Regret Bounds for Online Learning for Hierarchical Inference

Ghina Al-Atat
IMDEA Networks Institute, Madrid
Spain

Puranjay Datta
Indian Institute of Technology Bombay, Mumbai
India

Sharayu Moharir
Indian Institute of Technology Bombay, Mumbai
India

Jaya Prakash Champati
IMDEA Networks Institute, Madrid
Spain

ABSTRACT

Hierarchical Inference (HI) has emerged as a promising approach for efficient distributed inference between end devices deployed with small pre-trained Deep Learning (DL) models and edge/cloud servers running large DL models. Under HI, a device uses the local DL model to perform inference on the data samples it collects, and only the data samples on which this inference is likely to be incorrect are offloaded to a remote DL model running on the server. Thus, gauging the likelihood of incorrect local inference is key to implementing HI. A natural approach is to compute a confidence metric for the local DL inference and then use a threshold on this confidence metric to determine whether to offload or not. Recently, the HI online learning problem was studied to learn an optimal threshold for the confidence metric over a sequence of data samples collected over time. However, existing algorithms have computation complexity that grows with the number of rounds and do not exhibit a sub-linear regret bound. In this work, we propose the Hedge-HI algorithm and prove that it has $O\left(T^{\frac{2}{3}}\mathbb{E}_{\mathbf{Z}}[N_T]^{\frac{1}{3}}\right)$ regret, where T is the number of rounds, and N_T is the number of distinct confidence metric values observed till round T. Further, under a mild assumption, we propose Hedge-HI-Restart, which has an $O\left(T^{\frac{2}{3}}\log(\mathbb{E}_{\mathbf{Z}}[N_T])^{\frac{1}{3}}\right)$ regret bound with high probability and has a much lower computation complexity that grows sub-linearly in the number of rounds. Using runtime measurements on Raspberry Pi, we demonstrate that Hedge-HI-Restart has a runtime lower by order of magnitude and achieves cumulative loss close to that of the alternatives.

CCS CONCEPTS

• **Theory of computation** → **Online learning algorithms**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Edge AI, Deep Learning, Online Learning, Hierarchical Inference

ACM Reference Format:
Ghina Al-Atat, Puranjay Datta, Sharayu Moharir, and Jaya Prakash Champati. 2024. Regret Bounds for Online Learning for Hierarchical Inference. In *The Twenty-fifth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing(MOBIHOC '24), October 14–17, 2024, Athens, Greece*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3641512.3686392

1 INTRODUCTION

Deep Learning (DL) inference on data generated by end devices such as IoT sensors, smartphones, and drones results in significant optimizations and efficiency gains across industrial automation, smart cities, remote healthcare, smart agriculture, and other sectors. Given the compute and memory requirements of the DL models, DL inference has been primarily cloud-centric [21]. However, processing and analyzing data locally reduces response time and network bandwidth usage, enhances privacy, and improves scalability. This has initiated significant research efforts toward deploying DL models on end devices using model compression techniques such as pruning, quantization, and model distillation [7, 9]. Therefore, performing DL inference tasks at the edge, i.e., computing inference for data samples on end devices and/or edge servers efficiently in terms of execution time, energy, and accuracy, is gaining importance [33].

