Optimizing Guided Traversal for Fast Learned Sparse Retrieval

Yifan Qiao, Yingrui Yang, Haixin Lin, Tao Yang
Motivation

Problem

Fast top $k$ document retrieval with an inverted index using a learned sparse representation: e.g. SPLADE [Formal et al. SIGIR’21 and 22], uniCOIL, DeepImpact

Standard retrieval with dynamic index pruning: MaxScore or VBMW

Prior work: GTI [Mallia et al. SIGIR’22]

- Store both BM25 and learned weights of a document in an inverted index
- Uses BM25 based scoring to skip documents while final ranking uses a linear combination of learned neural weights and BM25 weights
Motivation

Weakness addressed

- When $k$ becomes relatively small, the relevance drops significantly, indicating BM25 based guidance for pruning is too aggressive.
- Token inconsistency in BM25 model and a learned neural model creates un-smoothed weighting and results in significant relevance drop.
Proposed Solution: 2GTI

Two level pruning guidance with different scoring and thresholding

- View pruning in standard MaxScore retrieval algorithm in two levels
  - Global level: partitioning of the non-essential and essential terms
  - Local level: skipping a document selected during possible deep visitation
- Allow different scoring/thresholding at these two levels and at final ranking

![Diagram showing global and local levels of pruning]

- 2GTI on VBMW is similar

Dept. of Computer Science
Proposed Solution: 2GTI

Maintain 3 top $k$ queues with different rankings

- $Q_{Gl}$ uses ranking $R_\alpha$ for global pruning: $Global(d) = \alpha \cdot w_{BM25} + (1 - \alpha) \cdot w_{learned}$
- $Q_{Lo}$ uses ranking $R_\beta$ for local pruning: $Local(d) = \beta \cdot w_{BM25} + (1 - \beta) \cdot w_{learned}$
- $Q_{Rk}$ uses ranking $R_\gamma$ for final ranking: $RankScore(d) = \gamma \cdot w_{BM25} + (1 - \gamma) \cdot w_{learned}$

Maintain 3 thresholds for dynamic index pruning

- $\theta_{Gl}$ for essential term partitioning based on $R_\alpha$
- $\theta_{Lo}$ for minimum top $k$ score based on $R_\beta$
- $\theta_{Rk}$ for top $k$ thresholding based on final ranking $R_\gamma$. 

---

Dept. of Computer Science
Relevance Properties of 2GTI

**Objective:** Analyze relevance behavior of 2GTI formally and its competitiveness

(GTI is a special case of 2GTI with $\alpha = \beta = 1$)

**#1:** Top documents agreed by top $k$ of each ranking $R_\alpha$, $R_\beta$, and $R_\gamma$ are kept on the top $k$ by 2GTI.
Relevance Properties of 2GTI

#1: Top documents agreed by top $k$ of each ranking $R_\alpha$, $R_\beta$, and $R_\gamma$ are kept on the top $k$ by 2GTI.

#2: Properly configured 2GTI can outperform the two-stage algorithm $R_2$: retrieval with $R_\alpha$ and re-ranking with $R_\gamma$.

(1) When $\alpha = \beta$ or $\beta = \gamma$, the average rank score of the top $k$ positions produced by 2GTI is equal or higher than this two-stage algorithm $R_2$.

$$\sum_{d \in 2GTI} RankScore_\gamma(d) \geq \sum_{d \in R_2} RankScore_\gamma(d)$$

(2) When $R_\gamma$ outmatches $R_\beta$ which outmatches $R_\alpha$, 2GTI retrieves equal or more relevant results at top $k$ positions than $R_2$.

$Recall@k(2GTI) \geq Recall@k(R_2)$
### MS MARCO Passage

<table>
<thead>
<tr>
<th>MS MARCO Passage</th>
<th>( k = 10 )</th>
<th>( k = 1000 )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dev</strong></td>
<td><strong>MRR@10</strong></td>
<td><strong>MRT (( P_{99} ))</strong></td>
</tr>
<tr>
<td>SPLADE+++-Original</td>
<td>0.3937</td>
<td>121 (483)</td>
</tr>
<tr>
<td>-GTI</td>
<td>0.2687</td>
<td>118 (440)</td>
</tr>
<tr>
<td>-2GTI-Accurate</td>
<td>0.3939</td>
<td>31.1 (171)</td>
</tr>
<tr>
<td>-2GTI-Fast</td>
<td>0.3934</td>
<td>22.7 (116)</td>
</tr>
</tbody>
</table>

