaches like the Differentiable Search Index (DSI) aim to integrate indexing and retrieval processes into a unified model. By doing so, they offer the potential for more seamless and efficient document ranking in response to user queries.

Problems

- **Sequential nature of traditional IR methods:** Indexing occurs first, followed by retrieval
  → Latency issues, especially when dealing with large datasets or in real-time search scenarios
- **Struggle with handling dynamic or evolving datasets:** The index becomes outdated over time
  → Stale search results and diminish the user experience
- **Futile optimization efforts:** Improvements made in one stage may not directly benefit the other
  → Limit the system's ability to adapt and improve over time.

Goal

- Inspired by the Differentiable Search Index concept, our goal is to merge these traditionally separate stages and developing a unified model, designated as 'f', utilizing a sequence-to-sequence architecture, which handles user queries ('q') and employs an auto-regressive approach to generate relevant document IDs.

Contents

Dataset

The MS MARCO dataset → Pre-built index provided by **Pyserini** library

1 Building Resources

- **documents (dictionary):** document ID (docid) as key, a dictionary with field 'raw' (containing the raw text as string) as value
- **queries (dictionary):** query ID as key , a dictionary with field 'raw' (containing the raw text as string) and 'docids_list' (containing the list of correlated document IDs) as value

2 Computing Word2Vec Embeddings for queries and documents

- Using the trained Word2Vec model (on corpus data, containing all the processed 'raw' data
- **Processed 'raw' data:** Converted to lowercase, tokenized, and with stopwords and punctuation removed
- **Created 2 types of embeddings for each query/document:** emb, obtained as the average of the embeddings of all words, and first_L_emb, obtained by concatenating the embeddings of the first MAX_TOKENS words

3 Creating Datasets for Siamese Models

- **Pairwise Dataset (query, document, relevance):**
  → Designed for pairwise learning, where each sample consists of a query paired with a document
- **Triplet Dataset (query, doc+, doc-):**
  → Designed for triplet learning, where each sample comprises a query along with a positive and a negative document for training

Dataset

Example of Pairwise/Triplet Datasets

4 Creating Datasets for Sequence-to-Sequence (seq2seq) Models

- **Document Dataset (encoded document, encoded docid):**
  → Designed to facilitate training Seq2Seq models by providing documents as input sequences
- **Retrieval Dataset (encoded query, encoded docid):**
  → Designed for Seq2Seq models used in retrieval tasks. It pairs documents with corresponding queries for training

Documents and queries encodings are computed using the **T5-small tokenizer**, and we choose to pick the first **L=32** tokens for representing the documents, and the first **L=9** for representing the queries.

Example of Document/Retrieval Datasets

Contents

Baseline Models: Siamese Neural Networks

Siamese Neural Network: Neural network architecture that consists of two identical subnetworks, which have the same architecture and share the same parameters (weights)

→ Learn the similarity or dissimilarity between pairs of inputs (Differential learning approach)

1 Convolutional Siamese Neural Network (CNN-SNN) Baseline (using Word2Vec embeddings)

Steps:

- **Changing Dataset Return Type:** The dataset pairs_dataset return type is modified to 'emb' (Word2Vec embeddings)
- **Initializing Siamese Network (Parameters):**

  input_size: Set to the dimensionality of Word2Vec embeddings

  conv_channels: The number of channels for convolutional layers
- **Training the Model (Parameters):**

  pairs_dataset, siamese_net, max_epochs, batch_size, split_ratio, etc.

- Inability to effectively discriminate between relevant and random documents, assigning high similarity scores to irrelevant documents

	MAP		nDCG		Precision@10		Recall@1000	
	Train	Test	Train	Test	Train	Test	Train	Test
CNN-SNN	0.1181	0.1392			0.0399	0.0600	0.3900	0.5700

Baseline Models: Siamese Neural Networks

2 Siamese-Attention-Net Transformer Baseline (using token embeddings)

- **Attention network:** Designed to identify the highest correlations amongst words within a sentence, assuming that it has learned those patterns from the training corpus
- This enables learning representations based on token-level information (capture dependencies regardless of their distance in the input sequence.)
- Generates a relevance score using cosine similarity

Steps:

- **Changing Dataset Return Type to Token Embeddings:** The return type of the dataset pairs_dataset is adjusted to 'first_L_emb' (embeddings of first L tokens) - Use the stack of the first MAX_TOKENS embeddings (first_L_emb) of queries and documents
- **Initializing Siamese Transformer Network (Parameters):**
  embedding_size (total): Set to the product of EMBEDDING_SIZE and MAX_TOKENS
- **Training the Model (Parameters):** pairs_dataset, siamese_transformer, max_epochs, batch_size, split_ratio, etc.

