Optimization of Hardware Cache Management Using AI

FUNDAMENTALS OF COMPUTER ENGINEERING

Course Instructor-
Prof. Naveen Naik Sapavath
Assistant Teaching Professor

Ajayanath Chittela
Keerthana Mohana
Suraj Patel Muthe Gowda

OVERVIEW

- Introduction
- Data source
- Algorithms
- Metrics
- Tools

INTRODUCTION

- **Cache Optimization**- Cache optimization improves how data is stored and retrieved from cache memory to reduce latency and increase system performance.
- **Significance** - In high-performance computing systems, efficient cache management can reduce memory access time, minimize processor stalls, and improve overall performance.
- **Using AI** - AI can dynamically adapt cache policies by learning from past memory access patterns, making better predictions for future access and improving cache efficiency.

PROBLEM STATEMENT

Current challenges:

- Cache misses
- Static Policies
- Performance Bottlenecks

PROBLEM STATEMENT

The goal of AI-based Cache Optimization:

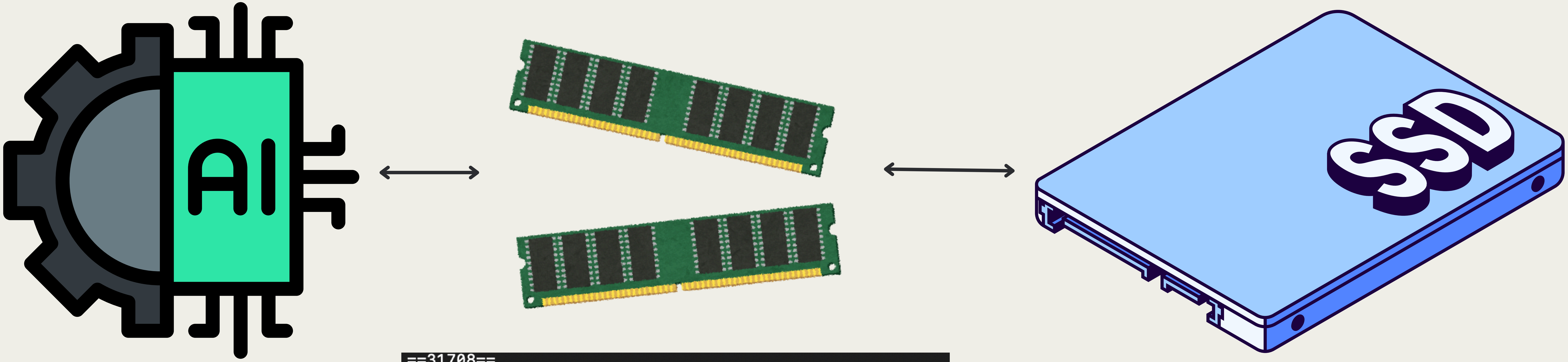
- Use AI/ML techniques to create adaptive cache management strategies that can reduce cache misses and optimize overall performance.

