An Imitation Learning Approach for Cache Replacement

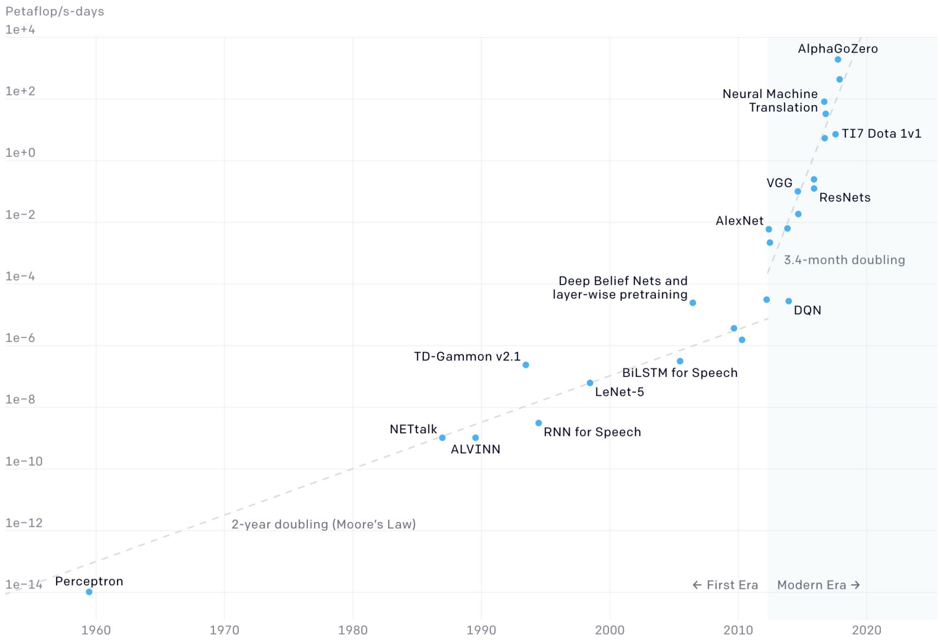
Evan Z. Liu, Milad Hashemi, Kevin Swersky,
Parthasarathy Ranganathan, Junwhan Ahn

Google Research





The Need for Faster Compute



(https://openai.com/blog/ai-and-compute/)

Small **cache** improvements can make large differences! (Beckman, 2019)

- E.g., 1% cache hit rate improvement → 35% decrease in latency (Cidon, et. al., 2016)

Caches are everywhere:

- CPU chips
- Operating Systems
- Databases
- Web applications

Our goal: Faster applications via better cache replacement policies

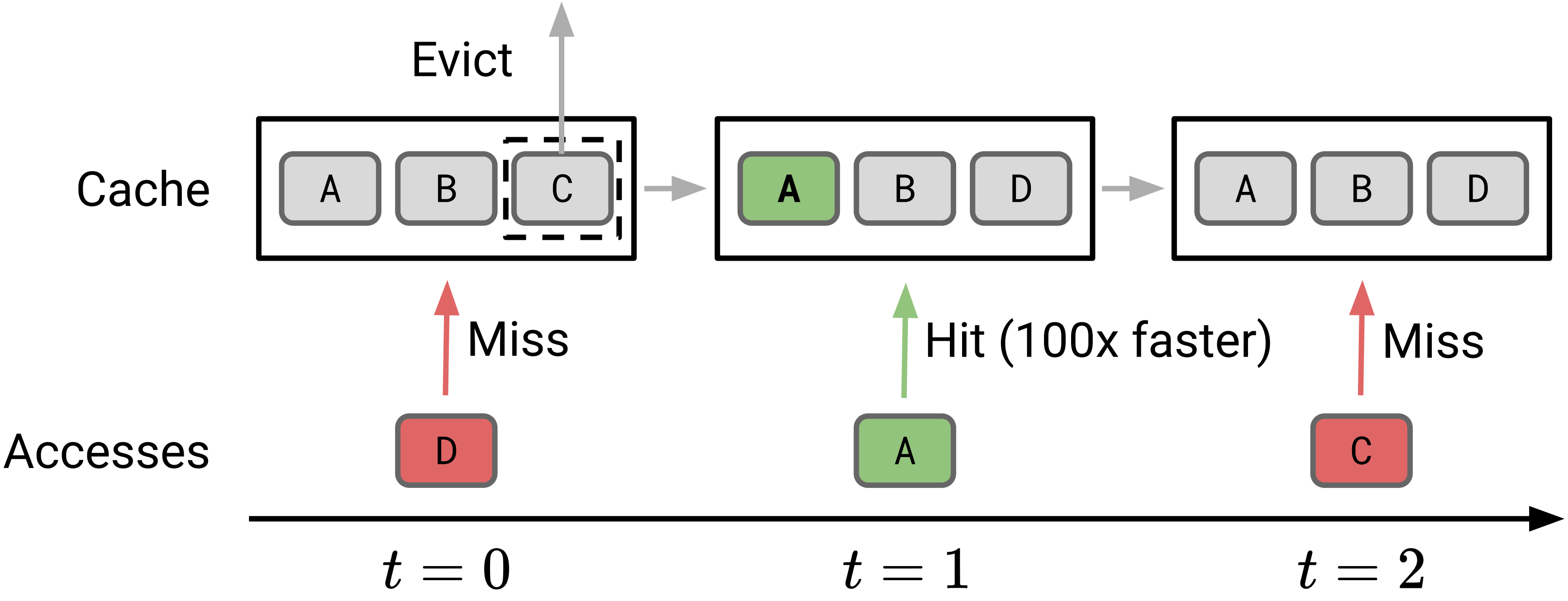
Google Research

TL;DR:

