A Hierarchical Neural Model of Data Prefetching

Siting Ma

Introduction

- Data prefetching presents two challenges to machine learning:
- 1. class explosion problem
- 2. labeling problem

- Previous Notable Work: Hashemi, delta correlations
- This Paper's Goal: Voyager, delta & address correlations

Introduction

- Data prefetching presents two challenges to machine learning:
- 1. class explosion problem
- 2. labeling problem

- 1. decompose address prediction into two sub-problems:
  - page prediction
  - offset prediction (offset aliasing problem)
- 2. multi-label training scheme

Problem Formulation

- How should we think of Data Prefetching?

  Data prefetching – a probabilistic prediction problem

  Output – a probability distribution

- Address Correlation

  P(Addr | $Access_1$, $Access_2$, ..., $Access_t$)
- Delta Correlation

  P($Stride_{t+1}$ | $Stride_t$)

* output label, input features

Previous Work – Address Correlations

- **STMS**, predict next addr in global memory access stream

  $P(Addr_{t+1} \mid Addr_t)$

- **ISB**, predict next addr by same program counter (PC localization)

  $P(Addr_{PC} \mid Addr_t)$

- **Domino**, use previous two addresses

  $P(Addr_{t+1} \mid Addr_{t-1}, Addr_t)$

Previous Work – Delta Correlations

- **IP stride prefetcher**, use a per-PC output (PC localization)

  $P(Stride_{PC} \mid Stride_{t})$

- **VLDP prefetcher**, use history of past deltas

  $P(Stride_{t+1} \mid Stride_{t0}, Stride_{t1}, \ldots, Stride_{tn})$

- **Hashemi neural prefetcher**, use history length of l past deltas

  $P(Stride_{t+1} \mid Stride_{t-l}, Stride_{t-l+1}, \ldots, Stride_{t})$

Voyager Approach

- Hierarchical Neural Structure:
  - one part predicting page addresses;
  - other part predicting offsets.
- Vocabulary: includes both addresses and deltas.
- Multi-label training scheme:
  - instead of predicting the next address in the global address stream
  - predict the most predictable address from multiple possible labels

* A neural network's vocabulary is the set of words that the model can admit as input and can produce as output.

Voyager Approach

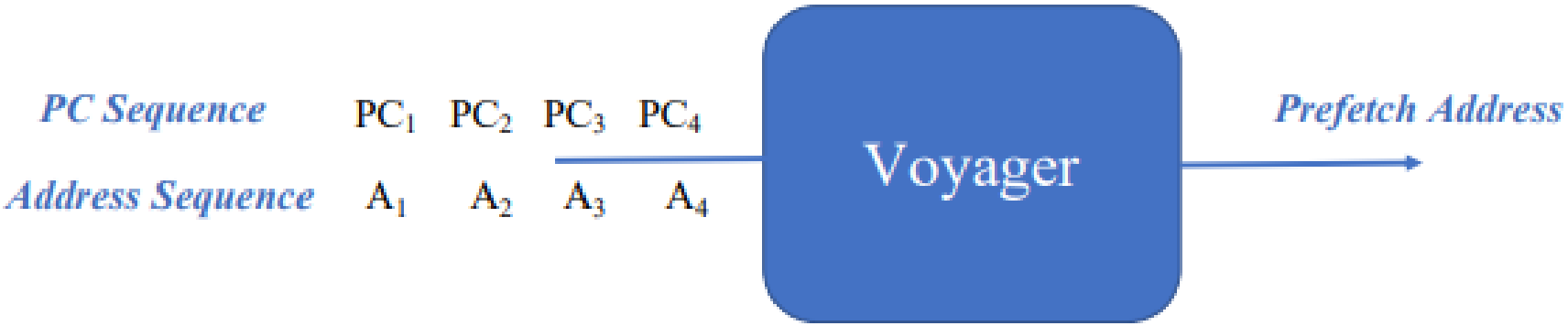

Figure 1: Overview of Voyager.