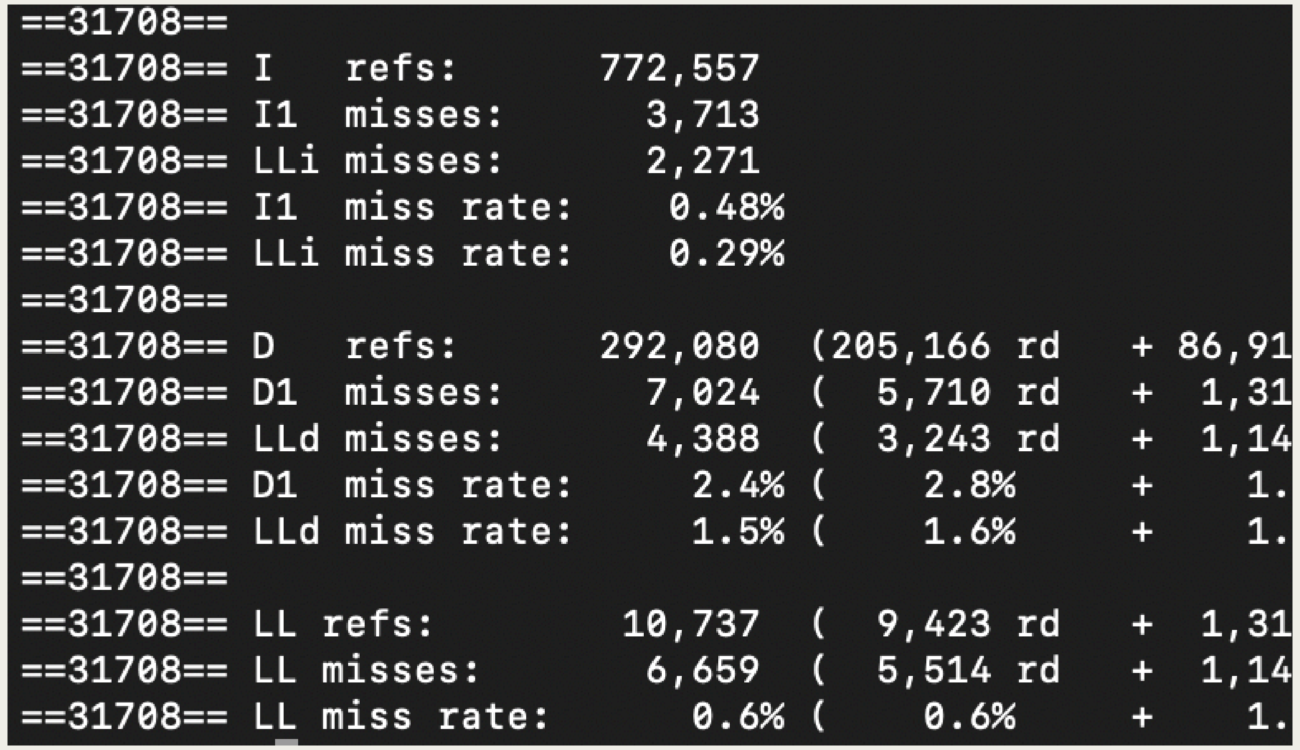
DATA SOURCE

Data Used for Training AI Models:

- **Cache Access Patterns**: Logs that show which memory blocks are being accessed by the CPU and how frequently.
- **Memory Traces**: Collected from hardware simulators or performance counters, providing detailed information about every memory access (e.g., read/write operations).
- **Hardware Performance Counters**: Real-time data collected from processors (e.g., Intel’s or ARM’s built-in performance counters) that capture cache hits/misses, branch predictions, etc.