Existing techniques for DL inference at the edge can be broadly classified into: *(i)* on-device inference [25], *(ii)* DNN-partitioning [17], *(iii)* inference load balancing [12], and *(iv)* Hierarchical Inference (HI) [22]. On-device inference is doing inference locally without further support from edge/cloud servers. There exists a plethora of DL models, e.g., MobileNet [26], EfficientNet [28], Gemma 2B large language model [8], that can be embedded on moderately powerful smartphones, and tinyML models [3] that can be embedded on extremely resource-limited IoT devices such as micro-controller units. However, due to their small size, these on-device or local DL models have relatively poor inference accuracy, which limits their generalization capability and robustness to noise. The DNN partitioning technique performs inference execution of a large-size DL model by partitioning the model's layers between a device and a remote server. However, the benefits of this technique were only realized for computationally powerful devices (such as smartphones) with mobile GPUs [10, 11]. The inference load balancing technique divides the computational load between the device and a server to minimize the time or energy consumption per inference [12, 23, 24]. Recently, HI has emerged as a promising distributed DL inference technique [2, 4, 5, 22, 23]. Under HI, inference is performed first on the local DL model for each data sample. A data sample is offloaded for remote DL inference on an edge/cloud server only if the local inference is likely to be incorrect. In contrast to on-device inference, HI considers inference offloading. Further, HI examines the local inference before making an offloading decision, thus improving the existing inference load-balancing algorithms. Unlike DNN partitioning, HI is appealing for resource-constrained devices as one may design customized local DL models for these devices and use off-the-shelf high-accuracy DL models on the remote server.

Gauging the likelihood of an incorrect inference is key to implementing HI. For classification applications, the local DL model outputs a soft-max (confidence) value for each class. A data sample is typically classified into the class with the maximum soft-max value. Past works on HI used a threshold on the maximum soft-max value to decide whether to accept the local inference or offload the data sample for the remote inference. In [2, 4, 23], the authors computed the threshold offline using the labeled training data and fixed offloading costs. Since the data samples in the inference phase differ from those in the training phase, in [22], the authors studied the HI Learning (HIL) problem to learn the best threshold to make the offloading decisions. In the HIL problem [22], the data samples arrive periodically, and the device uses the local DL model to obtain the maximum soft-max values. In each period or round t, if the device accepts the local inference and is correct, it does not incur any loss. Otherwise, it incurs a loss of 1. If the device rejects the inference and offloads, it incurs a fixed loss. An optimal fixed threshold for the maximum soft-max value is the one that minimizes the total loss in T rounds. The problem was posed as a variant of the classical Prediction with Expert Advice (PEA) [6]. The authors proposed an online algorithm with computation complexity $O(t)$ in round t, and proved $O(\sqrt{T \log 1/\lambda_{\min}})$ regret bound, where $\lambda_{\min}$ is the minimum difference between any two distinct maximum soft-max values seen in the T rounds. Note that $\lambda_{\min}$ is dataset-dependent. Further, the runtime $O(t)$ for inference in round t is prohibitive at larger values of t since the HIL algorithm needs to run on resource-constrained devices. The authors in [5] extended the HIL problem setting to multiple devices and studied maximizing average accuracy subject to average offloading cost constraint under the assumption that the offloading costs are i.i.d. They solve the underlying HIL problem by discretizing the soft-max values and assuming that the cumulative loss of the optimal threshold and the minimum cumulative loss achieved from the discretized soft-max values are bounded by a small constant. Thus, their algorithm's regret bound and computation complexity depend on the resolution at which the soft-max values are discretized. This raises the question if the HIL problem has a sub-linear regret algorithm. Further, both works present regret-bound analysis, considering that the remote DL has 100% accuracy.

The HIL problem we study is presented in Figure 1. In contrast to [5, 22], we consider a more general setting where the remote DL model is not 100% accurate. Further, unlike [5, 22], we consider general loss (or cost) functions with values in $[0, 1]$ and do not assume that the offloading costs are i.i.d. Our key contributions are summarized below:

– We show that a special case of the HIL problem is related to the branching experts problem [14] and establish that any online algorithm for HIL has a regret bound at least $\sqrt{T\mathbb{E}_{\mathbf{Z}}[N_T]}$, where $\mathbb{E}_{\mathbf{Z}}[N_T] \leq T$ is the expected number of distinct soft-max values observed in T rounds. Thus, for the HIL problem, a sub-linear bound only exists if $\mathbb{E}_{\mathbf{Z}}[N_T]$ is sub-linear in T.
– We propose a novel algorithm called Hedge-HI and show that its regret is $O\left(T^{\frac{2}{3}}\mathbb{E}_{\mathbf{Z}}[N_T]^{\frac{1}{3}}\right)$.
– Noting that computation complexity of Hedge-HI, and also the algorithm in [22], grows linearly in round t, we propose Hedge-HI-Restart, which has $O\left(\min\left(t, \sqrt{T}\right)\right)$ computation complexity. Further, under a mild assumption, we prove that, with high probability, it has a $O\left(T^{\frac{2}{3}}\log(\mathbb{E}_{\mathbf{Z}}[N_T])^{\frac{1}{3}}\right)$ regret bound. Unlike the regret bound for Hedge-HI, this regret bound for Hedge-HI-Restart is sub-linear even if $\mathbb{E}_{\mathbf{Z}}[N_T]$ grows linearly with T.
– Via simulations, we demonstrate that Hedge-HI-Restart has a much lower runtime while achieving cumulative loss close to that of Hedge-HI and the algorithm in [22].

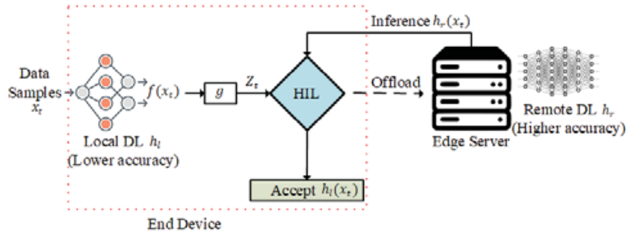


Figure 1: An edge intelligence system comprising an end device, embedded with a small-size DL model h_l, and an edge server with a large-size DL model h_r. The HI learning algorithm utilizes the local DL model's confidence (Z_t) to accept the local inference or offload the data sample. The functions f and g are defined in Section 3.

2 RELATED WORKS

In this section, we discuss the related works on DL inference at the network edge, present the contrasting differences with [5, 22] in detail, and present related works on PEA.

2.1 DL Inference at the Edge

Inference Load Balancing. Since the initial proposal of edge computing in [27], significant attention has been devoted to the computational offloading for generic compute-intensive applications [20]. Due to the growing importance of edge intelligence, recent studies have examined computation offloading for applications using ML inference [12, 24, 31]. This line of research aims to load-balance the inference task by partitioning the data samples between the device and the server, considering parameters such as job execution times, communication times, energy consumption, and average test accuracy of ML models. However, in [2], we demonstrated that load-balancing data samples may lead to adversarial scenarios where most data samples scheduled on the device receive incorrect inferences. HI circumvents this issue by utilizing the local DL output and offloading a data sample if its local inference is likely incorrect.

DNN Partitioning. DNN partitioning, proposed in [17], partitions the layers of a large-size DL model and deploys the front layers on the mobile device while the deep layers are deployed on an edge server. It received considerable attention recently [16, 19]. However, as demonstrated in [2], for DNN partitioning to be beneficial in reducing the execution time per inference, the processing times of the DL layers on the mobile device should be small relative to the communication time of the data generated between layers. Thus, this technique requires mobile GPUs making it infeasible for resource-constrained end devices such as IoT devices.

On-Device Inference. As discussed in Section 1, considerable effort has been dedicated to the design of compact DL models tailored for deployment on end devices [7, 9, 25]. Further, techniques such as early-exit [29] have been explored to reduce the execution time of DL inference. As a result, state-of-the-art compact DL models can perform complex ML inference tasks on devices, ranging from face recognition on mobile phones [18] to visual wake word and keyword spotting on IoT devices [3]. However, these models have fewer neurons, which results in lower inference accuracy and limits their generalization capability and robustness to noise. Additionally, it may be infeasible to perform post-deployment active learning to improve the accuracy of the models on extremely resource-constrained devices such as IoT sensors or MCUs. Therefore, offloading data samples to edge servers or the cloud equipped with high-accuracy DL models is essential for reliable inference.

