Learned Buffer Replacement for Database Systems

Donghyun Sohn

Northwestern

Contents

- Motivation
- Idea
- Example
- Experiments
- Discussion

Motivation

- Problems of existing eviction algorithms
  - LRU, CLOCK, ARC
  - Cannot adapt to workload changes
- Demonstration of ML as an efficient for prediction
  - Machine translation
- Success of improving the CPU cache efficiency by using ML
  - Computer Architecture - LeCaR

Idea

- Supposition :
  - Prediction should be based on the past access behavior of the page itself and past accesses of other pages coming along with it
- Makes problem similar with word prediction in NLP
  - Adopt Transformer

Architecture

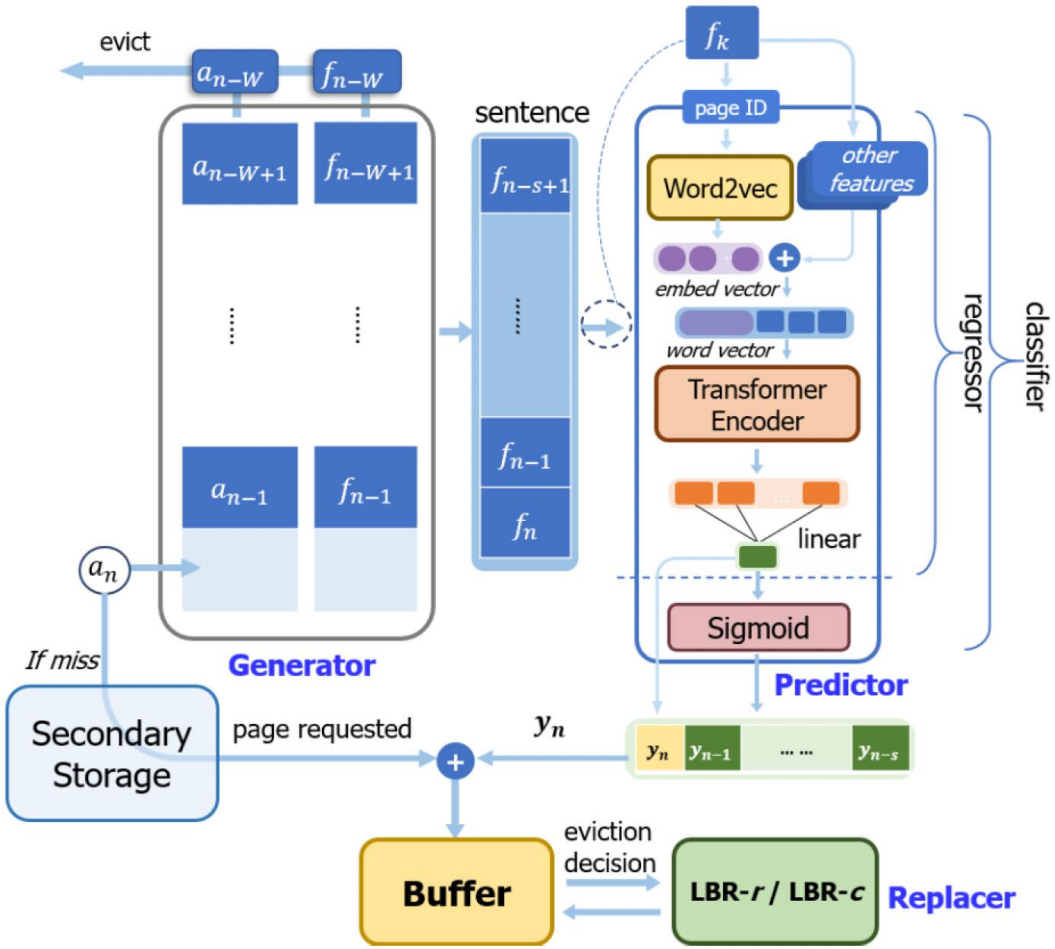

Figure 1: Architecture of the Learned Buffer Replacement.