- On MS MARCO Passage Dev, \( k = 1000 \)
  - 2GTI-Accurate produces slightly higher MRR@10 (due to BM25 interpolation) than the original SPLADE while being 2.6x faster
  - 2GTI-Fast has similar MRR score while being 6.5x faster

- Similar trend observed in TREC DL’19 and DL’20
Token and weight alignment between BM25 index and learned index

For those missing weights in the BM25 model
  - /0: do nothing
  - /1: fill with 1
  - /s: fill with learned scores scaled by ratio of mean values of non-zero weights

<table>
<thead>
<tr>
<th>Weight alignment for GTI ($\alpha = 1, \beta = 1, \gamma = 0.05$)</th>
<th>MRR@10</th>
<th>Recall@10</th>
<th>MRT</th>
<th>$P_{99}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTI/0</td>
<td>0.2687</td>
<td>0.5209</td>
<td>118</td>
<td>440</td>
</tr>
<tr>
<td>GTI/1</td>
<td>0.3036</td>
<td>0.5544</td>
<td>26.7</td>
<td>114</td>
</tr>
<tr>
<td>GTI/s</td>
<td>0.3468</td>
<td>0.5774</td>
<td>9.1</td>
<td>36.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Weight alignment for 2GTI-Accurate ($\alpha = 1, \beta = 0, \gamma = 0.05$)</th>
<th>MRR@10</th>
<th>Recall@10</th>
<th>MRT</th>
<th>$P_{99}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2GTI/0</td>
<td>0.3933</td>
<td>0.6799</td>
<td>328</td>
<td>1262</td>
</tr>
<tr>
<td>2GTI/1</td>
<td>0.3933</td>
<td>0.6818</td>
<td>89.3</td>
<td>393</td>
</tr>
<tr>
<td>2GTI/s</td>
<td>0.3939</td>
<td>0.6812</td>
<td>31.1</td>
<td>171</td>
</tr>
</tbody>
</table>

Faster & more accurate

10.5x faster
**Evaluation**

Zero-shot performance (13 BEIR datasets)

<table>
<thead>
<tr>
<th>BEIR</th>
<th>$k = 10$</th>
<th>Avg. Speedup</th>
<th>$k = 1000$</th>
<th>Avg. Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPLADE++-Original</td>
<td>0.500</td>
<td>-</td>
<td>0.500</td>
<td>-</td>
</tr>
<tr>
<td>-GTI/s</td>
<td>0.430</td>
<td>6.1x</td>
<td>0.496</td>
<td>2.1x</td>
</tr>
<tr>
<td>-2GTI/s-Fast</td>
<td>0.499</td>
<td>2.0x</td>
<td>0.501</td>
<td>2.5x</td>
</tr>
</tbody>
</table>

**Efficiency-driven SPLADE**

- Apply 2GTI on the efficiency-driven SPLADE model [Lassance et al. SIGIR’22] with a relevance tradeoff ($k = 10$)

<table>
<thead>
<tr>
<th>BT-SPLADE-L</th>
<th>MRR@10</th>
<th>Recall@10</th>
<th>MRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original MaxScore</td>
<td>0.3799</td>
<td>0.6626</td>
<td>17.4</td>
</tr>
<tr>
<td>2GTI/s ($\alpha=1$, $\beta=0.3$, $\gamma=0.05$)</td>
<td>0.3772</td>
<td>0.6584</td>
<td>8.0</td>
</tr>
<tr>
<td>GTI/s ($\alpha = \beta = 1$, $\gamma = 0.05$)</td>
<td>0.3284</td>
<td>0.5520</td>
<td>6.6</td>
</tr>
</tbody>
</table>
Conclusions

- 2GTI retrieval manages 3 top $k$ queues with 3 linear combinations of neural and BM25 weights to rank/skip docs
  - Pruning decision is more accurate than GTI
  - Can outperform a two-stage retrieval algorithm at least

- Sample configurations for SPLADE++:
  - $R_\alpha: \alpha \cdot w_{BM25} + (1 - \alpha) \cdot w_{learned}$, with $\alpha = 1$
  - $R_\beta: \beta \cdot w_{BM25} + (1 - \beta) \cdot w_{learned}$, with $\beta = 0$ or 0.3
  - $R_\gamma: \gamma \cdot w_{BM25} + (1 - \gamma) \cdot w_{learned}$, with $\gamma = 0.05$

- Smooth weight alignment is necessary to address token inconsistency between BM25 and neural models

- For MS MARCO passages with SPLADE++, 5x to 7x faster than original MaxScore and GTI

Thanks and Q/A?