Hold That Exit:
Near Optimal Early-Exit Inference via Recall

Yuanyuan Yang **Ruimin Zhang** **Jamie Morgenstern** **Haifeng Xu**

Abstract

Early-exit (EE) models improve the efficiency of deep neural networks by attaching auxiliary classifiers to intermediate layers, enabling predictions before the final layer and reducing inference latency and cost. A central challenge, however, is the principled design of *provably efficient* exit rules—a dimension often underexplored in practice, where simple confidence-thresholding dominates.

We provide theoretical guidance for designing such rules. We prove that exit strategies without recall—including standard thresholding—fail to achieve any constant-factor approximation of the optimal accuracy–latency trade-off. To address this, we formalize and analyze *with-recall* strategies, which permit revisiting earlier exits to balance accuracy and efficiency. Our results show that recall is indispensable for provable performance guarantees.

Empirical evaluations on computer vision tasks further elucidate the structure of optimal exit rules. In these settings, the optimal strategy reduces to adaptive thresholding with recall, offering a theoretical foundation for the practical deployment of early-exit models.

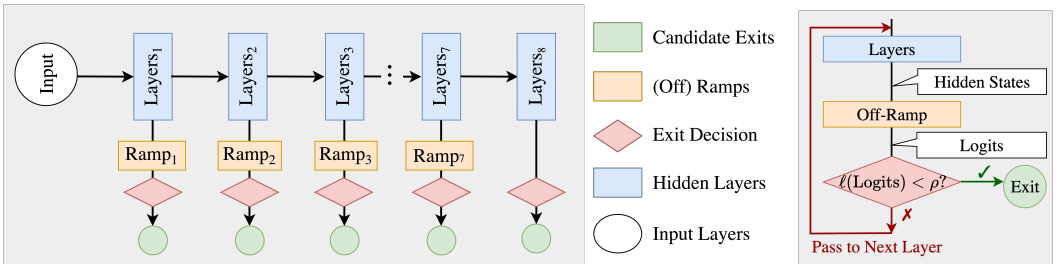
1 Introduction

Early exit (EE) models are essential for latency-sensitive inference tasks [RPSC+24, ZCL+19], by enabling input-adaptive early termination within deep neural networks. The core intuition underlying EE models is that the input of a neural network exhibits varying levels of difficulty, leading to *heterogeneous* computational requirements across these inputs. For low-complexity inputs, the feature representations extracted by shallow layers (i.e., early stages of the forward pass) may already be informative enough for confident prediction. By inserting auxiliary classifiers—also known as off-ramps—at intermediate layers, the model can exit early when the prediction loss falls below a tunable *threshold*, thereby bypassing deeper computations and substantially reducing inference latency. With their ability to balance accuracy and efficiency, early exit models have attracted growing research interest and proven effectiveness across both vision [TMK16] and language tasks [XTL+20].

Substantial advances in the development of EE classifiers have not been matched by provably efficient approaches to thresholding, which remains largely heuristic. In most prior work, confidence thresholds at off-ramps are selected through greedy search or computationally expensive grid search procedures [RPSC+24, DPI+24]. These methods, however, fail to resolve the fundamental issue of *interdependency*: thresholds in EE models are inherently coupled, as the decision at one exit directly influences the distribution of inputs to subsequent exits. As a result, the performance of any single threshold depends on the configuration of the others, making isolated or sequential tuning not only unreliable but also devoid of theoretical performance guarantees.

While loss thresholding is a natural candidate for exit strategies with rigorous theoretical guarantees, we show that its performance can be fundamentally *limited*. Such strategies operate in a *no-recall* setting, where one must irrevocably decide whether to exit at the current opportunity or continue

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Structured Probabilistic Inference & Generative Modeling.



(a) Pipeline for Early Exit Models (b) Exit by Thresholding

Figure 1: Exit Decisions in Early-Exit Models. (a) illustrates the pipeline of an early-exit model; (b) depicts the exit decision process under a thresholding strategy. During inference, the decision rule (◊) is the only component that can be modified.

exploring future ones. We theoretically model the no-recall exit problem and prove that no strategy—including thresholding—can guarantee a bounded approximation to the offline optimal loss (§3). These findings highlight the theoretical limitations of no-recall exit decisions and motivate the pursuit of alternative approaches and more suitable benchmarks.

In this work, we propose a novel theoretical framework for characterizing and computing optimal exit policies in early-exit models (§4). Our framework captures both the *adaptivity* of the accuracy-latency trade-off and the *interdependencies* among exit decisions. To model adaptivity, we formulate the objective as minimizing a weighted sum of prediction loss and latency, governed by a tunable parameter that balances accuracy against responsiveness. The interdependencies among exit decisions can be more precisely characterized by two key structural properties: (1) the decision process exhibits a Markovian structure, where each exit depends only on the outcomes of preceding ones; and (2) the overall utility is determined solely by the cumulative latency of the visited exits and the minimum observed loss. We solve this problem with an exit algorithm that achieves the online optimal loss.

For a detailed discussion on the related work, please refer to Appendix A.