I. We approximate the **optimal** cache replacement policy by (implicitly) **predicting the future**

II. Caching is an attractive benchmark for the general **reinforcement learning / imitation learning** communities

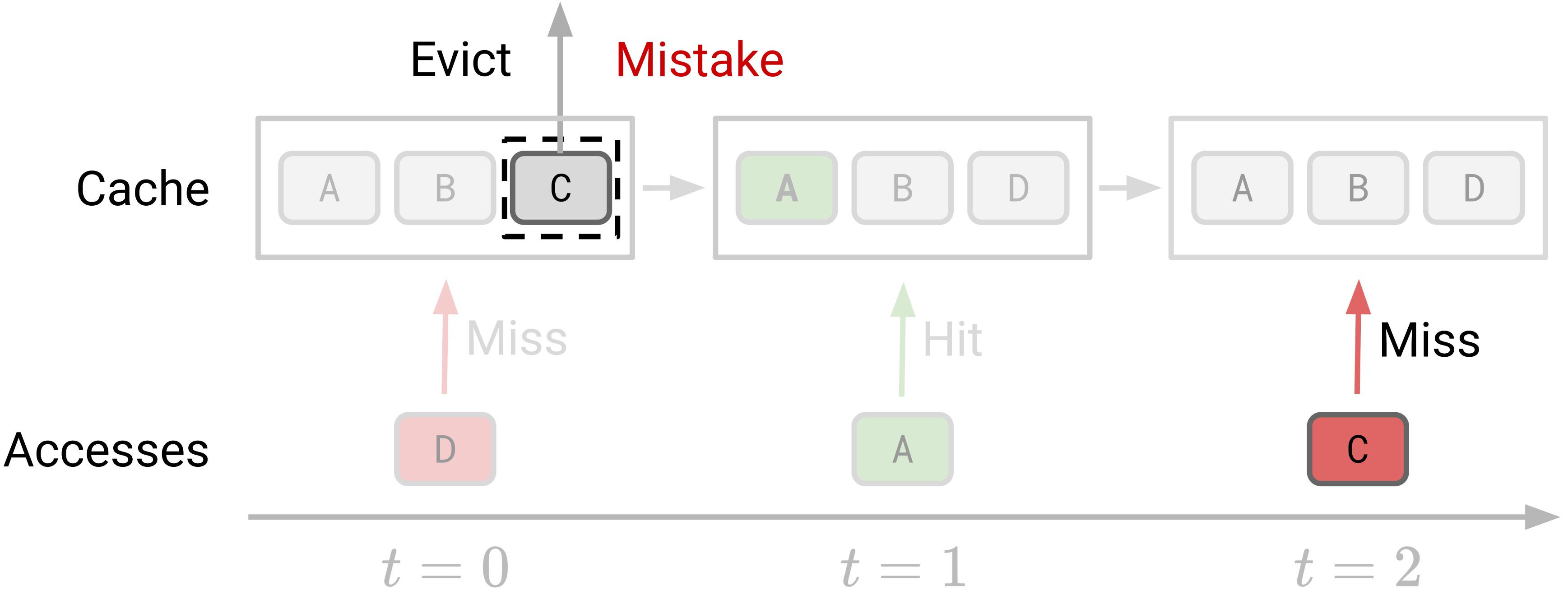
Cache Replacement



Goal: Evict the cache lines to maximize cache hits

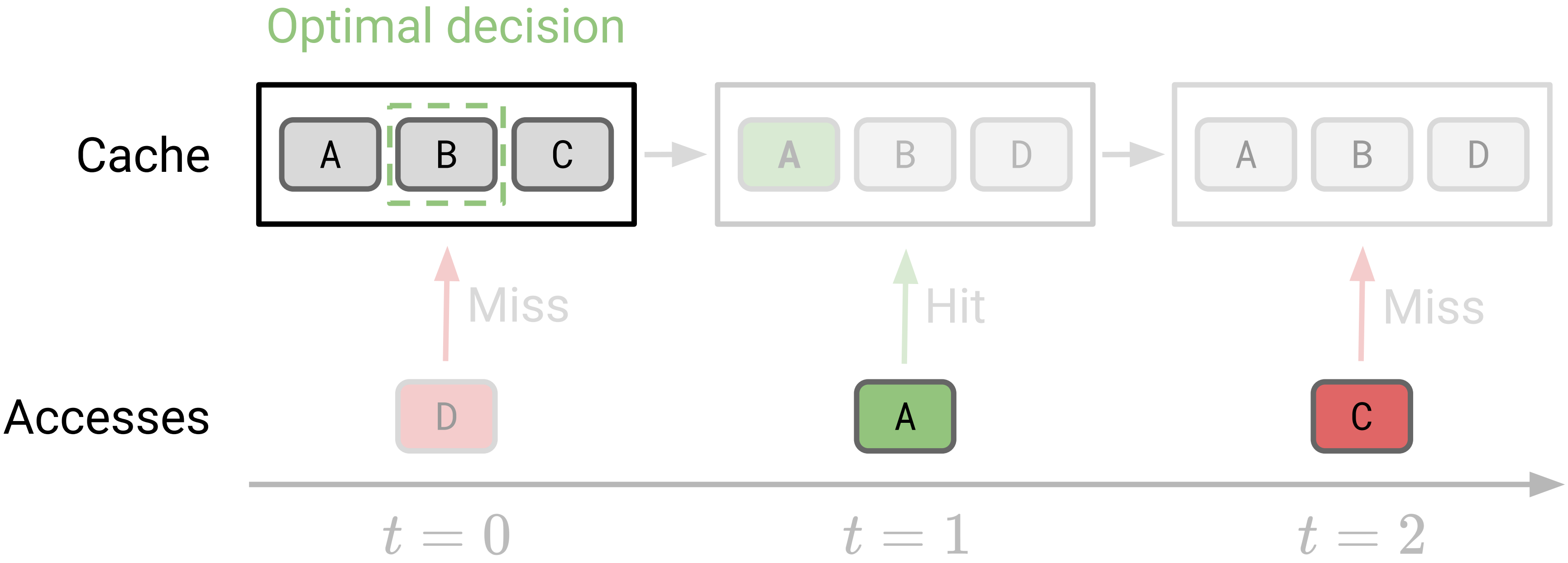
Google Research

Cache Replacement



Google Research

Cache Replacement

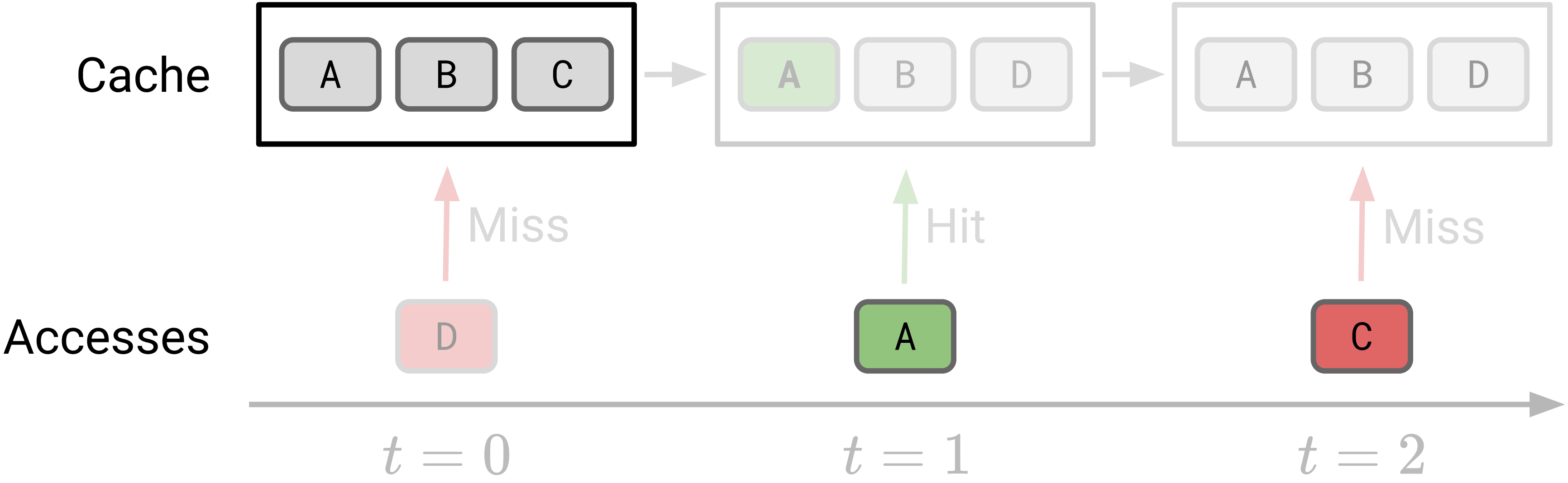


Google Research

Cache Replacement

