A Dense Subset Index for Collective Query Coverage

Kartik Nair[12], Pritish Chakraborty[1], Atharva Tambat[1], Indradyumna Roy[1], Soumen Chakrabarti[1], Anirban Dasgupta[3], and Abir De[1]

[1]IIT Bombay [2]Carnegie Mellon [3]IIT Gandhinagar

Background: Traditional Vector-Bag Retrieval

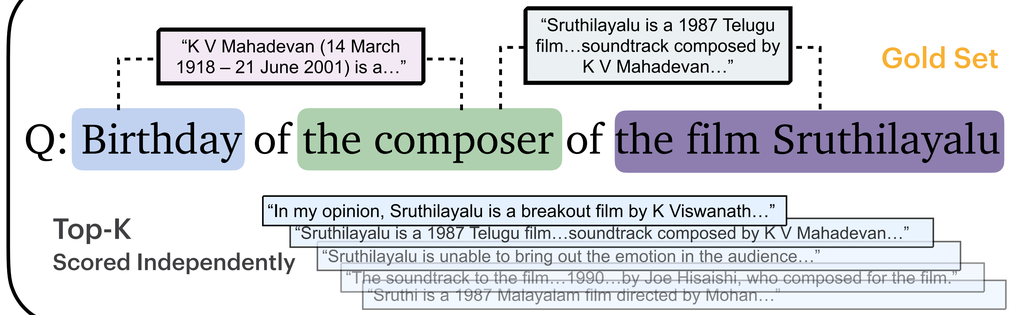
Consider bag-of-words scoring (MaxSim) popularized by ColBERT [Khattab et al. 2020] — given a query and corpus as sets of **token embeddings**

$$Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_M\} \qquad X = \{\mathbf{x}_1, \ldots, \mathbf{x}_L\}$$

$$F(X_c, \mathbf{q}) = \max_{\mathbf{x} \in X_c} \mathbf{q}^T \mathbf{x}$$

Traditional retrieval is *competitive*: documents are scored *against* each other to rank higher.

$$\mathrm{S} \leftarrow \text{top-k}_{X \in C}\, F(X, Q)$$



Collaborate, not compete

Collective scoring is a natural extension to MaxSim. Scoring a set of documents $S = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k\}$ against the query.

$$F(S, Q) = \sum_{\mathbf{q} \in Q} \max_{\mathbf{x} \in \cup_{s \in S} X_s} \mathbf{q}^T \mathbf{x} \quad = \sum_{\mathbf{q} \in Q} \max_{s \in S} \max_{\mathbf{x} \in X_s} \mathbf{q}^T \mathbf{x}$$

Coverage Objective:

$$\max_{S \subseteq C} F(S, Q) \quad s.t.\ |S| \leq K$$

This objective is sub modular in S, hence admits a $(1 - 1/e)$ greedy approximation with the marginal gain function $F(c \mid S, Q)$

1: Initialize $S_0 \leftarrow \emptyset$
2: **for** $k = 1, \ldots, K$ **do**
3: $\quad c_k = \arg\max_{c \in C \setminus S_{k-1}} F(c \mid S_{k-1}, Q)$
4: $\quad S_k \leftarrow S_{k-1} \cup \{c_k\}$
5: **return** S_K

The greedy solution requires exhaustive scoring, which **grows linearly** in complexity with the size of corpus.

We can instead **formulate this as a first stage retrieval problem**!

Adaptive Probing : updated marginal gain at each step

$$S_0 = \varnothing \xrightarrow[F(\bullet|S_0,Q)]{\text{probe with}} S_1 \xrightarrow[F(\bullet|S_1,Q)]{\text{probe with}} S_2 \to \ldots \to S_K$$

Challenge: adapt and/or approximate score for retrieval

(1) Separate into corpus-dependent term for indexing
(2) Convert to inner-product form

