Can Learned Indexes be Built Efficiently? A Deep Dive into Sampling Trade-offs

Minguk Choi, Seehwan Yoo, and Jongmoo Choi
Dankook University, South Korea
{mgchoi, seehwan.yoo, choijm}@dankook.ac.kr

Background

- **Learned Index Structure**
  - Index structure employs machine learning techniques
  - View the index as a model that predicts the position of a key

- **Performance of Learned Index: Space-efficient**
  - Pareto optimal in terms of index size and lookup latency in read-only
  - No alternative exists that has both a smaller size and lower latency

Motivation

- **Long Index Build Time**
  - Up to about 2,000x slower than traditional indexes
  - But still there are application where build time is crucial (e.g., LSM-tree)

- **Why Building the Learned Index is Slow?**

  - 1) Complete traversal and training
  - 2) Higher per-element training overhead
  - But it’s still longer than traditional indexes

  - This study began with the question ...

  Since the learned index uses the model, Can’t it learn efficiently even with less data?

Evaluation

1. **Sampling Trade-offs**
   - Sampling interval (l) → (a) build time ↓
   - Each error-bound (ε) has threshold interval (l(t))
     - Until l(t), (b–i) rest of metrics remain consistent
     - After l(t), # of segments ↑
     - (b) Size ↑, (d) Height ↑
     - (e) Pred. cache miss ↑
     - (f) Pred. latency ↑
   - After l(t), # of segments ↑ → (g) MSE ↓
     - (h) Corr. cache miss ↓, (i) Corr. latency ↓

2. **Design Space of Learned Indexes**
   - Without sampling, absence of trade-offs between build, size, and lookup
   - Sampling introduce trade-offs between build, size, and lookup
   - Broaden design space of learned indexes from 2D to 3D

3. **Build Speed-up**
   - Explore Safe down-sampling, where size & lookup latency increased by less than 5%*
   - Max build speedup without performance loss
     - sRMI: 1/44,514, sPGM: 1/14,479, sRS: 1/14,479

4. **Pareto Analysis**
   - Can learned indexes be built more efficiently in terms of build time and lookup latency than traditional indexes through sampling?
   - To the best of our knowledge, this is first to show that learned indexes are also Pareto optimal in terms of build time and average (and tail) lookup latency

Design

- **Our Approach: Sampling**
- **Challenges**
  1. Losing the error-bound property due to sampling loss
  2. Complex trade-offs in terms of model, index, and micro-architecture
  3. Absence of benchmark for sampling applied indexes

1. **Error-bound Preserving Sample Learning Algorithm**
   - EB-PLA (Error-bounded Piece-wise Linear Approximation) Model
     - Train all keys with error-bound ε → V_k, Error(k) ≤ ε
     - Train sample kth keys with the error-bound ε → V_k, Error(k) ≤ ε
   - **Sample EB-PLA Algorithm**
     - (c) Refine the error-bound due to sampling loss → V_k, Error(k) ≤ ε’(= ε + l – 1)
     - Preserve the error-bound property
     - (d) Replace the sample learning error-bound to δ(= ε – l + 1) → V_k, Error(k) ≤ ε
     - Preserve the error-bound (ε) by learning less data with smaller & stricter error-bound (δ)
   - **Sample EB-Histogram**
   - PLR with Simple Linear Regression

2. **Internal Changes due to Sampling**
   1) Dynamic Segmentation (Key range of each segment is different)
      - Aggressive sampling can increase the number of segments
   2) Fixed Segmentation (Key range of each segment is equal)
      - Aggressive sampling can increase the number of under-fitting segments

3. **Unified Sampling Algorithm & Implementation**
   - **BASIL** (Benchmark of Sampling Applied Learned Indexes)
     - 1) Unified Sampling Algorithm: Systematic Sampling
       - Extract every 2εth key form to last key (l-sampling interval)
     - 2) Unified Sampling Implementation
       - Index access and train only sample key-value data from entire dataset

4. **Paradigm**
   - Use learned index for application (e.g., analytics)
   - With learning, expand index to any size and latency
   - Real-time learning: can adjust index to specific application

---

(The Case for Learned Index Structures, SIGMOD '18)