A Hierarchical Neural Model of Data Prefetching

Zhan Shi[1], Akanksha Jain[1], Kevin Swersky[2], Milad Hashemi[3], Parthasarathy Ranganathan[3], Calvin Lin[1]
{zshi, akanksha}@cs.utexas.edu, {kswersky, miladh, parthas}@google.com, lin@cs.utexas.edu

[1]The University of Texas at Austin [2]Google Research [3]Google Cloud

1. Motivation

Data prefetching is an important memory latency-hiding technique that has seen significant advances over decades of research. Nevertheless, considerable headroom still exists, particularly for irregular workloads. For example, on Google's *search* and *ads* workloads, we find that an idealized version of the current state-of-the-art [3] sees coverage of just 13.8% and 26.2%, respectively.

Since prefetching is a prediction problem, it is natural to wonder if powerful machine learning techniques, such as neural networks, can help. Unfortunately, there are two challenges in applying machine learning to data prefetching. First, there is a *labeling problem:* It is difficult to provide ground truth labels that can be used to train a neural model for data prefetching. Unlike a branch predictor, which can be trained based on the ground truth answer from the program's execution, a data prefetcher can potentially target any of the many addresses that the program will access in the near future. Second, there is a *class explosion* problem: the output space—i.e., the set of possible addresses to predict—is orders of magnitude larger than those used for traditional machine learning tasks in vision and natural language processing [4, 7]. For example, for a 64-bit address space, a neural data prefetcher needs to predict from among tens of millions of unique address values, which is two orders of magnitude larger than what state-of-the-art language models need to handle.

2. Limitations of the State of the Art

Previous attempts at using neural networks for data prefetching [2, 6] have three limitations.

First, to avoid the class explosion problem, they restrict the problem so that their prefetchers need only learn a few deltas. In particular, these solutions spatially partition the address space and only prefetch addresses that lie in the same partition as the trigger address. While this strategy works well for workloads with spatial locality, they are unable to capture *irregular* memory accesses that can span the entire address space.

Second, they avoid the labeling problem by always attempting to prefetch the next address in the global access stream, which can lead to poor predictability and poor timeliness.

Third, these neural prefetchers are expensive in both storage and computation. For example, Hashemi et al.'s LSTM-based prefetcher [2] consumes 100MB to several GBs of metadata, which is too large to be stored on chip.

3. Key Insights

This paper presents Voyager, a novel neural network for data prefetching. Unlike previous neural models for prefetching, which were limited to learning a relatively small number of delta correlations, our model can also learn address correlations, which are significantly more powerful.

Our solution is based on two insights.

First, we solve the class explosion problem by decomposing the learning problem into two sub-problems, namely, a page prediction and an offset prediction. While it may seem obvious to split physical addresses into pages and offsets, the challenge is to not treat the two as independent learning problems, because the page provides important context for predicting the correct offset. We solve this challenge by introducing a novel attention-based *embedding layer* that allows pages to provide context for the learning of offsets.

Second, to solve the labeling problem, our model *learns* to identify, for each trigger address, the subsequent memory address that is most predictable. We find that for most SPEC workloads, our model automatically picks the next address by the same PC, but for a few benchmarks, it identifies labels that are not consecutive in either the global access stream or the PC-localized stream.

4. Main Artifacts

This paper presents two main artifacts.

- We present Voyager, a neural model that can perform address correlation as well as delta correlation.
- We evaluate Voyager in a hardware setting by training Voyager *online* and evaluating its impact through detailed microarchitectural simulation.

5. Key Results and Contributions

We advance the state-of-the-art in irregular data prefetching by developing a neural network model that can perform both delta and temporal prefetching, and we demonstrate for the first time that LSTM-based neural prefetchers can significantly outperform existing rule-based prefetchers.

Using a set of irregular benchmarks, we show that Voyager achieves coverage of 79.6%, compared with 57.9% for ISB [3], a state-of-the-art rule-based temporal prefetcher. This coverage translates to a a 41.6% performance improvement for Voyager over a baseline with no prefetcher, compared to just 28.2% for ISB. More significantly, on Google's *search* and *ads*, two applications that have proven remarkably resistant to

hardware prefetchers, our model achieves 37.8% and 57.5% coverage, respectively, where an idealized version of ISB sees coverage of just 13.8% and 26.2%, respectively.