Reuse distance d_t(line): number of accesses from access t until the line is reused

$d_0(A) = 1$, $d_0(B) > 2$, $d_0(C) = 2$



Optimal Policy (Belady's): Evict the line with the greatest reuse distance (Belady, 1966)

Google Research

Belady's Requires Future Information

Reuse distance d_t(line): number of accesses from access t until the line is reused

Problem: Computing reuse distance requires knowing the future

So in practice, we use **heuristics**, e.g.:

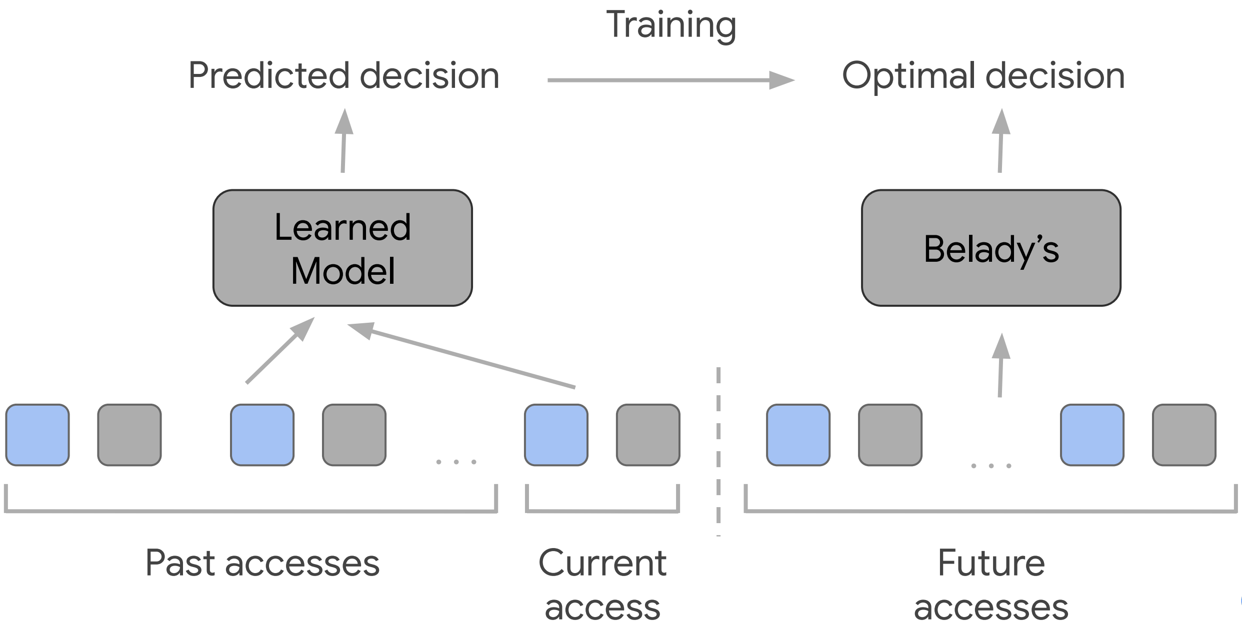
- Least-recently used (LRU)
- Most-recently used (MRU)