Northwestern

LBR-c : Classifier-based

- Buffer replacement problem : Binary classification
  - Criteria : Hawkeye
  - The requested page will be hit in the future, or to cause a miss

  -> 1 : Accesses that are expected to cause a hit

  -> 0 : Accesses that cause a miss
- Train the Transformer network for classification
  - take Transformer's output as the label.
  - Replacer manage the buffer according to the labels

LBR-r : Regressor-based

- Predictor : trained to predict the next visit time for each page
  - Predicts the distance D from current access to the next access
- Replacer : chooses the page with the largest next visit time as victim

Feature Learning

- Page ID : identifies the page it accesses
  - useful when the content of a page is mostly fixed, therefore stable mapping between page ID and page content
- Frequency / LFU
  - rise of one page could mean that more visits to it
- Reuse Distance (rd) / LRU
  - Number of accesses between current access and the last access to the same page
- Penultimate Reuse Distance (prd)
  - Used for a buffer to acquire scan resistance

Example

Access		Generator	Feature			
			page ID	*frequency*	*rd*	*prd*
1	a	a	a	1	4	0
2	b	a b	b	1	4	0
3	c	a b c	c	1	4	0
4	b	a b c b	b	2	2	4
5	a	a b c b a	a	1	4	0
6	a	a b c b a a	a	2	1	4

rd window *frequency* window

Figure 2: Features for access sequences ($W = 4, w = 3$).

Experiments

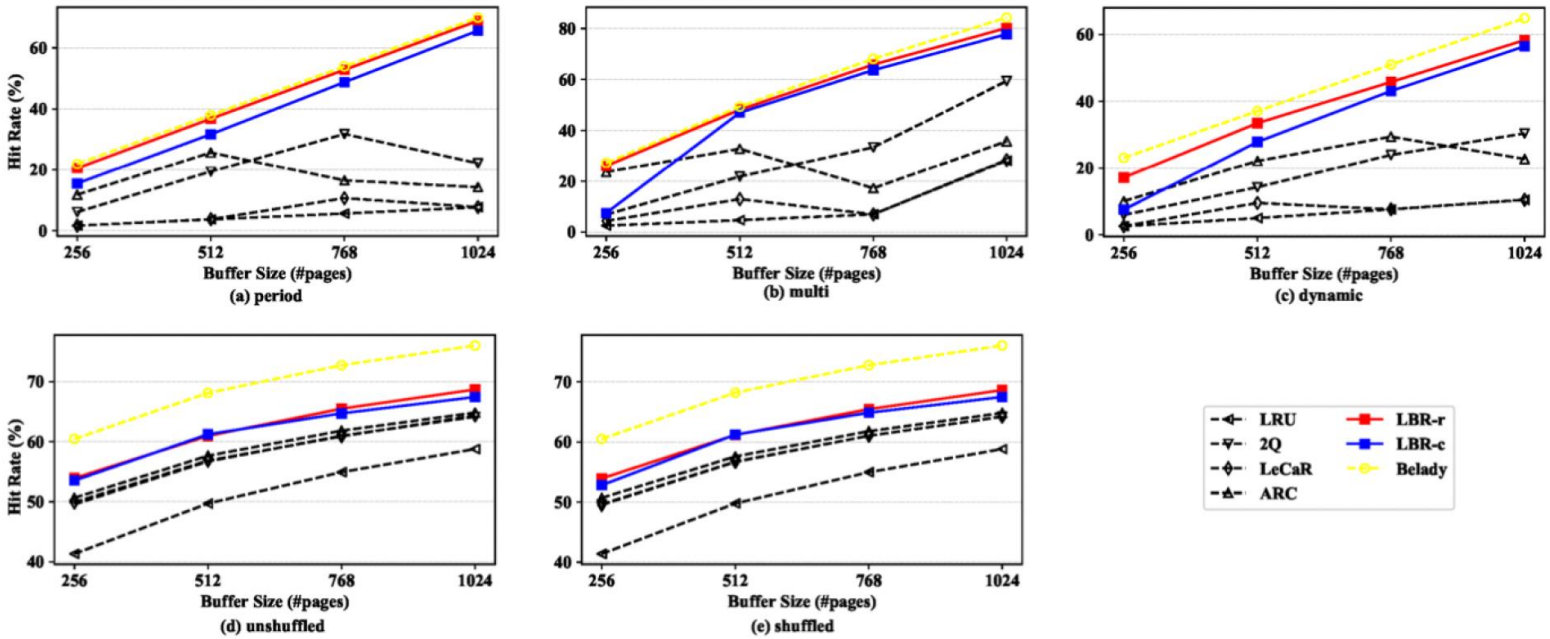

Figure 3: Comparison of the hit rate of LBR and baselines.

Discussion

- Find some pre-fetch algorithm from other fields
  - Computer architecture CPU cache

- Can we use the Predictor part of this architecture

Thank you!

Northwestern