1.1 Contributions

We list our contribution as follows:

- **Formalizing Exit Policies in EE models**. We formally model the exit decision problem in EE inference as a sequential policy task optimized under a latency-aware objective that balances accuracy and computational cost. We introduce two theoretical abstractions: no-recall and with-recall exit policies—both grounded in the structural properties of EE models and incorporating Markovian dependencies to reflect typical architectural design. Central to our formulation is a latency-aware loss function, parameterized by FLOPs, that captures the trade-off between accuracy and efficiency. This framework enables principled analysis of exit strategies and guides the design of inference-efficient models aligned with SLO.
- **Inapproximability of Thresholding Strategy**. We show that there is no algorithm can obtain a constant approximation of the utility of the no-recall exit problem, including the widely used thresholding strategies. Modeling the no-recall exit problem as an optimal stopping process over take-it-or-leave-it losses, we show that no algorithm can achieve a constant approximation to the offline optimum.
- **Theoretical Optimal With-Recall Strategy**. For the with-recall formulation, we derive an online-optimal policy via dynamic programming. We show that the optimal strategy reduces to adaptive thresholding: the model continues only if the current minimum loss exceeds a dynamically updated threshold that depends on recent observations. This demonstrates that allowing recall not only improves the performance but also enables provably optimal inference.
- **Empirical Evaluation**. We investigate the structure of optimal exit rules under empirical settings. Our experiments (§5) show that the theoretically optimal rule reduces to a simplified form—adaptive thresholding with recall for VGG workloads.

2 Problem Formulation

2.1 Exit Decisions of Early Exit Architecture

Early-exit (EE) models extend standard deep neural networks by introducing multiple intermediate exit points, enabling early termination of inference on certain inputs. An EE model typically consists of three components: (1) a *backbone model*, which corresponds to the original single-exit architecture; (2) a set of *ramps*, which act as intermediate exits attached to selected layers of the backbone; and (3) an *exit decision policy*, which determines whether to halt the forward pass at a given branch and return a prediction early. Notably, only the exit policy remains controllable during inference.

Thresholding. A common exit strategy in EE models is thresholding under a given service level objective (SLO). This approach assigns a fixed loss threshold to each intermediate exit. During inference, if the loss at a given exit falls below the predefined threshold, inference is terminated early, and the prediction from that exit is returned as the final output. Inference may also be terminated based on confidence thresholding, whereby execution halts once the predicted confidence score exceeds a predefined threshold. See Figure 1 for an illustration of loss thresholding under classification tasks.

Markovian Dependency. For efficiency considerations, EE models are typically designed to insert ramps at locations where the available data flow is maximally utilized [RPSC+24]. For instance, ramps are rarely placed within individual ResNet blocks, where intermediate representations may be incomplete. In this work, we adopt this common design convention. Such a structure naturally induces a *Markovian* property: the output distribution at each exit depends only on the immediately preceding exit and is conditionally independent of all earlier exits given this one.

Notations. Let n denote the number of exits in an EE model, with *the final exit being the backbone output.* Given input x, let $b_i(x)$ denote the additional data flow introduced at ramp i relative to its immediate predecessor, and $\ell_i(b_i(x))$ denote the task-specific loss at the i-th ramp. For the first ramp, b_1 denotes the data flow from the input to its output. Notably, we abuse notation by calling b_i a **box**, a representation that we adopt in our subsequent theoretical models. We denote the input dimension by d, and let T be the number of input samples to be inferred.

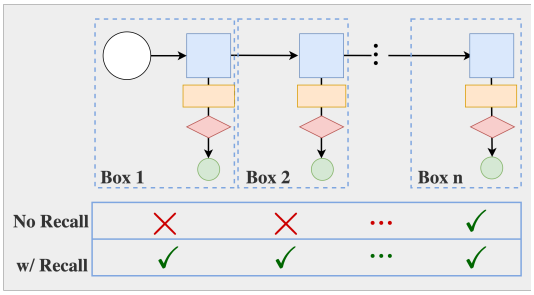


Figure 2: Box Abstraction of EE with Exit Options @ Box n

Regarding inference on a given dataset, the input at the t-th instance is denoted by $x_t \in \mathcal{X}_x \subseteq \mathbb{R}^d$ for $t \in [T]$, where $\mathcal{X}_x$ denotes the feasible input region.

2.2 From Exit Policies to Theoretical Models

Next, we formalize the exit policy as a search problem and abstract the properties of EE models into a set of assumptions. We begin by introducing our main assumptions, which are consistent with standard principles of supervised learning [GBCB16].

Assumption 2.1 (Positive Loss)**.** *For any $j \in [n]$, and any $x \in \mathcal{X}_x$, $\ell_j(b_i(x)) > 0$.*

Assumption 2.2 (Distributional Input)**.** *Each x_t, for $t \in [T]$, is assumed to be i.i.d. from a fixed distribution $\mathcal{D}$ over $\mathbb{R}^d$, matching the training input distribution.*

During inference, as the forward pass progresses through the EE model, each encountered exit ramp generates a prediction. Based on this sequential prediction, the model decides to either exit or proceed, as shown in Figure 3.

The exit policy consists of two components: deciding when to stop and which exit to select. Based on how exits are chosen, policies are classified as recall or no-recall.

Definition 2.3 (Recall vs. No-Recall Strategy)**.** *Consider an exit policy that stops at exit $i \in [n]$. It is called **no-recall** if the model must exit exactly at i, and **recall** if the model may exit at any $j \leq i$.*

For each input $x \in \mathcal{X}_x$:

1. Initialize by observing the input x and setting the ramp index $i \leftarrow 1$.
2. Perform a forward pass up to ramp i and obtain the loss at ramp i as $\ell_i(b_i(x))$.
3. The policy decides whether to stop or continue:
   - **Continue**: set $i \leftarrow i + 1$ and return to Step 2.
   - **Stop**: select an exit from previously visited ramps and output its prediction.

Figure 3: Exit policy decision process

Figure 2 shows the exit options for recall and no-recall policies that terminates at the final exit (inclusive). In both cases, the exit decision reduces to determining when to stop based on the current observation and which exit to select. In our work, we quantify the performance of exit policies using a latency-aware loss function. This objective combines accuracy and latency through a weighted sum, aligning with standard multi-task learning formulations [LZL+19].