... but these **perform poorly** on complex access patterns

Google Research

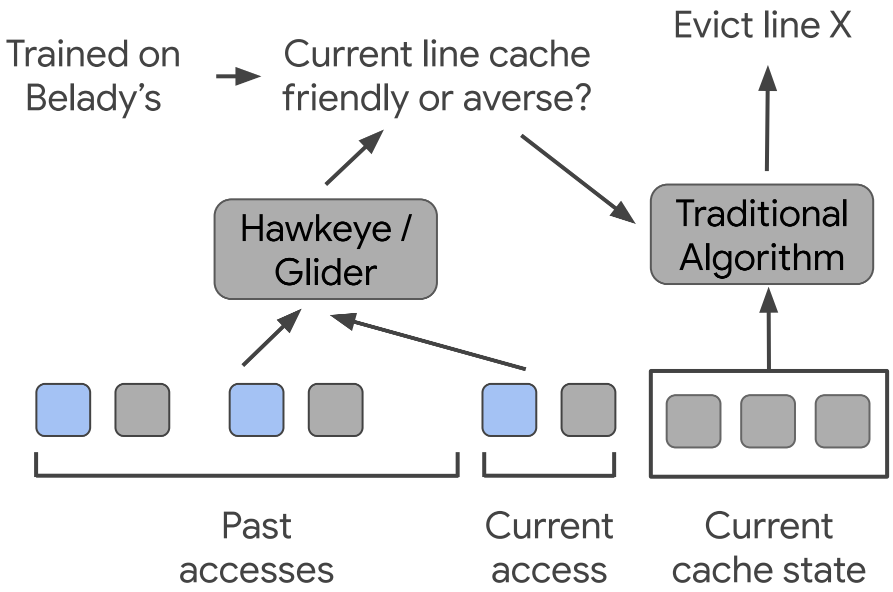
Leveraging Belady’s

Idea: approximate Belady’s from **past accesses**



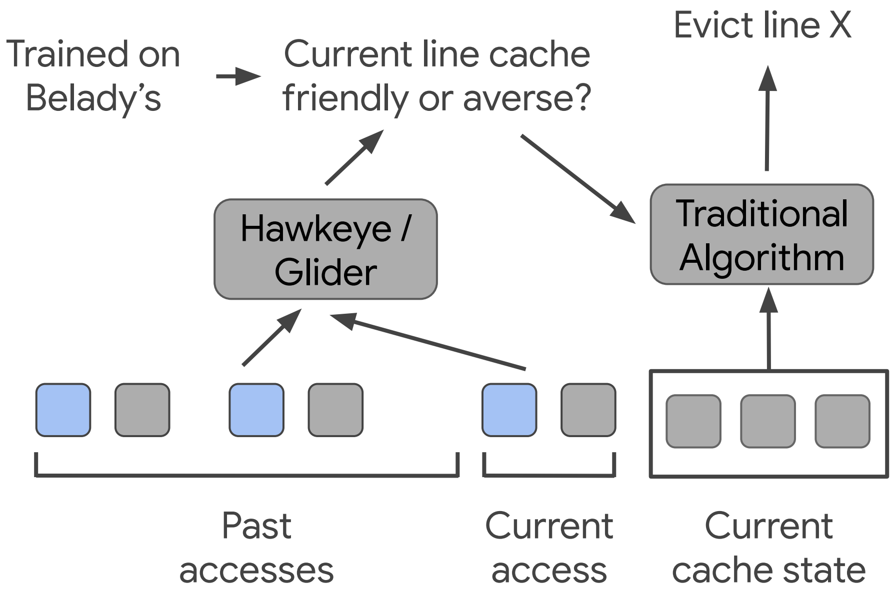
Google Research

Prior Work



Current **state-of-the-art**
(Shi et. al., '19, Jain et. al., '18)

Prior Work

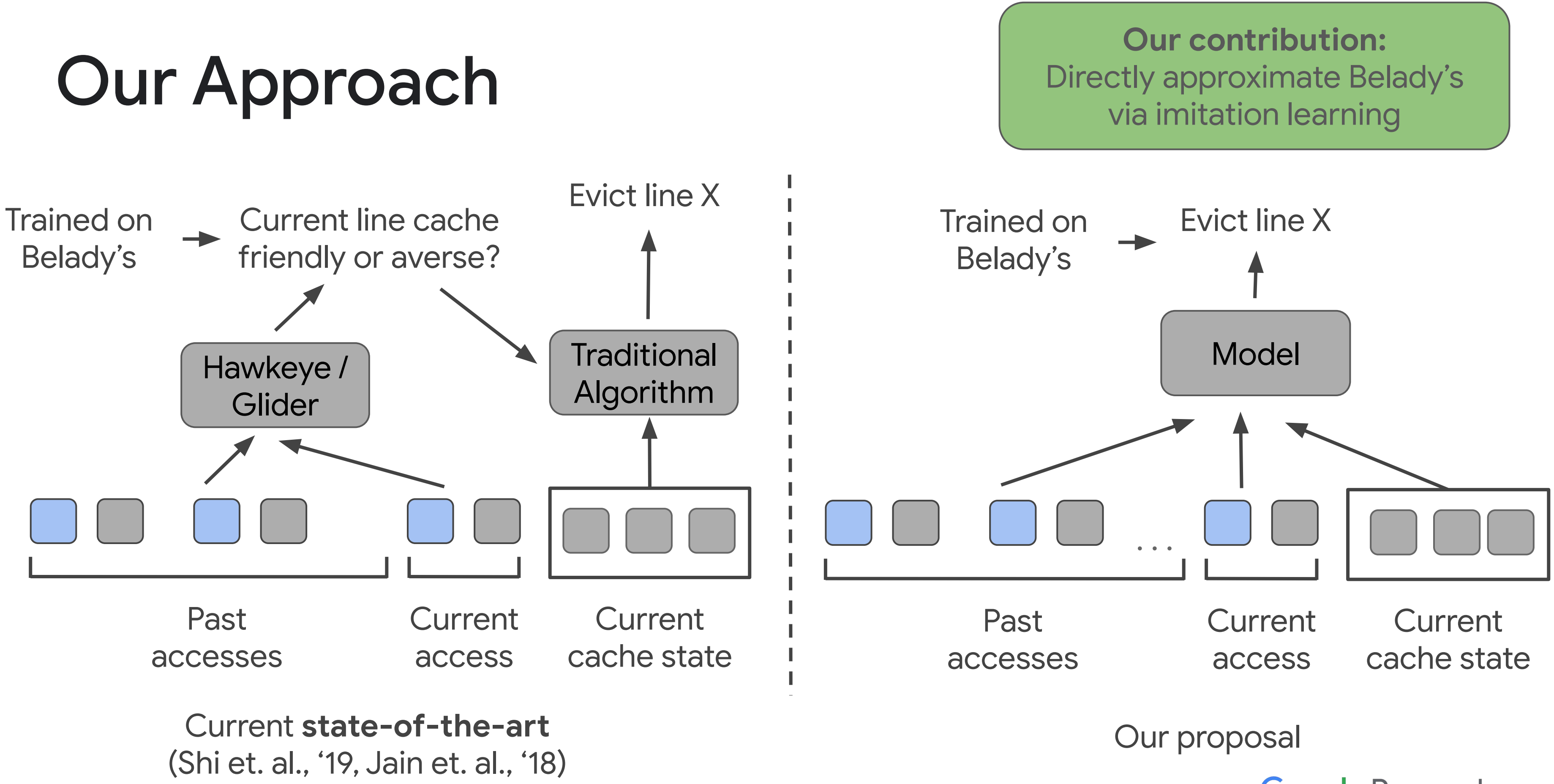


Current **state-of-the-art**
(Shi et. al., '19, Jain et. al., '18)