Hierarchical Inference. Given the drawbacks of the previous approaches, HI has received considerable attention in the recent past [2, 4, 5, 22, 23]. In [23], the authors computed the HI offloading decision by computing a threshold for the maximum soft-max value based on the transmission energy constraint of the device. In [2], we proposed a general definition for HI, presented multiple use cases, and compared HI with existing distributed DL inference approaches at the edge. Similar to [23], we computed a threshold based on the trade-off between local misclassifications and offloading costs. In [4], we showed that the HI offloading decision can be improved by using linear regression on the first two highest soft-max values.

However, the above works compute the optimal threshold offline using the training dataset and assume a constant offloading cost. This approach fails if the data distribution during the deployment/inference phase differs from the training data distribution or if the offloading cost varies over time. Modeling the latter aspect is important for offloading data using wireless communication, as the channel conditions vary over time. Motivated by this, the authors in [22] studied the HI learning problem (HIL) for dynamically learning an optimal threshold for the maximum soft-max value. They proposed an algorithm by extending the celebrated Hedge algorithm [13] for continuous expert problems. However, for the proposed algorithm, they provided a dataset-dependent regret bound $O(\sqrt{T \log 1/\lambda_{\min}})$, where $\lambda_{\min}$ is the minimum difference between any two distinct maximum soft-max values.

The authors in [5] extended the HI learning problem setting to multiple devices and studied maximizing average accuracy subject to average offloading cost constraint. To solve the HIL problem, they discretized the soft-max values and assumed that the cumulative loss of the optimal threshold and the minimum cumulative loss achieved from the discretized soft-max values is bounded by a small constant. The regret bound and computation complexity of their algorithm depend on the resolution at which the soft-max values are discretized. In contrast to [5, 22], we study a more general setting where the remote DL model is not 100% accurate, and the offloading costs need not be i.i.d. We propose two novel Hedge-based algorithms, Hedge-HI and Hedge-HI-Restart, which do not use discretization. The challenge in analyzing the proposed algorithms is that the number of experts grows with the number of experts. We present a new analysis for Hedge-HI and a novel high-probability regime analysis for the restart algorithm to obtain the regret bounds. Further, the computation complexity Hedge-HI-Restart is $O(\min\{t, \sqrt{T}\})$ which is significantly lower, asymptotically, compared to that of $O(t)$ of the HIL algorithm in [22].

To sum up, while the core idea of HI has been studied previously, highlighting the relevance and importance of the problem, our work differentiates itself with improved theoretical bounds and a novel algorithm with better performance. Additionally, our model is generalized to incorporate dynamic threshold learning, non-ideal remote DL models, and non-i.i.d. offloading costs.

2.2 Prediction with Expert Advice (PEA)

In the classical PEA setting [6], a learner has N experts to choose from, and the objective is to learn the expert with minimum cumulative loss over T rounds. PEA and a plethora of its variants are well-studied in the literature, and the Hedge algorithm [13] is known to provide an optimal regret bound $O(\sqrt{T \log N})$ for the standard setting. In [14], the authors introduced a PEA variant called the *branching experts*, where new experts may be revealed in each round. The cumulative loss of any new expert is either equal or close to the cumulative loss of one of the existing experts. They provided a $O(\sqrt{TN_T})$ regret algorithm. This setting was later studied for stochastic losses in [32]. However, in these works, N_T, the number of experts revealed till round T is assumed to be finite and is independent of T. We later show that the HIL problem is related to the branching experts problem. The key differences are that in HIL, 1) the losses are not revealed when a local DL inference is accepted, as the device does not know the ground truth, and 2) we do not assume that N_T is finite but may depend on T. Regarding the latter, in HIL N_T may be equal to T in the worst case. If N_T grows linearly with T, the regret for any algorithm for the HI learning problem is linear. This makes the problem more interesting as it raises the question: what are the minimal assumptions under which this problem will have sub-linear regret? We present one such assumption (Assumption 1) and propose Hedge-HI-Restart, which achieves sub-linear regret.