DATA SOURCE- METHODOLOGY

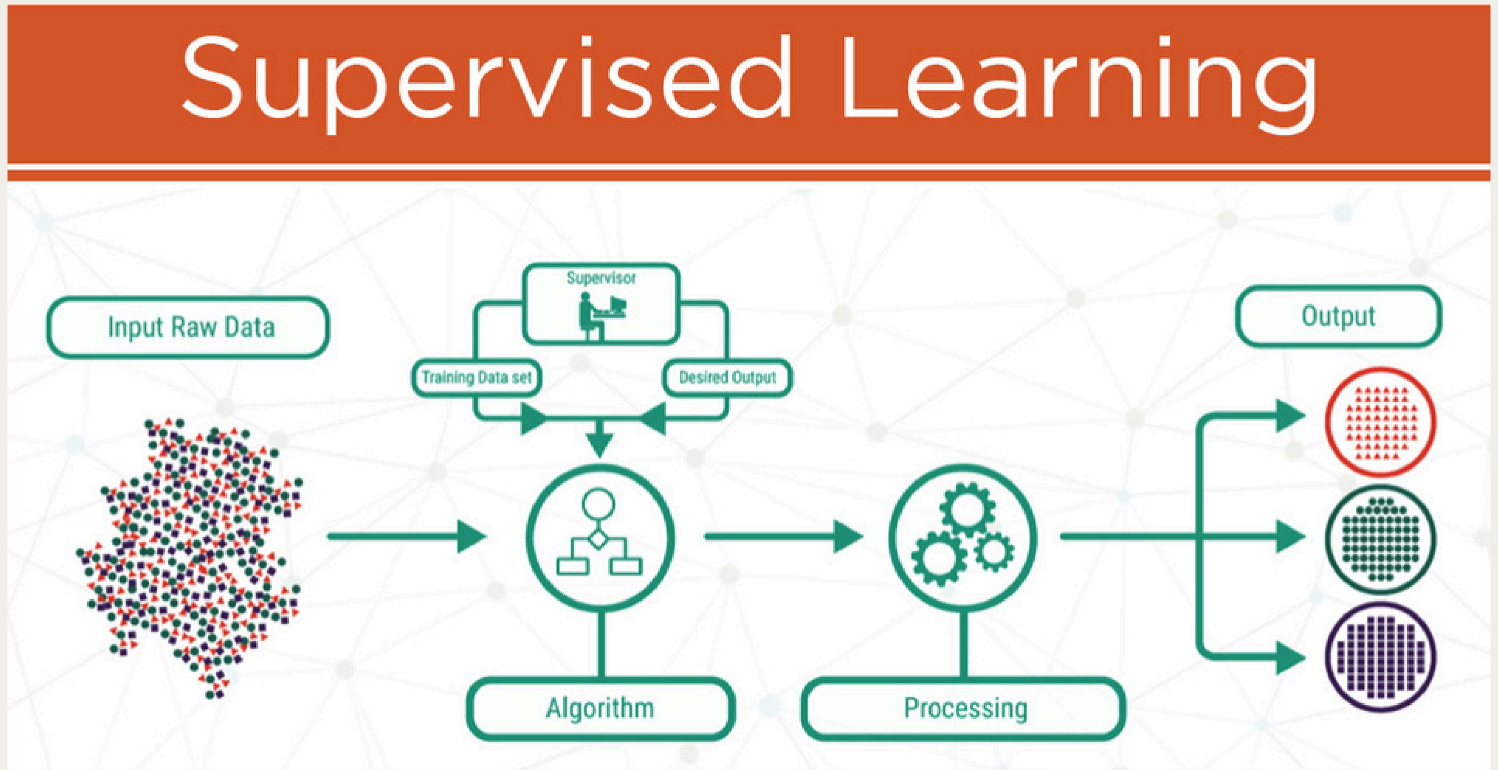


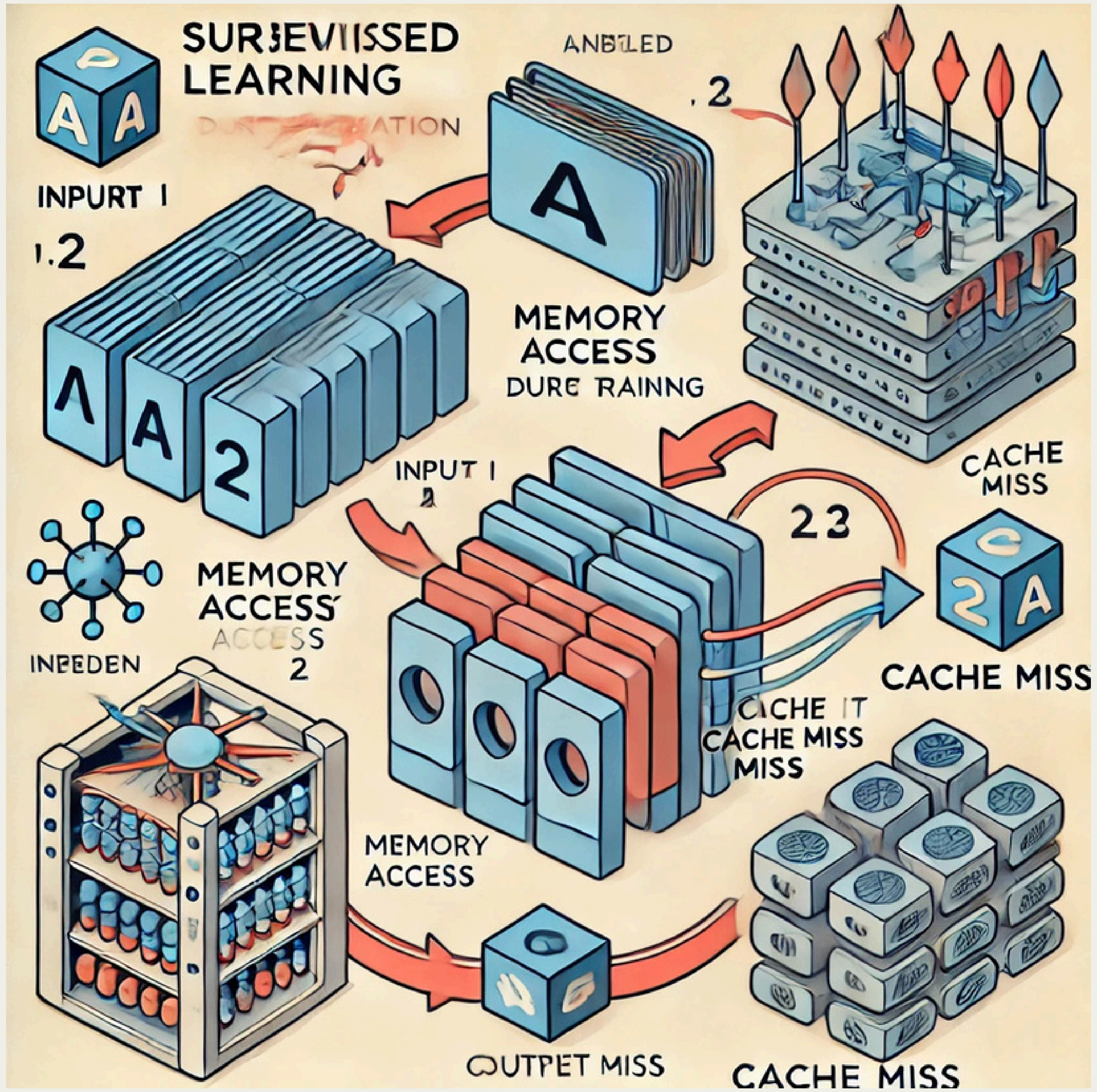


```
==31708==
==31708== I   refs:      772,557
==31708== I1  misses:      3,713
==31708== LLi misses:      2,271
==31708== I1  miss rate:    0.48%
==31708== LLi miss rate:    0.29%
==31708==
==31708== D   refs:      292,080  (205,166 rd   + 86,91
==31708== D1  misses:      7,024  (  5,710 rd   + 1,31
==31708== LLd misses:      4,388  (  3,243 rd   + 1,14
==31708== D1  miss rate:     2.4% (    2.8%     +    1.
==31708== LLd miss rate:     1.5% (    1.6%     +    1.
==31708==
==31708== LL refs:        10,737  (  9,423 rd   + 1,31
==31708== LL misses:       6,659  (  5,514 rd   + 1,14
==31708== LL miss rate:      0.6% (    0.6%     +    1.
```

