Revisiting Data Prefetching for Database Systems with Machine Learning Techniques

Yu Chen, Yong Zhang, Jiacheng Wu, Jin Wang, Chunxiao Xing

Presenter: Xiling Li

Northwestern | McCORMICK SCHOOL OF ENGINEERING

Overview

- Access page in memory is faster than access in the disk.
- Prefetch (Predict + fetch):
  - Predict future page access patterns
  - Fetch pages into buffer pool ahead before being accessed.
  - Existing methods are mostly heuristic-based methods
    - E.g. One Block Lookahead (OBL)
    - Inefficiently prefetch pages with random access by indexing
- Machine Learning (neural-network based approach)
  - Learning page access patterns from history
  - Formalize prefetching as a classification problem

Formulation

- Input:
  - A sequence of pages previously accessed
  - Represent as a sequence of page offsets (d1,d2,...,dn)

- Output:
  - A page in the disk
  - Page offset dt

- Classification
  - Get output distribution by softmax function

Sparsity Issue

- Sparsity of target space
  - E.g. TPC-H approximately contains 10^7 target pages
  - **Bad: Locate a page by its universal id (di)**

- Extent in MySQL can be a feasible solution
  - An extent is a group of consecutive pages within a data file
  - # of extents is much smaller than # of pages
  - **Better: Locate a page by extent it belongs and the offset in the extent**

Architecture

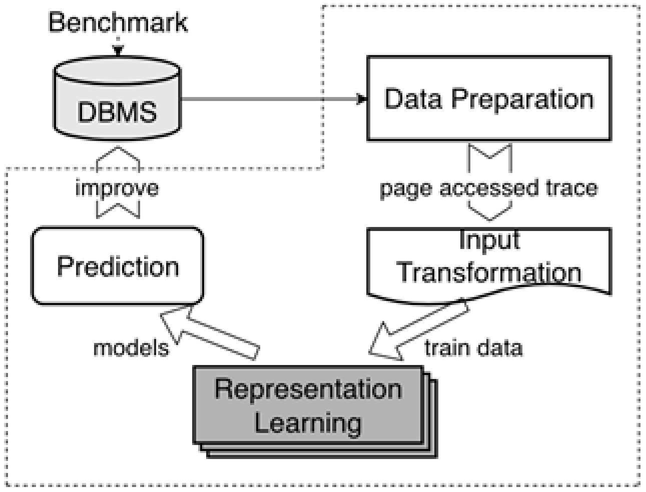

Fig. 1: Overall Architecture

- Data preparation
  - Collect access traces by running some queries
- Input transformation
  - Transform page offsets di into extent
  - **Overcome sparsity issue**
- Representation learning
  - Learn access pattern by traces
- Prediction
  - Fetch corresponding pages from disk

Transformation

- Transform a page offset di into extent offset ei and in-extent offset fi

$$e_i = \lfloor \frac{d_i}{\#\text{ pages in a extent}} \rfloor \qquad (1)$$

$$f_i = d_i - e_i \times \#\text{ pages in a extent} \qquad (2)$$

- In MySQL, each extent has 64 pages
  - E.g. if di = 76, ei = 1 and fi =12

- Triplet ti = <di, ei, fi> becomes input of latter NN models.

Single-model Framework

- Prediction model
  - Convolutional Neural Network
  - Recurrent Neural Network
  - Long Short-Term Memory
- Input: a sequence of ti
- Output: on = {en, fn}
- Loss function
  - Sum of two cross-entropies

$$\mathcal{L}(p(o_n), p(\hat{o}_n)) = -\sum_{e_n} p(\hat{e}_n) \log p(e_n) - \sum_{f_n} p(\hat{f}_n) \log p(f_n)$$