3 SYSTEM MODEL

We consider an edge intelligence system comprising an end device, or simply a device, and an edge server, or simply server. The device periodically senses/collects data samples and is tasked with performing multi-class classification of these data samples. The system is equipped with two DL models to perform this classification task. The device is embedded with the local DL model, and the server is equipped with a state-of-the-art large DL model, which we call remote DL. The remote DL is a state-of-the-art, higher classification accuracy model than the local DL. For each data sample, the device has the option of *accepting* the inference of the local DL model or enlisting the help of the server to use the remote DL model. We refer to this as *offloading* the task to the remote DL model.

***Multi-class Classification*.** Let $\mathcal{X}$ denote a feature space from which the data samples (e.g., images) are generated, and $\mathcal{Y} = \{1, \ldots, m\}$ denote label set with m classes. Given a pre-trained classifier $h : \mathcal{X} \mapsto \mathcal{Y}$, an *inference task* involves computing $h(x)$. Given (x, y), classification error on sample x is given by $\mathbb{1}[h(x) \neq y]$.

We consider a discrete-time system with a time index t. Let $x_t \in \mathcal{X}$ denote the data sample collected by the device in round t. In general, for each input, a DL model designed to carry out a multi-class classification task, outputs scores corresponding to each class using a *score function.* The higher the score for a class, the more confident the model is in classifying the input to that class. Let $f_i(x_t)$ denote the score for class i for input x_t. Let h_l denote the inference of the local DL model. Typically in a multi-class classification model, $h_l(x_t) = \text{argmax}_{i \in \mathcal{Y}} f_i(x_t)$. Similarly, let $h_r(x_t)$ denote the classification/inference provided by the server when x_t is offloaded to the remote DL.

We consider deterministic pre-trained DL models h_l and h_r, i.e., given the same data sample, they output the same scores. This is true for all traditional feed-forward networks, such as recurrent neural networks, convolutional neural networks, and transformers. During the DL training phase, a typical assumption is that the data samples are drawn i.i.d. using an unknown distribution from $\mathcal{X}$. We make the same assumption for the inference phase that x_t are drawn i.i.d. from $\mathcal{X}$ using an unknown distribution, potentially different from the training phase distribution.

***Confidence Metric*.** A confidence metric measures the confidence of h_l in classifying the data samples. The values of the score function can be used to obtain different confidence metrics. In general, a confidence metric is output by a deterministic function g, acting on $f(x_t) = [f_1(x_t)\ f_2(x_t)\ \cdots f_m(x_t)]$ for any x_t, given by $g : f(x_t) \mapsto \mathcal{Z}$, where $\mathcal{Z} \subseteq [0, 1]$. As discussed later, $\mathcal{Z}$ can be a discrete set if the DL model is quantized. Let $Z_t \in \mathcal{Z}$ denote the confidence metric computed using g for sample x_t. Since x_t are i.i.d. and h_l and g are deterministic functions, Z_t are i.i.d. We define the random vector $\mathbf{Z} = \{Z_1, \ldots, Z_T\}$. The confidence metric we use is the widely used maximum soft-max value [15], defined as follows. The soft-max value for class i is given by $\hat{g}_i(x_t) = e^{f_i(x_t)} / \sum_j e^{f_j(x_t)}$. The data sample x_t is classified into the class with maximum soft-max value, and thus, $Z_t = \max_i \hat{g}_i(x_t)$.

Multiple data samples may result in the same confidence metric value. In practice, this is a result of the precision at which the scoring function values are stored. For example, an 8-bit quantized ResNet8 outputs maximum soft-max values at a precision 0.0039 [3], resulting in a maximum of 256 distinct values. Let $N_t(\leq t)$ denote distinct confidence metric values observed till time t. We focus on the performance of the system in rounds 1 to T and derive the worst-case regret bound in terms of N_T. Note that the maximum number of experts revealed cannot exceed T. Therefore, we have $N_t \leq N_T \leq \min(|\mathcal{Z}|, T)$.