Definition 2.4 (Latency-Aware Loss)**.** *Given an input $x \in \mathcal{X}_x$, the latency-aware loss function*

$$\theta_\lambda(j, i)[x] := \ell_j(b_j(x)) + \lambda \sum_{k=1}^{i} c_k$$

captures the total cost of stopping at the i-th exit while producing output from the $j \leq i$-th exit. Here, $c_k := \mathrm{FLOP}(b_k) \geq 0$ serve as a proxy for runtime efficiency for k-th ramp, measured in floating point operations (FLOPs). The hyperparameter $\lambda \geq 0$ balances accuracy and efficiency.

Notably, the FLOP metric may be replaced by any hardware-invariant latency measure.

3 On Exit Policies with No Recalls

The no-recall exit problem can be formally modeled as an optimal stopping problem over a sequence of losses R_i exhibiting Markovian dependencies. More precisely, since x is drawn from an underlying distribution, each loss R_i follows the distribution of the intermediate loss $\theta_\lambda(i, i)$.

Problem 3.1 (No Recall Exit Problem)**.** *Let costs $R_1 \sim \mathcal{D}_1, \ldots, R_n \sim \mathcal{D}_n$ be non-negative random variables drawn from known distributions $\mathcal{D}_1, \ldots, \mathcal{D}_n$, with a joint distribution exhibiting Markovian dependency, i.e., for all $i \in [n]$, R_{i+1} is conditionally independent of the past given R_i:*

$$\Pr(R_{i+1} \mid R_1, \ldots, R_i) = \Pr(R_{i+1} \mid R_i), \quad \textit{for all } i.$$

A decision maker sequentially observes R_1 to R_n. After observing R_i, they must either stop and paying cost R_i or irrevocably discard and continue. The goal is to design a stopping rule ALG *that minimizes the expected cost.*

Ideally, ALG is benchmarked against the offline optimal loss with perfect knowledge of all R_i.

Definition 3.2 (Offline Optimal)**.** *The benchmark is an oracle who knows all realizations in advance and selects* $\min_i R_i$*, incurring expected cost* $\mathrm{OPT} = \mathbb{E}[\min_i R_i]$.

We concluded by showing that no algorithm can achieve a constant approximation ratio for no recall exit problem, even when the underlying distribution is bounded. For details, see Section B.

Theorem 3.3 (Impossibility of Constant Approximation, No-Recall Exit)**.** *For no-recall exit problem (Prob. 3.1), no algorithm achieves a bounded α-approximation ratio, even with $n = 2$ and bounded distributions.*

This theorem establishes the impossibility of designing a no-recall exit policy that achieves any non-trivial approximation to the offline optimum. As an alternative, one might consider whether it is possible to construct an algorithm ALG that approximates the performance of a restricted class of benchmark strategies. However, as demonstrated in the preceding proof, the gap between the offline optimum and such restricted benchmarks can be arbitrarily large.

In light of these negative results, it is natural to ask whether allowing recall could circumvent these barriers and yield meaningful approximation guarantees.

4 On Exit Policies with No Recalls

Motivated by the intrinsic limitations of no-recall policies discussed earlier, we turn our attention to analyzing the efficiency of with-recall policies. The exit-with-recall problem admits a similar theoretical abstraction. The key distinction from the no-recall formulation is that we interpret the box i's loss as R_i, and define the cost of each box as λc_i.

Problem 4.1 (Exit with Recall Problem)**.** *Let costs of the boxes $R_1 \sim \mathcal{D}_1, \ldots, R_n \sim \mathcal{D}_n$ be non-negative random variables drawn from known distributions $\mathcal{D}_1, \ldots, \mathcal{D}_n$, with a joint distribution exhibiting Markovian dependency, i.e., R_{i+1} is conditionally independent of the past given R_i.*

A decision maker sequentially observes R_1 to R_n, where each box i incurs an inspection cost λc_i, as defined in latency aware loss (Def. 2.4). After observing R_i, they must either stop-incurring a total cost of $\min_{k\in[i]} R_k + \lambda \sum_{j\in[i]} c_j$ by exiting at $\arg\min_{k\in[i]} R_k$, or continue. The goal is to design a stopping rule ALG *that minimizes the expected cost.*

Since Markovian-correlated distributions with continuous support cannot be directly stored or represented without additional assumptions [EK09], we address this issue by first prequantizing the distribution into a discrete domain, and then making decisions based on the resulting discretized representation. This form of discretization is commonly used in hyperparameter search for neural networks (i.e., grid search). Therefore, WLOG, we assume that $\mathcal{D}_1, \ldots, \mathcal{D}_n$ are discrete distributions.

4.1 Comparing Against Online Optimal

It is worth noting that the counterexample in Theorem 3.3 continues to produce arbitrarily large approximation guarantees against offline optimal, even under recall is allowed. We therefore turn to a more favorable benchmark with tractable guarantees.

Definition 4.2 (Online Optimal)**.** *The benchmark is an optimal online algorithm, which has access to the joint distribution $\mathcal{D}_1, \ldots, \mathcal{D}_n$ and achieves the minimum possible expected loss without observing realizations in advance.*

By the principle of optimality, we derive the optimal exit-with-recall policy via backward induction, starting from the subproblem involving only the last box and progressively extending to the last i boxes for $i = 1 \to n$. This loss-minimizing procedure yields an elegant result: the optimal policy exits if the minimum realized loss falls below a threshold σ that adapts to the current observation.

Algorithm 1: Exit with Recall, RECALL

Require: Exits $\{b_1, \ldots, b_n\}$, latency cost $\{c_1, \ldots, c_n\}$, threshold $\sigma(i, s)$ (Def. 4.4) for all i and $s \in V$.