- This model also struggles to distinguish relevant documents from random ones, frequently attributing high scores to both

	MAP		nDCG		Precision@10		Recall@1000	
	Train	Test	Train	Test	Train	Test	Train	Test
SNN-AT	0.0999	0.0			0.01	0.0	0.08	0.0499

Baseline Models: Siamese Neural Networks

3 Siamese Lightning module using embeddings (Contrastive Learning Approach with Triplet Loss)

- **Triplet loss function:** The network learns by comparing a set of three inputs: an anchor image, a positive image (similar to the anchor), and a negative image (dissimilar to the anchor). The goal is to bring the anchor and positive image embeddings closer while pushing the negative embedding further away.

Steps:

- **Changing Dataset Return Type to Embeddings:** The return type of the dataset `triplets_dataset` is adjusted to 'emb' (embeddings)
- **Initializing Siamese Lightning Module (Parameters):**
  input_size: Set to the dimensionality of embeddings (EMBEDDING_SIZE)
  margin: Defines the margin for the triplet loss function
  arch_type: Specifies the architecture type, set to 'linear'
- **Training the Model (Parameters):** triplets_dataset, siamese_lightning_module, max_epochs, batch_size, split_ratio, etc.

- This method, by leveraging the principles of contrastive learning, showed notable effectiveness over previous baselines and effectively discerns between similar and dissimilar query-document pairs

	MAP		nDCG		Precision@10		Recall@1000	
	Train	Test	Train	Test	Train	Test	Train	Test
SNN-AT	0.2725	0.1706			0.1659	0.1599	0.7319	0.666

Contents

DSI Transformer-Based Model

DSI Transformer-Based Model

DSI Transformer-Based Model

Contents

Conclusion

References

1 **Transformer memory as a differentiable search index**

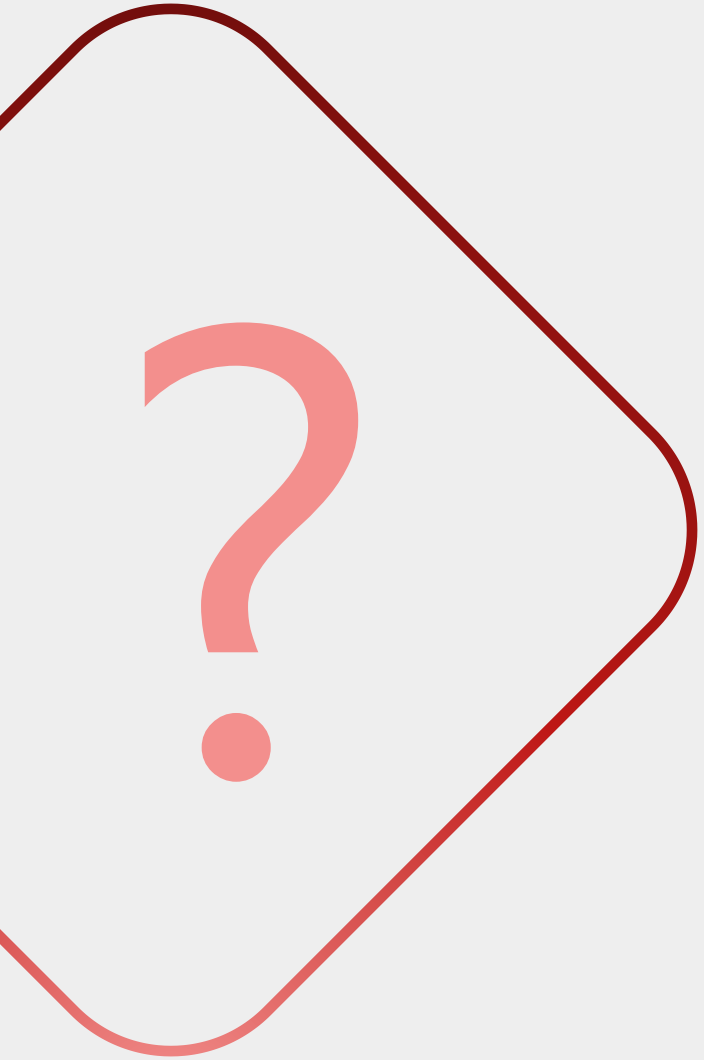
Yi Tay, Vinh Q. Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, Tal Schuster, William W. Cohen, and Donald Metzler.

2 **Fully Convolutional Siamese Networks for Change Detection**

Rodrigo Caye Daudt, Bertrand Le Saux, Alexandre Boulch

3 **Siamese Attention Networks**

Hongyang Gao, Yaochen Xie, Shuiwang Ji



Thank You!