Voyager Approach

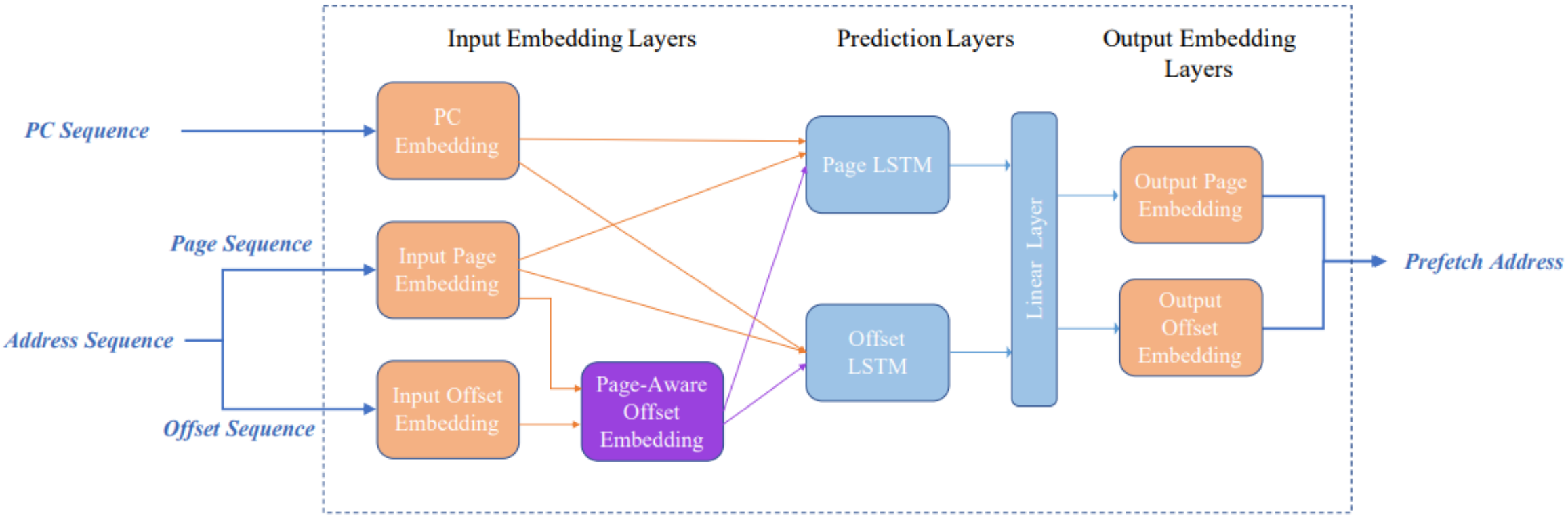

Figure 2: Voyager's Model Architecture.

Results

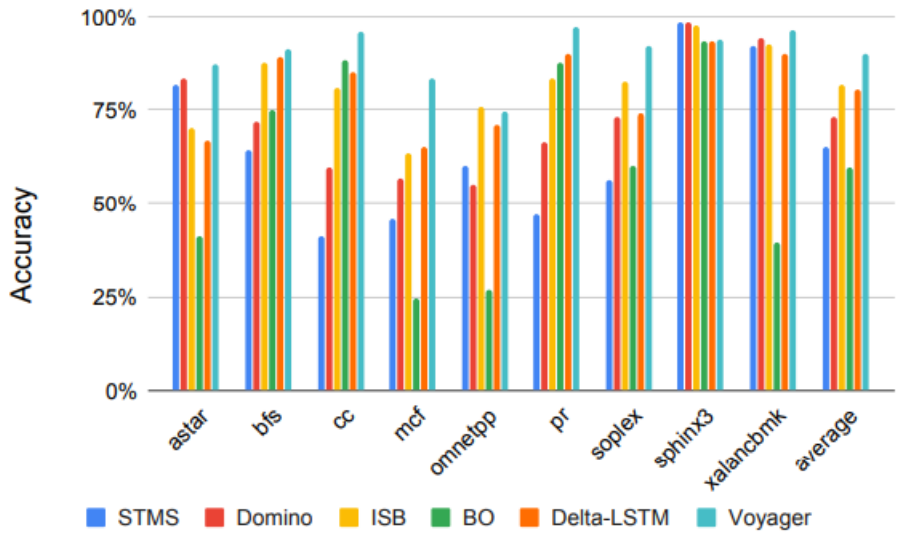

Figure 5: Accuracy.