+ binary classification is relatively **easy to learn**

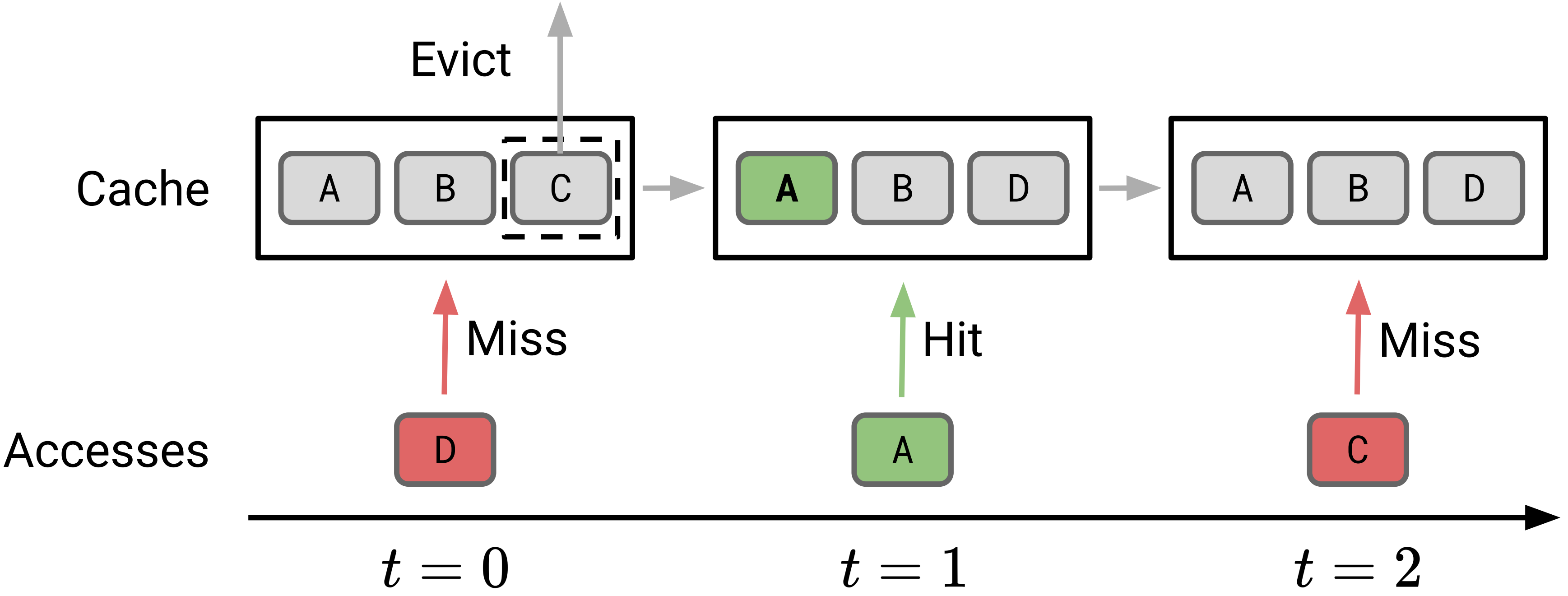
- traditional algorithm can't **express** optimal policy

Our Approach



Google Research

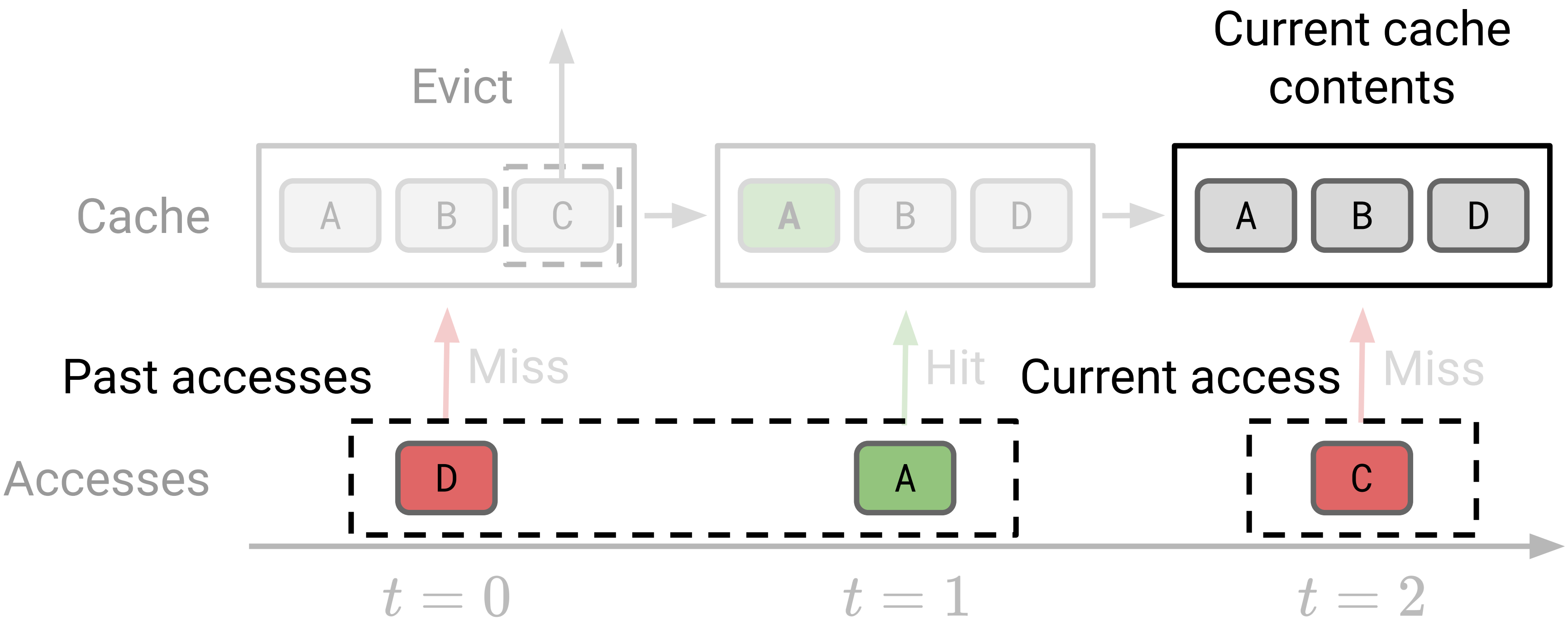
Cache Replacement Markov Decision Process