Initialize minimum reward $X \leftarrow \infty$, $i \leftarrow 1$, $\sigma \leftarrow \sigma(1, \emptyset)$.
while $X > \sigma$ **do**
    Pay c_i to open box b_i, observe loss R_i.
    $X \leftarrow \min\{X, R_i\}$. ▷ Update min reward.
    $\sigma \leftarrow \sigma(i + 1, R_i)$. ▷ Update threshold.
    $i \leftarrow i + 1$.
end while
Return box b_j that is opened and with the min reward.

4.2 Exit with Recall via Adaptive Thresholding

Algorithm 1 presents the structure of the optimal stopping rule. At each step, the algorithm updates a threshold σ—conditioned on R_i—to decide whether to stop given the current minimum loss. Thus, *the stopping strategy at each i-th exit can be represented as a stop/no-stop table* indexed by the possible values of minimum realized reward X and the state R_i of i-th ramp.

Additional Notations. We assume that each reward R_i for $i \in [n]$ takes values from a common finite support $V = v_1, \ldots, v_k$. For each $i \in [n]$, we let $\mathbf{p}_i$ be the probability density function (PDF) of R_i, where $\mathbf{p}_i[s_q] = \Pr[R_i = s_q]$. We denote by $P_i \in \mathbb{R}+^{k\times k}$ the transition matrix from $\mathcal{D}_i$ to $\mathcal{D}_{i+1}$, such that $\mathbf{p}_{i+1} = \mathbf{p}_i \cdot P_{i+1}$. We use τ and π interchangeably to denote the strategy/ stopping time.

We now formally describe the backward induction procedure to compute the adaptive threshold function σ. Any stopping time τ depends only on the current minimum reward X, the most recent loss R_i, and the next candidate index $i+1$. We refer to the tuple $(X, R_i, i{+}1)$ as the algorithm's *state*, and proceed to derive the expected loss incurred at this state under a given stopping time τ.

Definition 4.3 (Equivalent Loss)**.** *Given τ and (x, R_{i-1}, i), we define the expected loss of the state (X, R_{i-1}, i) following stopping rule τ as $\Phi^\tau(X, R_{i-1}, i)$.*

$$\Phi^\tau(X, R_{i-1}, i) := \mathbb{E}[\min\{X, \min_{j=i}^{\tau(X,R_{i-1},i)} R_j\} + \sum_{j=i}^{\tau(X,R_{i-1},i)} c_j]$$

In addition, we use

$$\Phi(X, R_{i-1}, i) = \Phi^{\tau^*}(X, R_{i-1}, i) = \min_\tau \Phi^\tau(X, R_{i-1}, i)$$

to denote the expected future loss following the optimal strategy τ^* starting at state (X, R_{i-1}, i).

Let Φ be the expected loss of a state, then Φ can be solved inductively using Bellman's principle of optimality as below:

$$\Phi^{\tau^*}(X, R_{i-1}, i) = \min\Big\{X, c_i + \mathop{\mathbb{E}}_{R_i|R_{i-1}}[\phi^{\tau^*}(\min\{X, R_i\}, R_i, i+1)]\Big\}.$$

where the first term corresponds to stopping immediately, and the second to continuing and applying the optimal stopping time τ^* for the remaining boxes.

Importantly, for fixed R_{i-1} and i, there exists a maximal X such that the decision maker is *indifferent* between stopping and opening the next box. This value defines the adaptive threshold σ that governs our algorithm (Alg. 1).

Definition 4.4 (Adaptive Threshold)**.** *Given any state (X, R_{i-1}, i), we define the adaptive threshold at the current state, denoted by $\sigma_i(X, R_{i-1}, i)$, as the largest solution to:*

$$\mathbb{E}\left[\Big(\sigma - \min_{j=i}^{\tau^*(\sigma,R_{i-1},i)} R_j\Big)_+ - \sum_{j=i}^{\tau^*(\sigma,R_{i-1},i)} c_j\right] = 0, \tag{1}$$

where τ^* is the optimal strategy.

Next, we prove that the threshold is not only well-defined but also optimal, thereby justifying the correctness of our algorithm (Alg. 1). In addition, this threshold could be implemented efficiently. For large sample sizes, the stop/continue decision at each state $(X, R_i, i{+}1)$ can be precomputed and stored in a table with space complexity $O(n|V|^2)$. During inference, the algorithm simply queries this table to decide whether to stop at each step, requiring $O(1)$ time per iteration and thus $O(n)$ time per sample. See Sec. C for more details.

Theorem 4.5 (Optimality and Efficiency of Adaptive Thresholding)**.** *For every state $(X, R_i, i{+}1)$, a solution to* (1) *exists and is independent of the current minimum loss X. Consequently, there is a well-defined adaptive threshold $\sigma(R_i, i{+}1)$ such that the policy that stops if $\sigma(R_i, i{+}1) > X$ and continues otherwise (Alg. 1) is online optimal.*

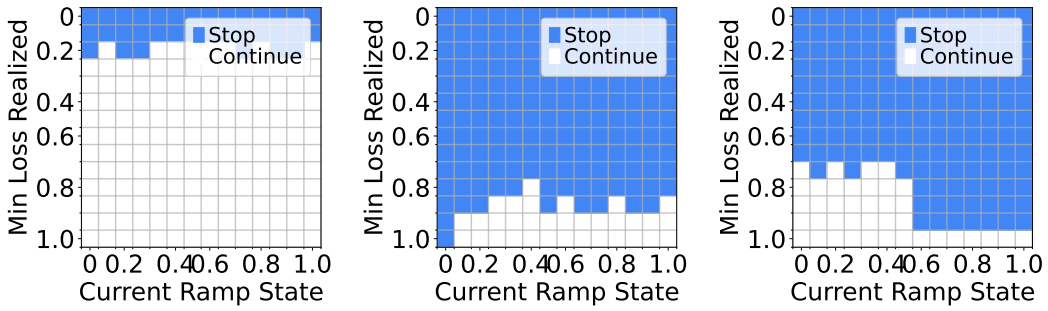
Moreover, the stopping decision for every state (X, R_i, i) for all $i \in [n]$ and $R_i, X \in V$ can be computed in total time $O(n|V|^2)$ and stored in a table of size $O(n|V|^2)$, enabling $O(1)$ inference time per decision and $O(n)$ time per sample.