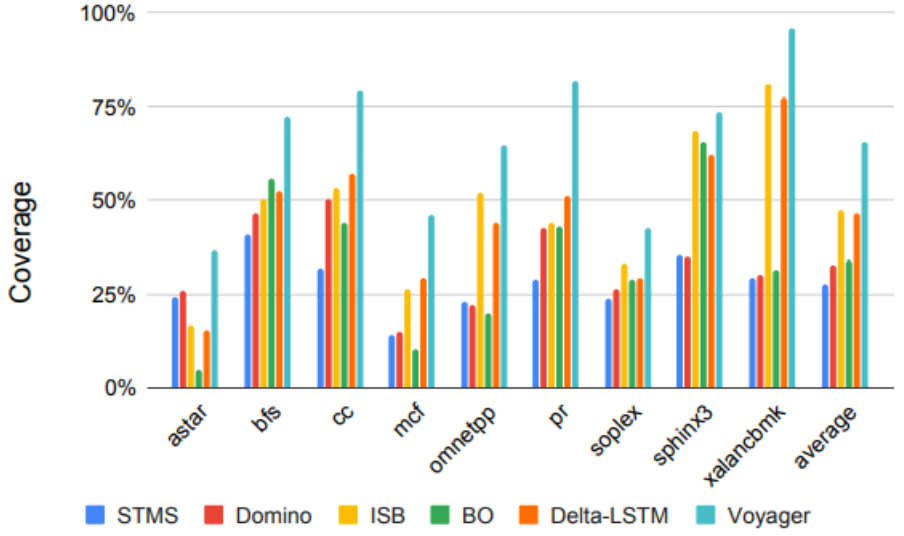

Figure 6: Coverage.

Results

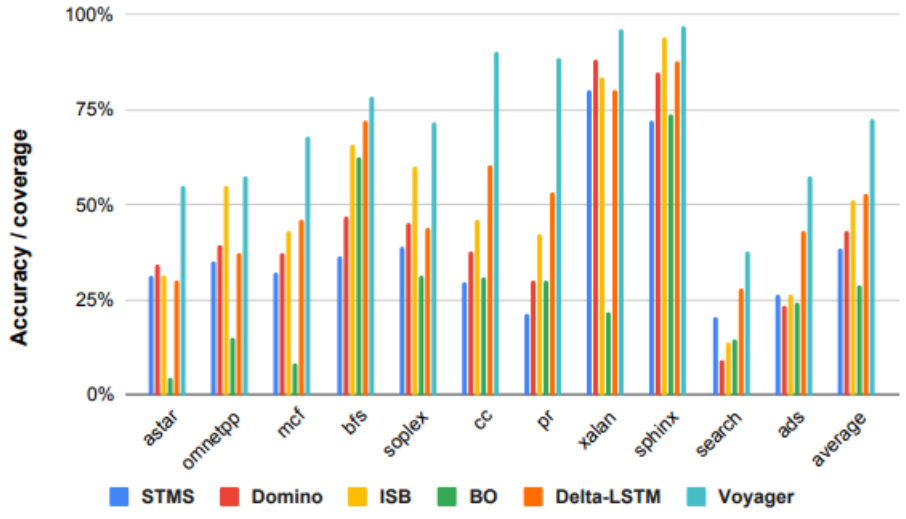

Figure 7: Unified accuracy/coverage, including Google's *search* and *ads*.

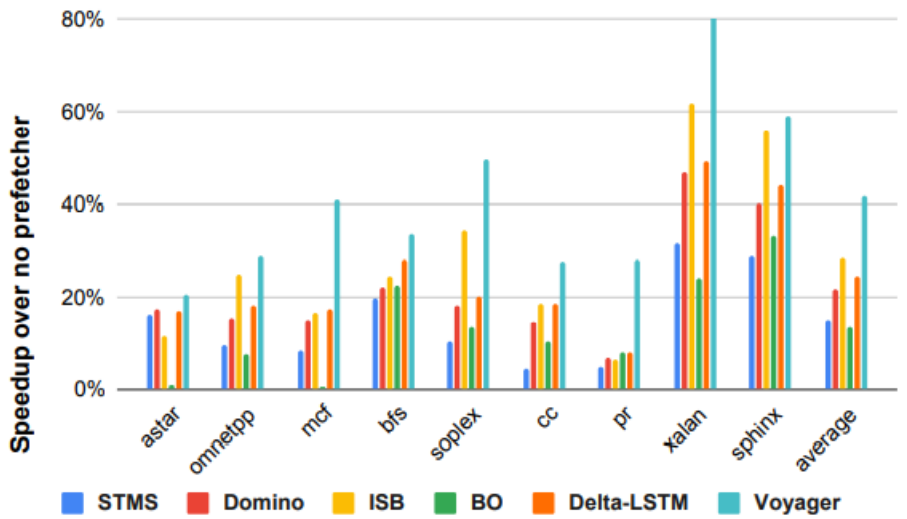

Figure 8: IPC.