Google Research

Similar to Wang, et. al., 2019

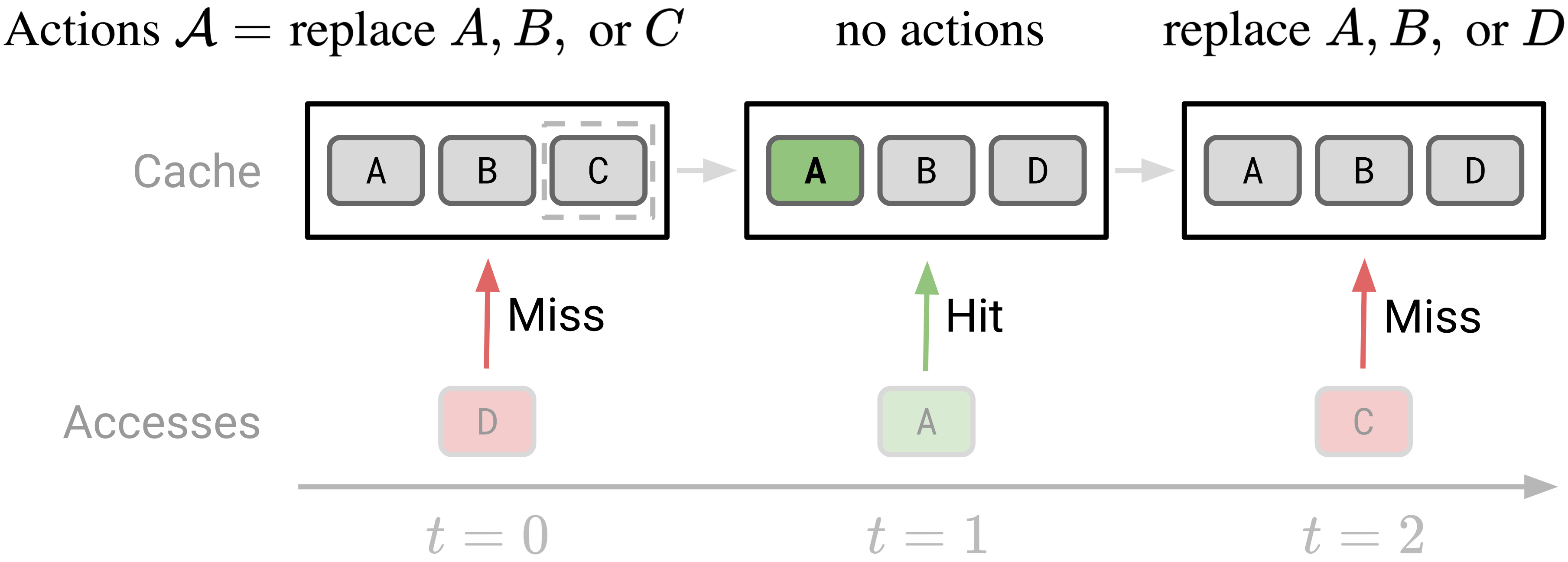
Cache Replacement Markov Decision Process



State s_2

Similar to Wang, et. al., 2019

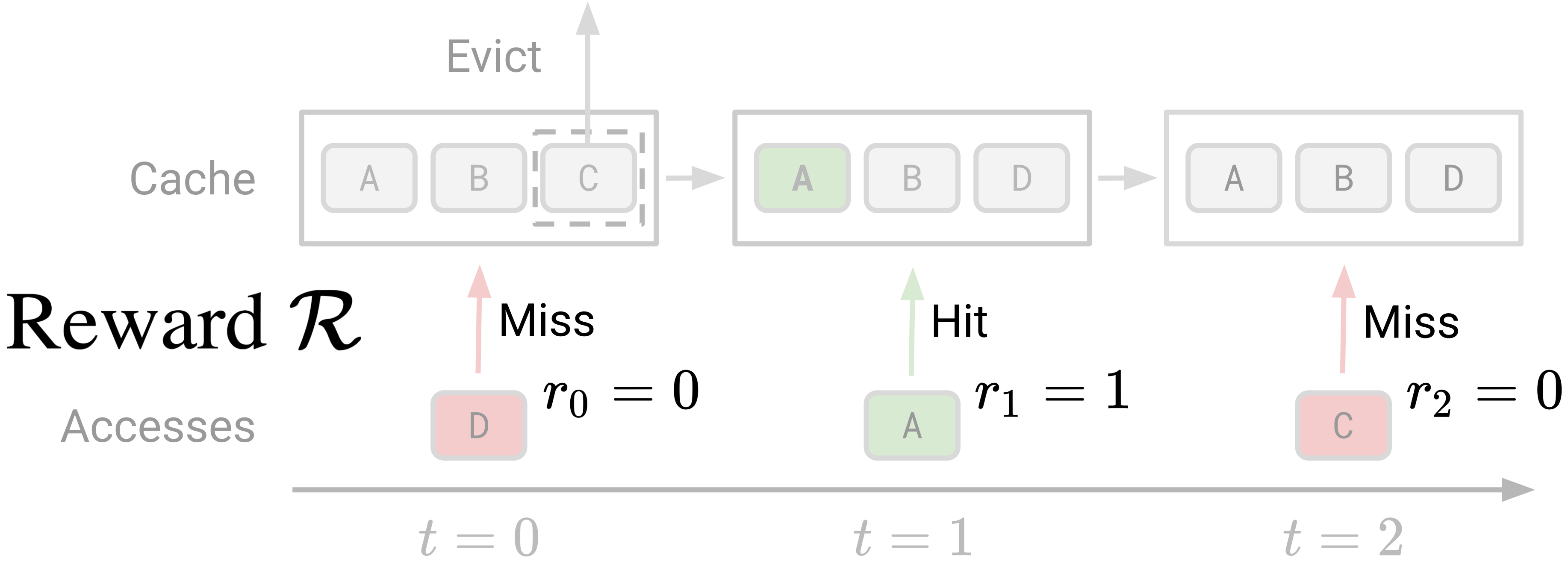
Cache Replacement Markov Decision Process