Remark. The theoretical optimality of the above Bayesian conditioning extends beyond the Markov assumption and remains valid under arbitrary correlation. In the single directed line setting, there is no routing choice and each step is simply a stop-or-continue decision. By Bellman's principle of optimality, the result therefore holds under both Markovian and non-Markovian dependence. In the experimental sections, we implement the algorithm under the Markovian correlation model.

5 Experiments

5.1 Datasets and Models

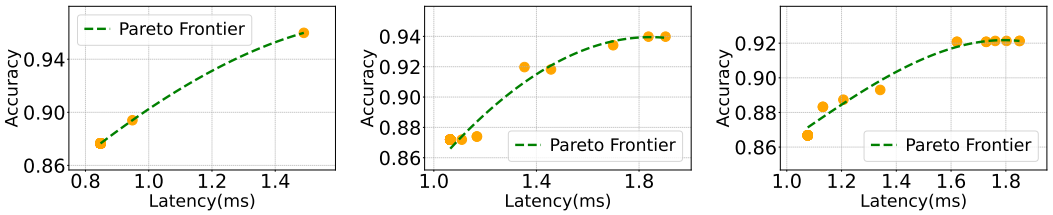
We evaluate our optimal stopping strategy RECALL on both synthetic datasets and real-world traces of CV workloads. For synthetic dataset, the loss histories are generated by sampling each loss independently from a uniform distribution over $[0, 1]$, with ramp latency cost fixed at 0.1 millisecond. For this setting, we fit the if-stop decision matrix at each ramp. The purpose is not to emulate practical workloads—where ramp losses are typically positively correlated—but rather to elucidate the general structure of the optimal strategy through analysis of the resulting if-stop matrices.



(a) Truncated Normal $\mathcal{N}(0.5, 0.5)$ (b) Truncated LogNormal $\mathcal{N}(0, 1)$ (c) Uniform $\mathcal{U}(0, 0.5)$

Figure 4: Visualization of the If Stop matrix for Synthetic Data

For CV experiments, we utilize the VGG11, VGG13 and VGG16 [SZ14]. These models are sourced from the PyTorch Model Zoo [PyT23] with weights pre-trained on ImageNet [DDS+09]. The CV workloads are drawn from real-time object recognition tasks using 8 one-hour videos from recent video analytics research [AN23, HAB+18].



(a) VGG-11, Vanilla Latency 3.26ms (b) VGG-13, Vanilla Latency 3.83ms (c) VGG-16, Vanilla Latency 4.51ms

Figure 5: Accuracy–latency trade-off by RECALL for VGG-11, VGG-13, and VGG-16.

Experimental Setup. Our testbed consists of a single server equipped with two Intel Xeon Gold 6438M processors (3.90 GHz), 512 GB of DDR5 memory, and one NVIDIA RTX. 4000 Ada GPU. The server runs Ubuntu 22.04 with Linux kernel 5.15, CUDA version 12.9, and Python 3.10.

Metrics and Baselines. We evaluate empirical accuracy by comparing model outputs against those of the *backbone model*. This represents the best attainable accuracy in our setting, as it is defined relative to the backbone rather than the true labels of the inputs.

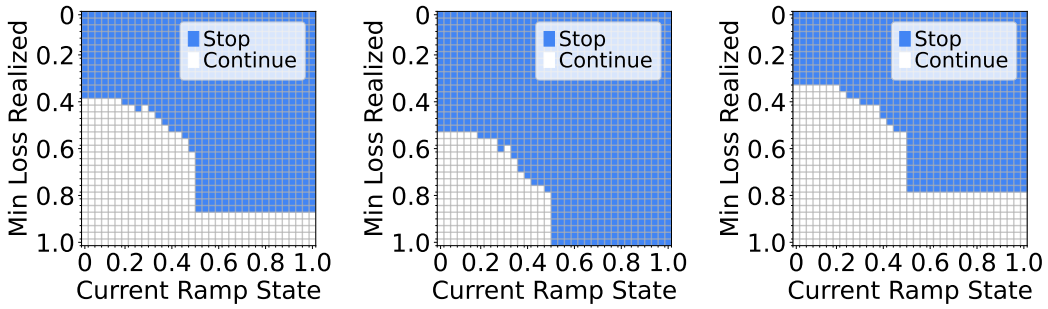
5.2 Accuracy–latency trade-off of Computer Vision Models

We conduct experiments with 1,500 training samples and 500 test samples from the VGG early exit model, each configured with three ramps, where the final exit corresponds to the third ramp. Each sample is discretized in increments of 0.08 over the range $[0, 1]$, and we use $1 - \text{SOFTMAX}$ as the loss function. Figure 5 highlights two key findings. First, our method reliably recovers the Pareto frontier,

with deviations attributable only to discretization and test-time randomness. Second, it achieves up to a $\mathbf{2}\times$ speedup over the backbone model with approximately 8% accuracy loss. We further note that this loss is largely induced by the discretization step.

5.3 If Stop Decision Matrix

Next, we examine the if-stop decision matrix selected by our RECALL strategy in the VGG early-exit model in Figure 6. In this matrix, the entry at row i and column j corresponds to the case where the current minimum is $(i-1)\times 0.08$ and the current state is $(j-1)\times 0.08$, indicating whether the strategy chooses to stop or to continue inference on the current sample. Interestingly, we find that for the VGG workload, the optimal strategy roughly reduces to a thresholding rule that adapts to the current minimum. This observation offers a theoretical perspective that helps explain the empirical success of thresholding strategies.