Results

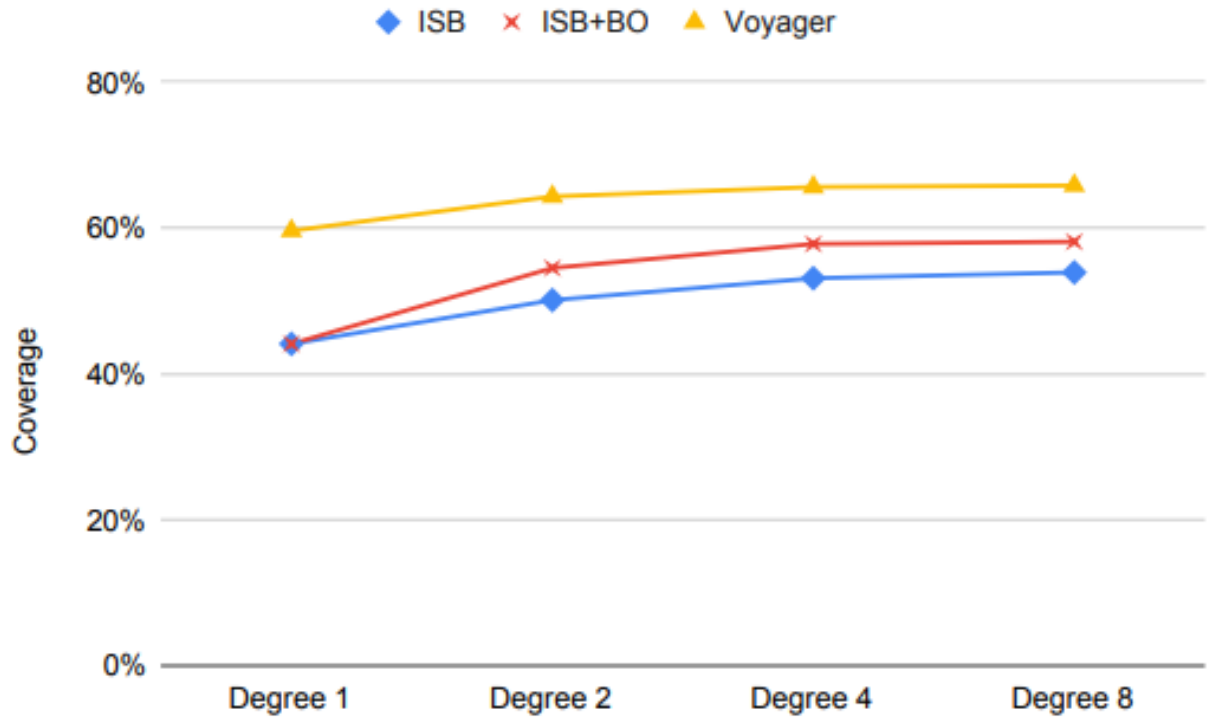

Figure 9: Sensitivity to Prefetch degree.

Results

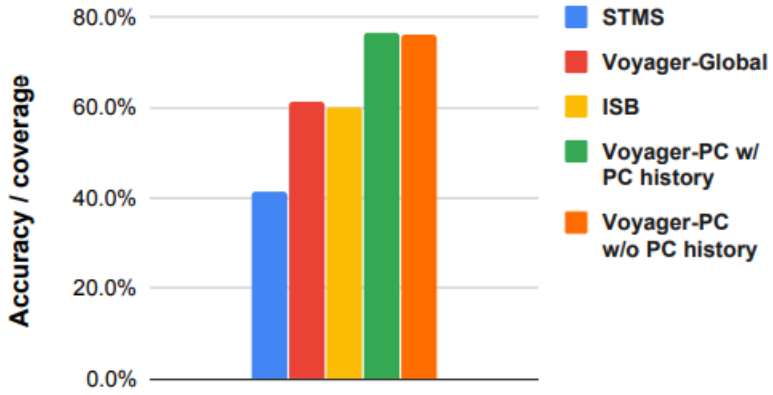

Figure 12: Comparison of different features. Voyager benefits from using data address history as a feature.

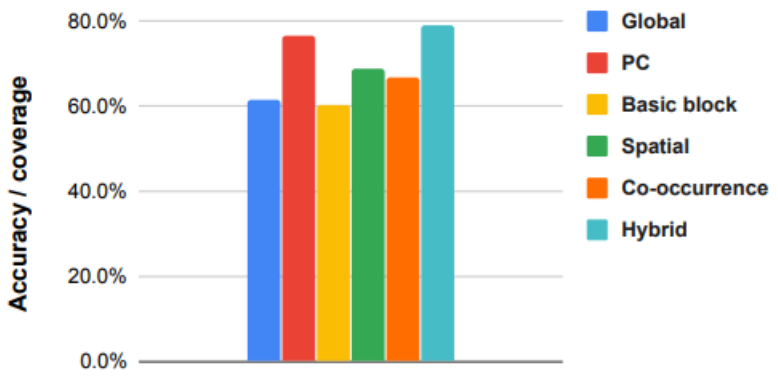

Figure 15: Comparison of different labeling schemes.

Discussion

- Is there anything that we can borrow from this paper into our own project practically?
- If Voyager is too difficult to implement, what about some previous work of prefetching method mentioned within this paper?