Google Research

Similar to Wang, et. al., 2019

Cache Replacement Markov Decision Process



Google Research

Similar to Wang, et. al., 2019

Leveraging the Optimal Policy

Typical imitation learning setting
(Pomerlau, 1991, Ross, et. al., 2011, Kim, et. al., 2013)

state s

optimal **action**
$a^* = \arg\max_a \pi^*(a \mid s)$

Learned policy
$\pi^\theta(a \mid s)$

$$\text{maximize}_\theta \log \pi^\theta(a^* \mid s)$$

Observation: Not all errors are equally bad

- Learning from optimal **policy** yields **greater training signal**

state s_t

Learned policy
$\pi^\theta(\text{line} \mid s_t)$

Approximate optimal **policy**
$\pi^*(\text{line} \mid s_t) \propto \exp d_t(\text{line})$

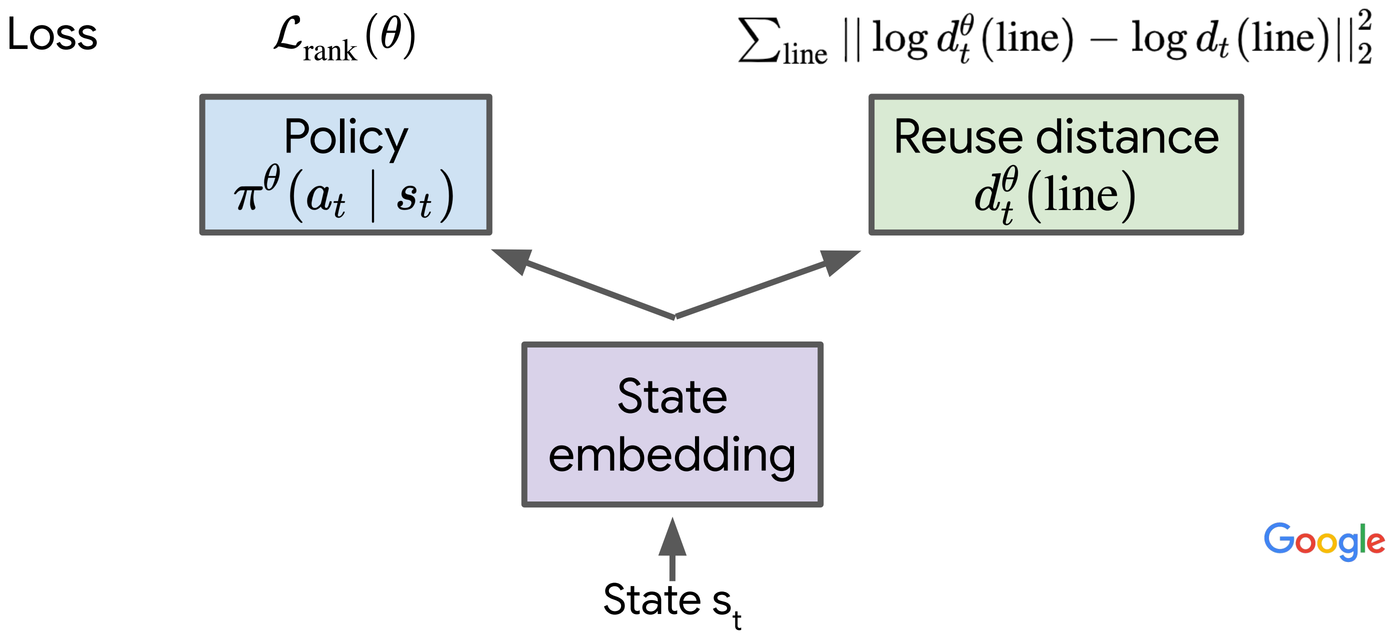
optimize, e.g., $D_{\text{KL}}(\pi^\theta(\text{line} \mid s_t) || \pi^*(\text{line} \mid s_t))$

Concretely: minimize a ranking loss $\mathcal{L}_{\text{rank}}$

Reuse Distance as an Auxiliary Task

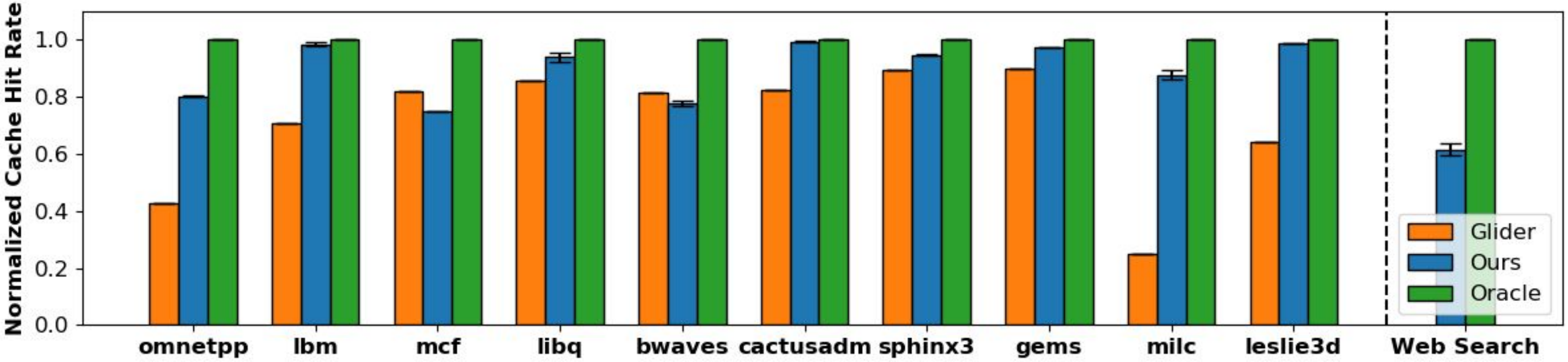
Observation: predicting reuse distance is correlated with cache replacement

- Cast this as an **auxiliary task** (Jaderberg, et. al., 2016)



Results

Optimal cache-hit rate



LRU cache-hit rate

~19% cache-hit rate increase over Glider (Shi, et. al., 2019) on memory-intensive SPEC2006 applications (Jaleel, et. al., 2009)