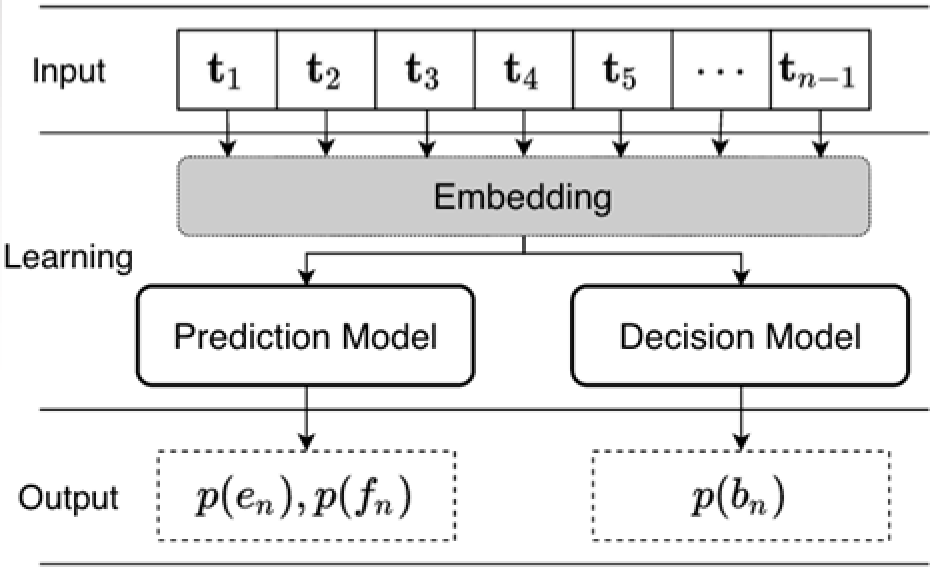

Fig. 2: Single Model with Prefetching Decision

Single-model Framework - Cont’d

- Avoiding wrong prefetching is to prevent extra dish I/O.
- Use decision model to decide if DBMS needs fetch or not at a certain timestamp n
- Input: a sequence of ti
- Output: bn = {0,1}
- Predict K pages for decision

Algorithm 1: Training for Decision Model

Input: Prediction Model $\mathcal{M}$, Input Representation t_i.
Output: Decision Model $\mathcal{D}$.

$train\_set = \emptyset$.
foreach *timestep* n **do**
  Obtain the label $\hat{o}_n$.
  Obtain $p(o_n)$ by applying $\mathcal{M}$ on input series.
  Select K page offsets from $p(o_n)$ as collection $\mathcal{S}$.
  if $\hat{o}_n \in \mathcal{S}$ **then**
    Mark timestep n with positive label.
  else
    Mark timestep n with negative label.
  Add timestep n into $train\_set$.
Train Decision Model $\mathcal{D}$ on $train\_set$.
return $\mathcal{D}$.

Multi-model Framework

- TXNs from different SQL templates produce distinct access patterns
- Same architecture with different parameters for prediction models
- Classification model provides probability for each model to be chosen
- Output distribution $\{\bar{p}(e_n), \bar{p}(f_n)\} = \{\sum_{j=1}^{G} c_j \cdot p_j(e_n), \sum_{j=1}^{G} c_j \cdot p_j(f_n)\}$
- Same loss function

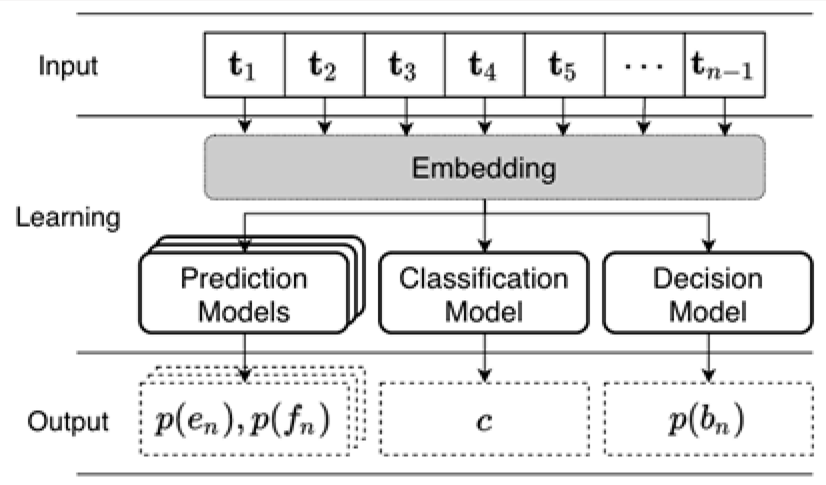

Fig. 3: Multi-Model Framework

Evaluation

TABLE I: Model Comparison

	TPC-H		TPC-DS		SSB	
	Precision(%)[1]	Recall(%)	Precision(%)	Recall(%)	Precision(%)	Recall(%)
LookAhead	20/-	81	7/-	81	22/-	88
Random	14/28	80	7/15	78	40/55	85
DNN	41/75	77	33/78	79	62/69	82
CNN	41/85	70	40/79	81	80/74	86
RNN	33/64	63	29/62	70	46/62	73
LSTM	33/64	63	30/62	71	46/63	73
Multi-Model	**76/87**	**82**	**78/87**	**94**	**87/84**	**94**

[1] We measure 2 precisions: Overall-Precision / Decision-Precision. Overall-Precision means we measure the prefetcher's precision all the timesteps, and Decision-Precision only counts when the prefetcher decides to make a prefetching.

Thank you!

Northwestern | McCORMICK SCHOOL OF ENGINEERING