***Online Algorithm and Losses*.** In each round t, an online algorithm $\boldsymbol{\pi}$ decides to offload x_t or accept the inference $h_l(x_t)$ after observing Z_t. We use π_t to denote the decision to accept/offload in round t. Let $\Lambda_t \in (0, 1)$ denote the random loss incurred if the data sample collected in round t, i.e., x_t, is offloaded to the server for inference. We assume that the inference $h_l(x_t)$ for an offloaded sample x_t will be received within round t. The loss Λ_t may encompass the costs for transmission energy and the idle energy consumed by the transceiver until the inference is received and/or the monetary cost paid to the server. Our analysis applies to the general scenario where the Λ_ts may be correlated across time.

If in round t, the inference $h_l(x_t)$ is accepted, then the loss incurred Φ_t is given by

$$\Phi_t = \mathbb{1}(h_l(x_t) \neq h_r(x_t)), \tag{1}$$

where $(\cdot)$ is the indicator function. The intuition behind the definition of Φ_t is that the inference $h_r(x_t)$ provided by the remote DL is more likely to be correct than $h_l(x_t)$, i.e., the inference of the local DL. Since the device does not know the ground truth of the label of x_t, it does not know if $h_l(x_t)$ and/or $h_r(x_t)$ are the correct labels. However, by matching the output label to $h_r(x_t)$, the accuracy increases. We note that (1) generalizes the loss function considered in [5, 22], where the authors presented the regret-bound analysis for the case where $h_r(x_t)$ is the correct label for all x_t, i.e., the remote DL has 100% accuracy. Also, note that $h_r(x_t)$ will be unknown when the device accepts the local inference $h_l(x_t)$. Thus, Φ_t is *unknown* when $h_l(x_t)$ is accepted. This is one of the challenges we tackle in designing and analyzing the proposed algorithms.

Loss incurred under $\boldsymbol{\pi}$ in round t, denoted by $l_t(\pi_t)$, is given by

$$l_t(\pi_t) = \begin{cases} \Phi_t & \text{if accept } h_l(x_t), \\ \Lambda_t & \text{if offload } x_t. \end{cases} \tag{2}$$

A *fixed-threshold online algorithm* chooses a threshold $\theta \in \mathcal{Z}$ and in each round t, the data sample x_t is offloaded if $Z_t \leq \theta$, else the inference $h_t(x_t)$ is accepted. Thus, for fixed-threshold algorithms, π_t is fully characterized by θ, and therefore, with a slight abuse in notation, we use $l_t(\theta)$ as the loss in round t, given by

$$l_t(\theta) = \begin{cases} \Phi_t & \text{if } Z_t \geq \theta, \\ \Lambda_t & \text{if } Z_t < \theta. \end{cases} \tag{3}$$

Note that $l_t(\theta) \in [0, 1]$. Let $\mathbf{J} = \{(\Lambda_t, \Phi_t) : t = 1, \ldots, T\}$ denote the vector of random losses in T rounds.

***Regret Minimization*.** The decisions of $\boldsymbol{\pi}$ depend on past observations $\{l_\tau(\pi_\tau), \tau < t\}$ and $\{Z_\tau, \tau \leq t\}$. We are interested in characterizing the performance of the system in rounds 1 to T. Given the realizations of the random vector $\mathbf{Z}$, the *sample path cumulative loss*, or simply cumulative loss, $L_T(\boldsymbol{\pi})$ under algorithm $\boldsymbol{\pi}$ in T rounds is given by $L_T(\boldsymbol{\pi}) = \sum_{t=1}^{T} l_t(\pi_t)$. Note that both $l_t(\pi_t)$ and $L_T(\boldsymbol{\pi})$ are functions of the realizations of $\mathbf{Z}$. However, we omit the latter from the notation to keep the expressions concise. The sample path cumulative loss $L_T(\theta)$ of a fixed-threshold policy is given by

$$L_T(\theta) = \sum_{t=1}^{T} l_t(\theta). \tag{4}$$

Let θ^* denote the optimal-fixed threshold algorithm that minimizes the expected cumulative loss among fixed-threshold algorithms, given by

$$\theta^* = \operatorname*{argmin}_{\theta \in \mathcal{Z}} \mathbb{E}_{\mathbf{Z}}[L_T(\theta)].$$