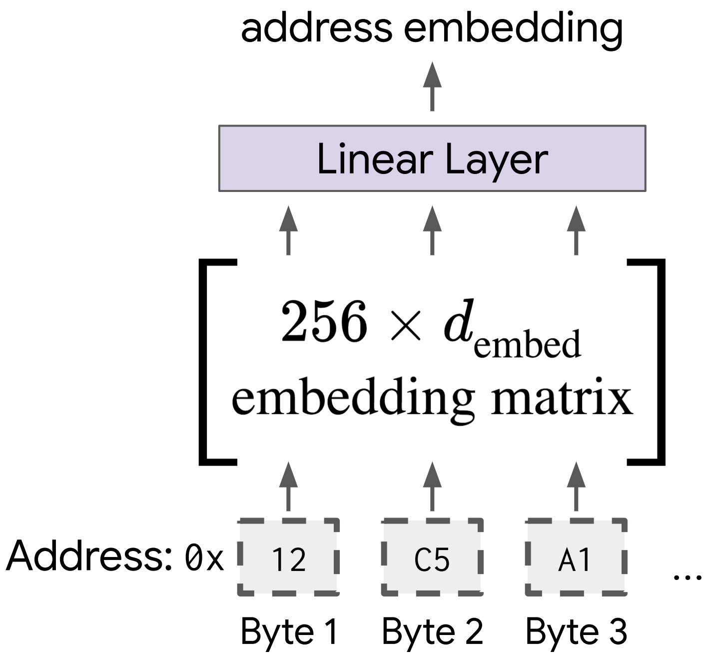
~64% cache-hit rate increase over LRU on Google Web Search

Google Research

A Note on Practicality

This work: Establish a **proof-of-concept**

Per-byte address embedding
- Reduce embedding size from **100MB** to **<10KB**
- **~6%** cache-hit rate increase on SPEC2006 vs. Glider
- **~59%** cache-hit rate increase on Google Web Search vs. LRU

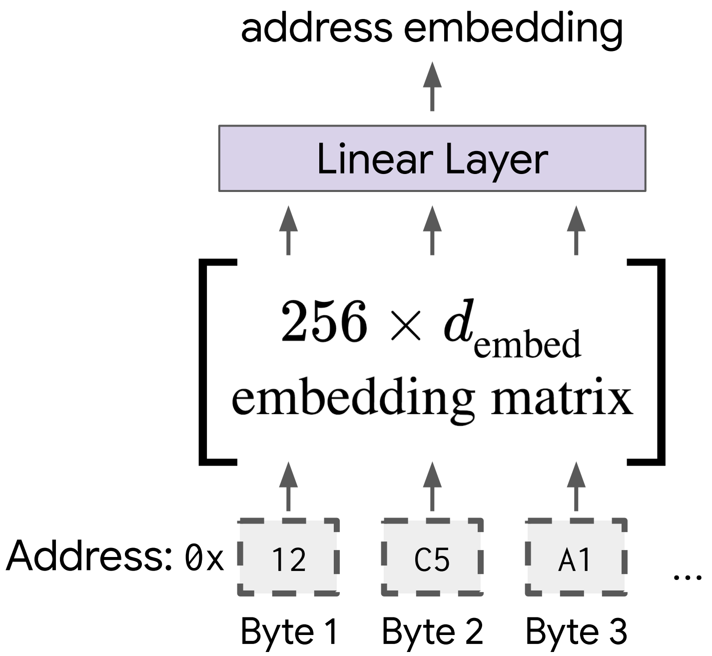


A Note on Practicality

This work: Establish a **proof-of-concept**

Per-byte address embedding

- Reduce embedding size from **100MB** to **<10KB**
- **~6%** cache-hit rate increase on SPEC2006 vs. Glider
- **~59%** cache-hit rate increase on Google Web Search vs. LRU



Future work: Production ready learned policies

- **Smaller models** via distillation (Hinton, et. al., 2015), pruning (Janowsky, 1989, Han, et. al., 2015, Sze, et. al., 2017), or quantization
- Target domains with **longer latency** and **larger caches** (e.g., software caches)

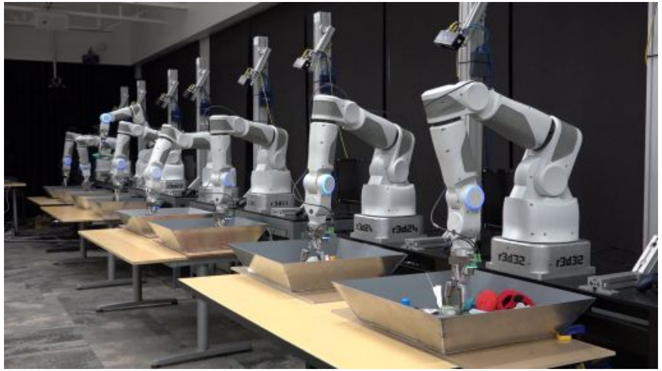
Google Research

A New Imitation / Reinforcement Learning Benchmark

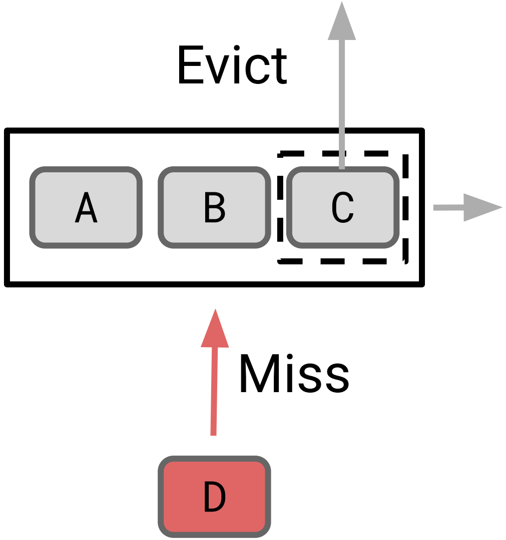
Bellemare, et. al., 2012,
Silver, et. al., 2017, OpenAI, 2019,
Vinyals, et. al., 2019

+ plentiful data
- delayed real-world utility

Levine, et. al., 2016, Lillicrap, et. al., 2015



- limited / expensive data
+ immediate real-world impact



+ plentiful data
+ immediate real-world impact

Google Research

Open-source cache replacement Gym environment coming soon!

Takeaways

- A new **state-of-the-art** approach for cache replacement by **imitating** the oracle policy
    - Future work: making this **production ready**

- A new **benchmark** for imitation learning / reinforcement learning research

Google Research