(a) Caption A (b) Caption B (c) Caption C

Figure 6: Visualization of the If Stop matrix for VGG11,13, and 16.

In general, however, our RECALL strategy does not exhibit such a simple structure. To illustrate this, we evaluate the if-stop matrices on 1,000 samples drawn from truncated uniform, normal, and log-normal distributions over $[0,1]$. For a fixed realized minimum loss, the stop decisions are essentially random, reflecting the inherent randomness of our sampling procedure.

6 Conclusions

In this work, we establish a principled foundation for designing provably efficient exit policies in early-exit (EE) models. By casting the exit decision as a sequential stopping problem, we introduce a latency-aware framework that formally distinguishes between no-recall and with-recall strategies, grounded in the structural properties of EE architectures. Within this framework, we prove that no-recall policies—including widely adopted thresholding rules—cannot guarantee any constant-factor approximation to the offline optimum. In contrast, we derive the first online-optimal algorithm for with-recall strategies, showing that recall fundamentally changes the problem landscape and enables provably optimal inference through adaptive thresholding.

Our empirical evaluation further demonstrates that, in practical workloads, the theoretically optimal strategy simplifies to adaptive thresholding with recall, bridging theory and practice. Together, these results provide new theoretical insights into constrained inference, highlight the limitations of existing heuristics, and chart a pathway toward principled, efficient deployment of EE models.

Looking forward, several directions merit exploration. One is to validate whether the empirical structures we observe (e.g., the reduction of recall strategies to adaptive thresholding) extend to other model families beyond VGG. Another is to investigate improved quantization schemes for discretization, so that accuracy loss induced by coarse additive steps can be further reduced. Finally, a natural extension is to integrate our theoretical framework with models that support "skip" behavior, such as SkipNet for image classification [WYD+18], thereby enriching the design space of efficient inference strategies.

References

[AN23] Neil Agarwal and Ravi Netravali. Boggart: Towards {General-Purpose} acceleration of retrospective video analytics. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 933–951, 2023.

[BB12] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[BFLL20] Shant Boodaghians, Federico Fusco, Philip Lazos, and Stefano Leonardi. Pandora's box problem with order constraints. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 439–458, 2020.

[CGT+20] Shuchi Chawla, Evangelia Gergatsouli, Yifeng Teng, Christos Tzamos, and Ruimin Zhang. Pandora's box with correlations: Learning and approximation. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1214–1225. IEEE, 2020.

[DCLT19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[DPI+24] Yinwei Dai, Rui Pan, Anand Iyer, Kai Li, and Ravi Netravali. Apparate: Rethinking early exits to tame latency-throughput tensions in ml serving. In *Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles*, pages 607–623, 2024.

[EK09] Stewart N Ethier and Thomas G Kurtz. *Markov processes: characterization and convergence*. John Wiley & Sons, 2009.

[GBCB16] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[GGM06] Ashish Goel, Sudipto Guha, and Kamesh Munagala. Asking the right questions: Model-driven optimization using probes. In *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 203–212, 2006.

[GR16] Rishi Gupta and Tim Roughgarden. A pac approach to application-specific algorithm selection. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 123–134, 2016.

[HAB+18] Kevin Hsieh, Ganesh Ananthanarayanan, Peter Bodik, Shivaram Venkataraman, Paramvir Bahl, Matthai Philipose, Phillip B Gibbons, and Onur Mutlu. Focus: Querying large video datasets with low latency and low cost. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 269–286, 2018.

[HKY18] Elad Hazan, Adam Klivans, and Yang Yuan. Hyperparameter optimization: a spectral approach. In *International Conference on Learning Representations*, 2018.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[JT16] Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*, pages 240–248. PMLR, 2016.

[KHD19] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *International conference on machine learning*, pages 3301–3310. PMLR, 2019.

[KS77] Ulrich Krengel and Louis Sucheston. Semiamarts and finite values. 1977.

[KS78] Ulrich Krengel and Louis Sucheston. On semiamarts, amarts, and processes with finite value. *Probability on Banach spaces*, 4(197-266):1–2, 1978.

[LJD+17] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.

[LKL21] Stefanos Laskaridis, Alexandros Kouris, and Nicholas D Lane. Adaptive inference through early-exit networks: Design, challenges and directions. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, pages 1–6, 2021.

[LM24] Vasilis Livanos and Ruta Mehta. Minimization is harder in the prophet world. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 424–461. SIAM, 2024.

[LZL+19] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. *Advances in neural information processing systems*, 32, 2019.

[PyT23] PyTorch. Model zoo. `https://pytorch.org/serve/model_zoo.html`, 2023.

[RPSC+24] Haseena Rahmath P, Vishal Srivastava, Kuldeep Chaurasia, Roberto G Pacheco, and Rodrigo S Couto. Early-exit deep neural network-a comprehensive survey. *ACM Computing Surveys*, 57(3):1–37, 2024.

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[SZ15] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR 2015)*. Computational and Biological Learning Society, 2015.

[TMK16] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 2464–2469. IEEE, 2016.

[Wei79] Martin L Weitzman. Optimal search for the best alternative. *Econometrica: Journal of the Econometric Society*, pages 641–654, 1979.

[WYD+18] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 409–424, 2018.

[XTL+20] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. Deebert: Dynamic early exiting for accelerating bert inference. *arXiv preprint arXiv:2004.12993*, 2020.

[XTYL21] Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. Berxit: Early exiting for bert with better fine-tuning and extension to regression. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: Main Volume*, pages 91–104, 2021.

[ZCL+19] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, 2019.

[ZXG+20] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. Bert loses patience: Fast and robust inference with early exit. *Advances in Neural Information Processing Systems*, 33:18330–18341, 2020.

