Reconfigurable Inverted Index

Yusuke Matsui (NII)  Ryota Hinami (UTokyo)  Shin’ichi Satoh (NII)

**Summary**
- Approximate nearest neighbor search
- Solved a subset-search problem
- Comparative performance with IVFPQ (Faiss); 10 ms for \( N = 10^9 \)

**Subset search problem**

- Approximate NN Search
  \[
  \min_{\mathbf{x} \in \mathbb{R}^D} \| \mathbf{q} - \mathbf{x} \|_2
  \]
  \[
  \mathbf{x} \in \mathbb{R}^D
  \]
  \[
  \mathbf{q} \in \mathbb{R}^D
  \]

**Preliminary 1: Product Quantization (PQ) linear scan [Jégou+, TPAMI 11]**

- Compress a vector into a short code
  \[
  \mathbf{q} \rightarrow \mathbf{c}
  \]
  \[
  \mathbb{R}^4 \rightarrow \{0, 1, 2, \ldots \}^2
  \]

**Preliminary 2: Inverted Index + PQ (IVFPQ) [Jégou+, TPAMI 11]**

- Inverted index over PQ-encoded vectors
  - Find the closest space: \( k^* = \min_k \| \mathbf{q} - \mathbf{c}_k \|_2 \)
  - Focus the \( k^* \) th space, filter out items \( \notin S \)
  - Re-rank the items via PQ-linear scan

**Proposed: Reconfigurable inverted index (Rii)**

- Store (1) PQ-codes linearly, and (2) IDs as an inverted index
- Can run either PQ-linear-scan or IVFPQ with a single data structure

**Evaluation (SIFT1M, top-R search)**

- The existing system is slow, especially when \( |S| \) is small
- Proposed Rii is always fast regardless of \( |S| \) and \( R \)

**Codes:**  
https://github.com/matsui528/rii

```python
import rii
import nanopq

# pip install rii

# Prepare a PQ/OPQ code with M=32 sub spaces
code = nanopq.PQ(M=32).fit(vecs=Xt)  # Trained using Xt

e = rii.Rii(fine_quantizer=code)

e.addConfigure(vecs=X)

e.search(idsq, topk=3, target_ids=s)  # e.g., [7484 8173 1556] [15.0 15.3 16.1]
```