We also show that Voyager outperforms previous neural prefetchers [2], while significantly reducing training cost, prediction latency, and storage overhead. For example, Voyager achieves 79.6% coverage (vs. 56.8% coverage for Delta-LSTM); it reduces training and prediction costs by 15-20×; and it reduces model size by 110-200×. In fact, we find that Voyager's model size is smaller than those of state-of-the-art temporal prefetchers [8, 1, 9]. Nevertheless, neural prefetchers are currently computationally impractical for hardware deployment, so our paper outlines some potential paths from Voyager to a practical neural prefetcher.

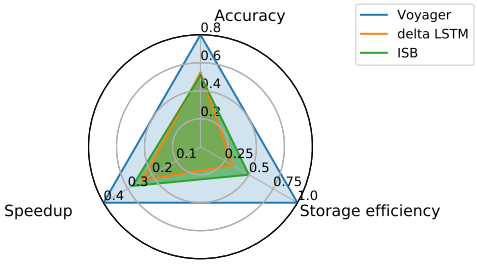

Figure 1: Voyager wins on accuracy, speedup, and storage efficiency. Here storage efficiency is log-scaled and defined as $\frac{1}{1+log_{10}(storage)}$

.

While Voyager is not practical from a computational perspective, Figure 1 illustrates the advancements that Voyager makes in three other dimensions by comparing Voyager against the previous best rule-based prefetcher (ISB [3]), and against the previous best neural prefetcher (delta-LSTM [5]).

6. Why ASPLOS

Prefetching is a common tool that is employed at many levels of the system stack to bring in various types of data. Prefetching can be performed by programmers, by compilers, by the OS, and by the hardware, and it can bring in data, instructions, TLB entries, and metadata. Because the advancement of this paper is an algorithmic one, it can potentially translate to other ares of the system stack and other uses of prefetching than data prefetching.

7. Citation for Most Influential Paper Award

This paper had a significant impact on the design of modern irregular data prefetchers and on the use of machine learning for hardware prediction. For irregular data prefetching, the paper showed the potential of employing powerful learning algorithms, which spurred future work in both neural and non-neural prefetchers. For the use of ML in systems, the paper highlighted the need to go beyond off-the-shelf neural models and to develop novel model architectures for hardware predictors.

References

[1] Mohammad Bakhshalipour, Pejman Lotfi-Kamran, and Hamid Sarbazi-Azad. Domino temporal data prefetcher. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 131–142. IEEE, 2018.

[2] Milad Hashemi, Kevin Swersky, Jamie A Smith, Grant Ayers, Heiner Litz, Jichuan Chang, Christos Kozyrakis, and Parthasarathy Ranganathan. Learning memory access patterns. *arXiv preprint arXiv:1803.02329*, 2018.

[3] Akanksha Jain and Calvin Lin. Linearizing irregular memory accesses for improved correlated prefetching. In *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 247–259. ACM, 2013.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[5] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[6] Ajitesh Srivastava, Angelos Lazaris, Benjamin Brooks, Rajgopal Kannan, and Viktor K. Prasanna. Predicting memory accesses: The road to compact ml-driven prefetcher. In *Proceedings of the International Symposium on Memory Systems*, MEMSYS '19, page 461–470, New York, NY, USA, 2019. Association for Computing Machinery.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[8] Thomas F Wenisch, Michael Ferdman, Anastasia Ailamaki, Babak Falsafi, and Andreas Moshovos. Practical off-chip meta-data for temporal memory streaming. In *2009 IEEE 15th International Symposium on High Performance Computer Architecture*, pages 79–90. IEEE, 2009.

[9] Hao Wu, Krishnendra Nathella, Dam Sunwoo, Akanksha Jain, and Calvin Lin. Efficient metadata management for irregular data prefetching. In *Proceedings of the 46th International Symposium on Computer Architecture*, pages 449–461, 2019.