Overview of the ML Prefetcher

1 Problem Setup

At each cache access t (at L2C), we observe a feature vector $x_t \in \mathbb{R}^d$ and choose one or more actions $a \in \mathcal{A}$ from a finite stride set

$$\mathcal{A} = \{+1, +2, +4, +8, -1, -2, -4, -8\}.$$

Each issued prefetch is rewarded later with a binary label

$$y = \begin{cases} 1 & \text{if a demand hits the prefetched line at L2C (useful)} \\ 0 & \text{if it times out (useless).} \end{cases}$$

Timeout is enforced after a fixed AGE = TIMEOUT (measured in accesses).

2 Model (Logistic Classifier Per Action)

For each action $a \in \mathcal{A}$, maintain a weight vector $w_a \in \mathbb{R}^d$. The score is computed with logistic regression:

$$p(a \mid x) = \sigma(w_a^\top x) = \frac{1}{1 + e^{-w_a^\top x}}.$$

3 Feature Vector

A compact $d = 16$ feature vector is built on each access:

- Bias term (1)
- Hashed PC bucket (4 one-hot)
- Hashed page-offset bucket (4 one-hot)
- Last access hit/miss flag (1)
- Access type one-hot: load/store/prefetch (3)
- Tiny PC-stride memory: $\{-1, 0, +1\}$ one-hot (3)

4 Action Selection Policy

1. Compute $p(a \mid x_t)$ for all $a \in \mathcal{A}$.
2. Rank actions by probability.
3. Issue those with $p(a) \geq$ `thr` up to `max_out`.
4. ε-greedy exploration: with probability ε_t, replace the top candidate with a plausible alternative (e.g., -1 vs. $+1$).

5. Lookahead (optional): if $p(a) \geq \texttt{thr} + \Delta$, prefetch one additional hop subject to $\texttt{max\_out}$.

Exploration schedule:

$$\varepsilon_t = \varepsilon_{\text{start}} + (\varepsilon_{\text{end}} - \varepsilon_{\text{start}}) \cdot \min\left(1, \frac{t}{T_{\text{decay}}}\right).$$

5 Credit Assignment (Delayed Labels)

Each issued prefetch is stored as

$$\text{pending}[pf\_line] \leftarrow (a, x_t, \text{age} = 0).$$

On later accesses:

- If a demand hits the prefetched line: update with $y = 1$ and remove.
- If age $\geq$ TIMEOUT: update with $y = 0$ and remove.

6 Online Learning with L2 Regularization

For the rewarded action a, the prediction is

$$\hat{y} = \sigma(w_a^\top x).$$

Gradient:

$$\nabla_{w_a} \ell = (\hat{y} - y)x.$$

SGD update:

$$w_a \leftarrow (1 - \eta\lambda)w_a + \eta(y - \hat{y})x,$$

with learning rate η and L2 regularization λ.

Decay $(1 - \eta\lambda)$ shrinks weights each update, preventing overfitting and ensuring adaptation to new phases.

7 Windowed Aggressiveness Controller

Every W L2C accesses, compute:

$$\alpha = \frac{\text{useful}}{\max(1, \text{issued})}, \quad \kappa = \frac{\text{useful}}{\max(1, \text{demand misses})}.$$

Rules:

- If $\kappa < 0.10$ and $\alpha \geq 0.80$: lower threshold, allow more prefetches:

$$\texttt{thr} \leftarrow \max(0.40, \texttt{thr} - 0.05), \quad \texttt{max\_out} \leftarrow \min(3, \texttt{max\_out} + 1).$$

- If $\alpha < 0.75$: pull back:

$$\texttt{thr} \leftarrow \min(0.65, \texttt{thr} + 0.05), \quad \texttt{max\_out} \leftarrow 1.$$

8 Metrics

- Accuracy:

$$\frac{\text{useful}}{\max(1, \text{issued})}.$$

- Coverage (true, relative to baseline no-prefetch run):

$$\frac{\text{useful}}{\max(1, \text{baseline demand misses})}.$$

- IPC:

$$\frac{\text{instructions}}{\text{cycles}}.$$

9 Full End-to-End Workflow

1. Build features x_t at each access.
2. Compute $p(a \mid x_t)$, apply exploration, select actions with threshold+cap.
3. Store pending prefetch entries with context (a, x_t).
4. On demand/timeout, generate labels $y \in \{0, 1\}$ and update weights.
5. Every W accesses, recompute accuracy/coverage and adjust knobs.
6. Output metrics IPC, accuracy, coverage.

10 Hyperparameters

- Learning rate $\eta \approx 10^{-2} - 10^{-3}$
- L2 coefficient $\lambda \approx 10^{-4} - 10^{-3}$
- TIMEOUT = 256–1024 accesses
- Window $W = 2048$–4096
- ε schedule parameters $(\varepsilon_{\text{start}}, \varepsilon_{\text{end}}, T_{\text{decay}})$
- Threshold bounds $[0.40, 0.65]$, step size 0.05
- `max_out` cap = 3 or 4

11 Why It Works

- Logistic regression provides calibrated probabilities of usefulness.
- Threshold balances benefit vs. cost.
- Timeout enforces timeliness and yields negative labels.
- Weight decay avoids blowup and adapts to phase changes.
- ε-greedy ensures exploration.
- Windowed controller balances accuracy and coverage.