ALGORITHMS

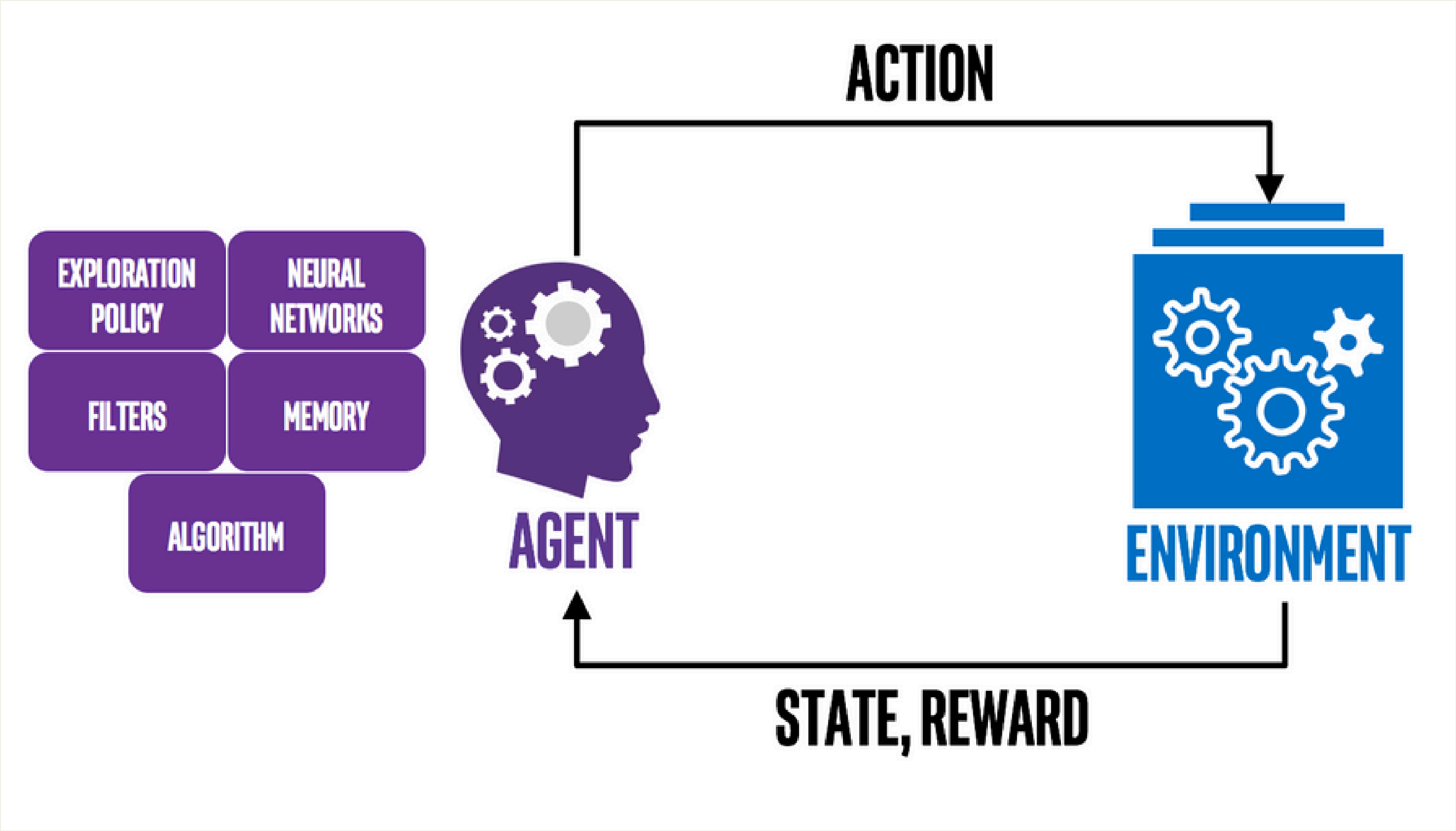
Supervised Learning - A type of machine learning technique where models are trained on labeled data, meaning the correct output (cache hit/miss) is already known for each input (memory access pattern).