Appendix

A Related Work

Early Exit Models. Early exit architectures have been widely adopted to accelerate inference in both computer vision and natural language processing, including in ResNet-[HZRS16], VGG-[SZ15], and BERT-based [DCLT19] models, see [RPSC$^+$24, LKL21] for a recent survey on EE models. The ramp architectures in these models are typically designed to align with the structural characteristics of the underlying backbone. Existing exit strategies commonly rely on metrics such as label confidence [XTYL21], prediction entropy [XTL$^+$20, TMK16], or more advanced mechanisms like ramp-level counters [ZXG$^+$20]. However, none of these approaches consider prediction with recall, which allows returning to the output of a previously visited exit—a setting where, as empirical evidence suggests, earlier exits can occasionally yield more accurate predictions than later ones [KHD19]. More recently, Apparate [DPI$^+$24] proposed adaptive thresholds that evolve during runtime. However, their method requires every inference to run to the end of the backbone model. In contrast, our work leverages the open-sourced CV and NLP workload traces released with their paper to evaluate exit strategies that allow early termination.

Pandora's Box and Prophet Inequality. This work is also relevant to the algorithmic game theory literature, particularly the Pandora's Box [Wei79, BFLL20, CGT$^+$20] and Prophet Inequality frameworks [KS77, KS78, LM24]. Specifically, our exit-with-recall model bears structural similarity to the Pandora's Box problem, while the no-recall variant aligns closely with the Prophet Inequality setting—both situated within a cost minimization framework. However, our setting imposes two key constraints: (i) a precedence constraint on the order in which boxes (or exits) may be inspected, and (ii) a Markovian correlation that the underlying distributions conform to. These constraints render existing algorithms for the aforementioned problems inapplicable.

Data-Driven Algorithm Design. Our framework relates to data-driven algorithm design [GR16] with cost, where practitioners refine parameterized algorithms via training instances to maximize expected future performance, including [GGM06, BB12, JT16, LJD$^+$17, HKY18].

B More details from Exit with No Recall

As achieving the offline optimal loss is generally infeasible without full future information, the best attainable guarantee lies in bounding the approximation ratio.

Definition B.1 (Approximation Ratio)**.** *We say* ALG *is an* α*-approximation if* $\mathbb{E}[\mathrm{ALG}] \leq \alpha \cdot \mathrm{OPT}$*, for some* $\alpha \geq 1$*.*

Theorem B.2 (Impossibility of Constant Approximation, No-Recall Exit)**.** *For no-recall exit problem (Prob. 3.1), no algorithm achieves a bounded* α*-approximation ratio, even with* $n = 2$ *and bounded distributions.*

Proof Sketch. Let $n = 2$ and $\alpha > 1$ be an arbitrary large constant. Consider the following random variables:

$$R_1 = \frac{1}{\alpha^2} \qquad \text{w.p. } 1, \qquad\qquad R_2 = \begin{cases} 0 & \text{w.p.} 1 - \frac{1}{\alpha}, \\ \frac{1}{\alpha} & \text{w.p. } \frac{1}{\alpha} \end{cases}$$

Under this construction, any algorithm achieves an expected reward of exactly $1/\alpha^2$, but a prophet achieves $\mathrm{OPT} = 1/\alpha^3$. This indicates an α-competitive ratio. This competitive ratio can be made arbitrarily large by increasing α. □

C More details from Exit with Recall

C.1 More Details for the Adaptive Threshold

Lemma C.1 (Properties of Φ and H_i)**.** *Given any state* (x, R_{i-1}, i)*,*

- $\Phi(\cdot, R_{i-1}, i)$ *is 1-Lipschitz and monotone non-decreasing.*

- *Let $H_i(x, R_{i-1}) := \Phi(x, R_{i-1}, i) - x$, then $H_i(\cdot, R_{i-1})$ is nonnegative, 1-Lipschitz and monotone non-increasing.*
- *For z_i as the threshold (Def.4.4) of the i-th box, then $\Phi(x, R_{i-1}, i) = x$ for any $x \geq z_i$.*

Proof. Given any $a < b$,

$$\begin{aligned}
&\Phi(b, R_{i-1}, i) - \Phi(a, R_{i-1}, i) \\
&\leq \mathbb{E}\left[\min\{b, \min_{j=i}^{\tau^*(a,R_{i-1},i)} R_j\} - \min\{a, \min_{j=i}^{\tau^*(a,R_{i-1},i)} R_j\}\right] \\
&\leq b - a
\end{aligned}$$

where in the first inequality, we used that $\tau^*(b, R_{i-1}, i)$ is a suboptimal strategy for $\Phi^\tau(a, R_{i-1}, i)$. Thus, $\Phi(\cdot, R_{i-1}, i)$ is monotone non-decreasing. Now consider H_i, we have

$$H_i(b, R_{i-1}) - H_i(a, R_{i-1}) = \Phi(b, R_{i-1}, i) - \Phi(a, R_{i-1}, i) - (b - a) \leq 0$$

where we use that $\Phi(\cdot, R_{i-1}, i)$ is 1-Lipschitz. The above inequality implies that $H_i(x, R_{i-1})$ is 1-Lipschitz and monotone non-increasing. Lastly, $\Phi(x, R_{i-1}, i) - x = 0$ for all $x \geq z_i$ follows from the that fact that z_i is the smallest such that $H_i(z_i, R_{i-1}) = 0$ and H_i is non-negative and monotone non-increasing.