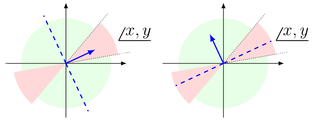
Making the Marginal Gain Index-able

1. Difference to dot product: separating state dependence

$$F(c \mid S, Q) = \sum_{\mathbf{q} \in Q} [F(X_c, \mathbf{q}) - F(S, \mathbf{q})]_+ \quad = \sum_{\mathbf{q} \in Q} \max_{\mathbf{x} \in X_c} \left[\underbrace{\begin{bmatrix} \mathbf{q} \\ F(S, \mathbf{q}) \end{bmatrix}}_{=:\hat{\mathbf{q}}_S}{}^{\top} \overbrace{\begin{bmatrix} \mathbf{x} \\ -1 \end{bmatrix}}^{=:\hat{\mathbf{x}}} \right]_+$$

state dependence absorbed by the query

2. Dealing with the hinge score with sign-hashing $\quad w \sim \mathcal{N}(0, \mathbf{I}_{d+1})$



Randomised Feature Map $\quad \phi_w(x) = \frac{1}{\sqrt{2}} \begin{bmatrix} x \\ \text{sign}(w^\top x) x \end{bmatrix}$

$$\Pr\left(\phi_w(x)^\top \phi_w(y) = [x^\top y]_+\right) = \frac{\blacksquare}{\blacksquare + \blacksquare} \geq \frac{1}{2}$$

We can improve our approximation by drawing multiple hyperplanes $\{w_1, w_2, \ldots, w_R\} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \mathbf{I}_{d+1})$

This yields a new scoring function $G_{1:R}$ $\quad \Pr\left(\max_{r \in [R]} \phi_{w_r}(x)^\top \phi_{w_r}(y) = [x^\top y]_+\right) \geq 1 - \frac{1}{2^R}$

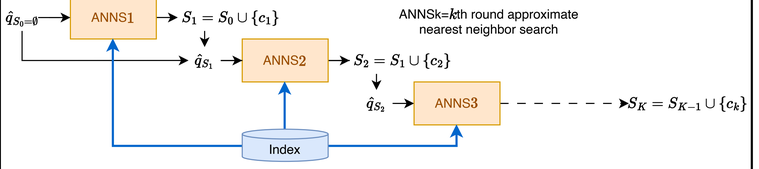
$$G_{1:R}(c|S, Q) = \sum_{\mathbf{q} \in Q} \max_{\mathbf{x} \in X_c} \left[\max_{r \in [R]} \phi_{w_r}(\hat{\mathbf{q}}_S)^\top \phi_{w_r}(\hat{\mathbf{x}}) \right]$$

Replacing the marginal with $G_{1:R}$ gives us a $(1 - 1/e - \delta)$-approximation where $\delta \leq |Q|/2^R$

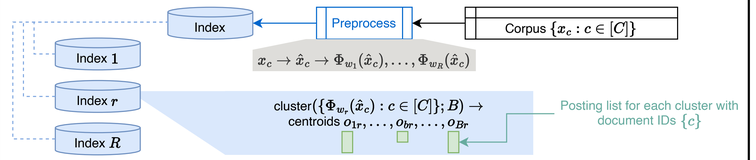
However, indexing this now requires R Replica indexes: with each replica storing the corresponding feature-mapped document tokens $\{\phi_{w_r}(\hat{\mathbf{x}}) | \mathbf{x} \in X\}$

The Retrieval Pipeline

Each greedy step is replaced by an approximate nearest neighbour search



Each replica index has a token-level IVF, with each token stored as a centroid and quantized residual.

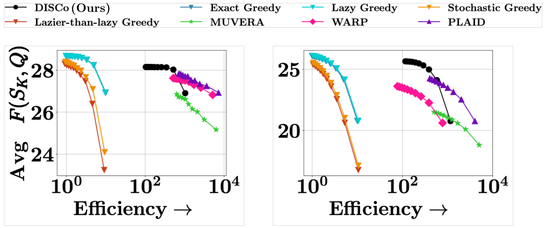


Retrieval consists of a token level lookup followed by progressive levels of candidate pruning

Experiments

We compare our method (DISCo) against variants of greedy sub modular solvers, and traditional multi-vector retrieval engines.

Mean Coverage Score



MAP on Multi-Hop QA

Dataset ↓ Method →	DISCo (Ours)	Exact Greedy	Lazy Greedy	PLAID	WARP
2WikiMultiHopQA	0.90	**0.91**	**0.91**	0.89	0.82
HotpotQA	**0.84**	0.83	0.83	0.81	0.77

Author Contact: ksnair@cs.cmu.edu, {pritish,indraroy15,soumen,abir}@cse.iitb.ac.in