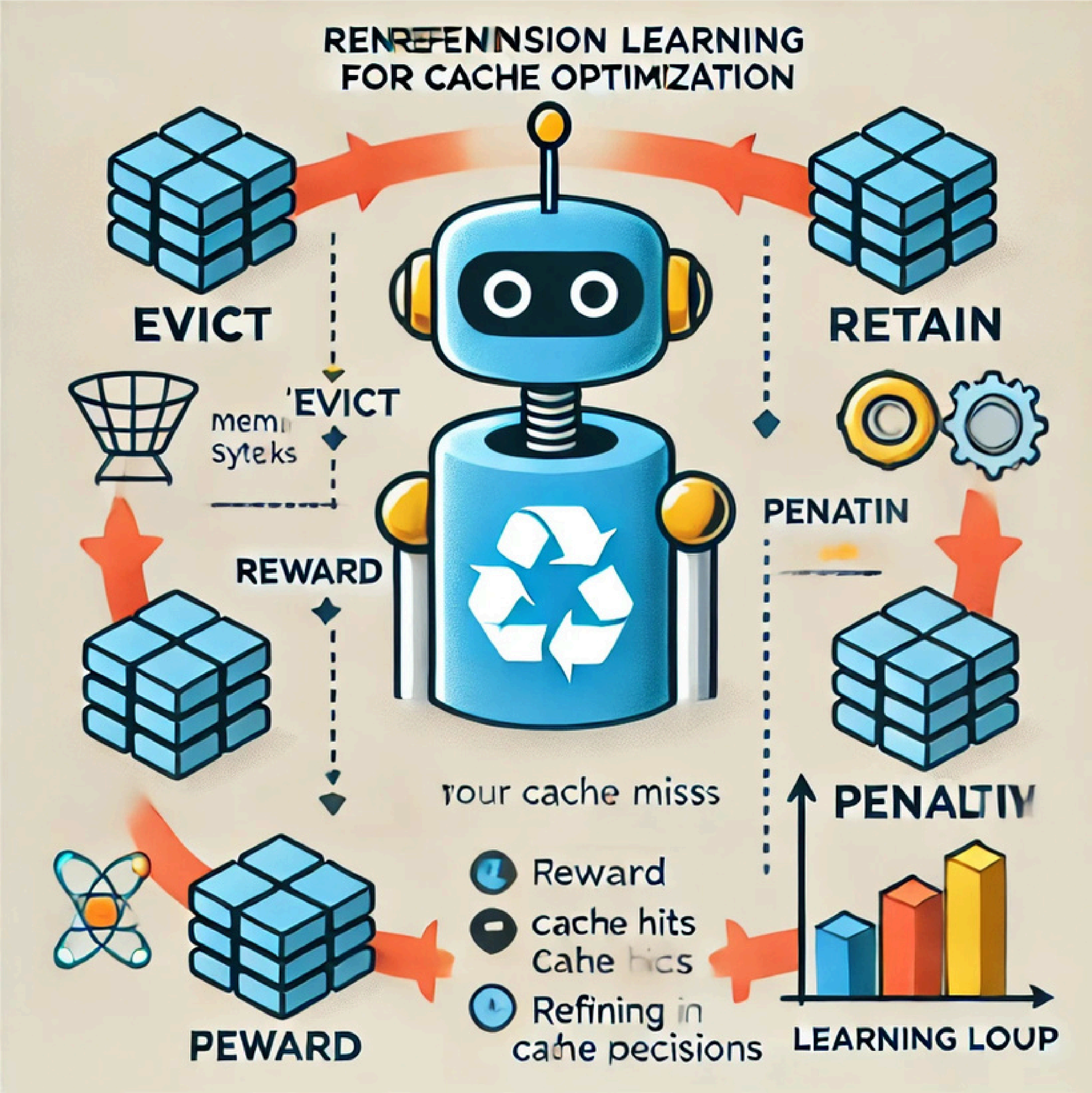




ALGORITHMS

Reinforcement Learning (RL) - A type of machine learning where an agent learns to make decisions through trial and error, receiving feedback (rewards/penalties) based on the outcomes of its actions.





METRICS USED TO EVALUATE PERFORMANCE

- Key Metrics to Evaluate Cache Performance:
  - **Cache Hit Rate:** The percentage of memory accesses found in the cache. A higher hit rate indicates better cache performance.
  - **Cache Miss Rate:** The percentage of memory accesses that are not found in the cache, leading to slower main memory access.
  - **Memory Access Latency:** The time taken to access data from memory, which is minimized with an optimized cache system.
  - **Energy Consumption:** The energy consumed by the cache subsystem. AI-based approaches can reduce energy usage by minimizing unnecessary cache lookups.

METRICS USED TO EVALUATE PERFORMANCE

- Benchmarking Approach:
  - Compare AI-optimized cache policies with traditional ones (e.g., LRU, Random) across these metrics to assess performance improvements.

TOOLS AND TECHNOLOGIES

AI/ML Tools:

- TensorFlow/PyTorch
- Google colab
- Open CV
- Hugging Face

Cache Simulation and Profiling Tools:

- Valgrind/ Cachegrind
- Perf

Hardware Used:

- Intel CPUs- X86
- ARM

Thank you!