□

Lemma C.2. *The smallest solution to (1) exists, and hence Definition 4.4 is well defined. Given current state (x, R_{i-1}, i), if the generalized reservation value $z_i = x$, then there exists some optimal stopping time $\tau^*(x, R_{i-1}, i) \geq i$.*

Proof. Given any state (x, R_{i-1}, i), consider function

$$H_i(x, R_{i-1}) = \Phi(x, R_{i-1}, i) - x$$

$H_i(x)$ is 1-Lipschitz and monotone non-increasing by lemma C.1. Since $H_i(0, R_{i-1}) = \Phi(0, R_{i-1}, i) \geq 0$ and $H_i(s_K, R_{i-1}) = 0$, there exist some $z_i \in S$, such that $H_i(z_i, R_{i-1}) = 0$. This proves the existence of z_i.

Now, we show that if $x = z_i$ is positive, then there exists an optimal stopping rule that proceeds to open b_i. Fix any i such that $z_i > 0$. Let $\widetilde{\tau}$ be the best strategy among all strategies that open b_i. To show that $\widetilde{\tau}$ is indeed optimal, we show that

$$\delta = \Phi(z_i, R_{i-1}, i) - \Phi^{\widetilde{\tau}}(z_i, R_{i-1}, i) = 0$$

Assume towards contradiction that $\delta > 0$. We have

$$\begin{aligned}
0 < \delta &= \Phi(z_i, R_{i-1}, i) - \Phi^{\widetilde{\tau}}(z_i, R_{i-1}, i) \\
&\leq \Phi(z_i, R_{i-1}, i) - \Phi^{\tau^*(z_i-\epsilon, R_{i-1}, i)}(z_i, R_{i-1}, i) \\
&= (\Phi(z_i, R_{i-1}, i) - \Phi(z_i - \epsilon, R_{i-1}, i)) + (\Phi(z_i - \epsilon, R_{i-1}, i) - \Phi^{\tau^*(z_i-\epsilon, R_{i-1}, i)}(z_i, R_{i-1}, i)) \\
&\leq 2\epsilon
\end{aligned}$$

where we used Lipschitzness of Φ for the last inequality, and the first inequality comes from the fact that $\tau^*(z_i - \epsilon, R_{i-1}, i)$ is a sub-optimal policy that opens b_i. We have $\tau^*(z_i - \epsilon, R_{i-1}, i) \geq i$ since z_i is the smallest such that $H_i(z_i, R_{i-1}) = 0$, this implies that $H_i(z_i - \epsilon, R_{i-1}) = \Phi^{\tau^*(z_i-\epsilon, R_{i-1}, i)}(z_i - \epsilon, R_{i-1}, i) - (z_i - \epsilon) > 0$ meaning the optimal policy will accumulate more reward than current best, thus it has to open b_i. As $\epsilon \to 0$, we get a contradiction. On the other hand, $H_i(z_i, R_{i-1}) = \Phi(z_i, R_{i-1}, i) - z_i = 0$ implies that the strategy that stops at b_{i-1} is also optimal. Thus, z_i is indeed the value for which we are indifferent between stopping and proceeding optimally. □

Theorem C.3 (Optimality of Adaptive Thresholding)**.** *Given the current state $(X, R_i, i+1)$, there exists a solution to* (1)*. This solution is independent of the current minimum loss X and can be denoted by $\sigma(R_i, i+1)$.*

The policy that stops based on the adaptive threshold σ—specifically, stopping when $\sigma > X$ and continuing otherwise (Alg. 1)—achieves online optimality.

Proof Sketch. To establish the uniqueness of σ, we study the properties of $\Phi(X, R_{i-1}, i)$, and prove that it is monotonically nondecreasing with respect to X. This implies that the adaptive threshold such that $\Phi(X, R_{i-1}, i) = X$ exists.

We show that when increasing X, the function $X - \Phi(X, R_{i-1}, i)$ exhibits a transition from negative to positive. This monotonicity allows us to perform binary search over X on the indicator function $\mathbf{1}\{X - \phi(X, R_{i-1}, i) > 0\}$ to identify the threshold point.

□

C.2 If Stop Table

Lemma C.4 (Efficient Computation of If-Stop Table)**.** *There is an efficient algorithm that computes the stopping decision for every state (X, R_i, i) for all $i \in [n]$, and $R, X \in V$ in $O(n \cdot |V|^2)$.*

Proof. Now we give an efficient algorithm for computing generalized reservation value. In fact, we will give an algorithm that uses dynamic programming to compute $\Phi(x, R_{i-1}, i)$ for all triples (x, R_{i-1}, i). Then, given the current state of the algorithm (x, R_{i-1}, i), the reservation value z_i for box i is the largest x in the table where $\Phi(x, R_{i-1}, i) = x$.

Denote by $T(x, R_{i-1}, i)$ our three dimensional dynamic programming table. Each entry $T(x, R_{i-1}, i)$ will store the following information:

1. Expected future reward: $\Phi(x, R_{i-1}, i)$
2. Indicator: $\mathbb{1}(x, R_{i-1}, i)$ indicating whether the optimal policy will open b_i in this state
3. The distribution of future random min reward[1]: $R_{\text{FRM}}(x, R_{i-1}, i) := \min_{j=i}^{\tau^\star(x,R_{i-1},i)} R_j$ where R_j's are the correlated random rewards for miniboxes that are yet to be opened given that the algorithm is at state (x, R_{i-1}, i).
4. The distribution of future random cost[2]: $c_{\text{FR}}(x, R_{i-1}, i) := \sum_{j=i}^{\tau^\star(x,R_{i-1},i)} c_j$

Since all random variables that appear in Algorithm we just described have finite support with size bounded by $\mathrm{poly}(K, n)$, and any $\min$ operation for random variables only has three or less arguments, it follows that constructing the table will only cost $O(n|V|^2)$. □

[1]The randomness comes from both random stopping time τ^* and correlated random variables R_i's,

[2]The randomness comes from τ^* being a random stopping time.