Here, θ^* balances the trade-off between improved inference accuracy and increased loss due to offloading. We aim to develop online algorithms for learning the optimal threshold. To this end, we adopt the regret minimization approach. The expected regret $R_T(\boldsymbol{\pi})$ of algorithm $\boldsymbol{\pi}$ is defined as

$$R_T(\boldsymbol{\pi}) = \mathbb{E}^{\boldsymbol{\pi}}_{\mathbf{Z}}[L_T(\boldsymbol{\pi})] - \mathbb{E}_{\mathbf{Z}}[L_T(\theta^*)], \tag{5}$$

where the expectation $\mathbb{E}^{\pi}_{\mathbf{Z}}[\cdot]$ is with respect to the joint probability distribution induced by $\boldsymbol{\pi}$ and the random vector $\mathbf{Z}$. The regret analysis we present is valid for adversarial losses, i.e., the bounds are valid for any sequence of losses $\mathbf{J}$.

Our objective is to devise online algorithms that have sub-linear regret, i.e., $\frac{R_T(\boldsymbol{\pi})}{T}$ goes to zero as T goes to infinity. We refer to this regret minimization problem as the HI Learning (HIL) problem.

Notation: We use $\mathbb{1}(\cdot)$ for the indicator function, capital letters Z, Φ, Λ for random variables, bold letters $\mathbf{J}$ and $\mathbf{Z}$ for vectors, and calligraphic letters $\mathcal{Z}$ and $\mathcal{B}$ for sets. The functions $O(\cdot)$ and $\Omega(\cdot)$ are the standard big-O and big-Omega notations.

4 SAMPLE PATH ANALYSIS AND RELATION WITH BRANCHING EXPERTS

In this section, we prove a structural property of the sample path cumulative loss of fixed threshold policies, i.e., $L_T(\theta)$ as defined in (4). Specifically, we show that if the goal is to design a fixed threshold policy that minimizes the sample path cumulative loss by round T, choosing a threshold from the set of distinct confidence metric values observed up to round T suffices. Given this structural property, a special case of the HIL problem, where the cost Φ_t is revealed immediately after accepting the inference $h_l(x_t)$, is related to the branching expert problem [14], a variant of the classical prediction with expert advice problem. Further, we use the structural result as the basis for the proposed Hedge-HI algorithm and to obtain a lower bound for the regret for the HIL problem.

Recall that N_t is the number of distinct confidence metric values observed in t rounds. The distinct values of any realization of $Z_1, \ldots, Z_t$ can be written in an ordered sequence $z_{[1]}, z_{[2]}, \ldots, z_{[N_t]}$ where $z_{[1]} < z_{[2]} < \ldots < z_{[N_t]}$. Let $z_{[0]} = 0$. We define sets B_t^k, where $k \in \{0, 1, \ldots, N_t\}$, that partition $\mathcal{Z}$ as follows.

$$B_t^k = \{\theta \in \mathcal{Z} : z_{[k]} \le \theta < z_{[k+1]}\},\ \forall k \in \{0, 1, \ldots, N_t\}. \tag{6}$$

Lemma 1. *Let $L_t(\theta)$ denote the cumulative loss incurred for using a fixed threshold θ for t rounds. Then, $L_t(\theta)$ is equal for all $\theta \in B_t^k$.*

Proof. In the following analysis, we partition the t rounds into two sets, wherein the first partition contains the rounds in which the confidence metric is strictly less than $z_{[k+1]}$ and the second partition contains the rounds in which the confidence metric is at least $z_{[k+1]}$. Let $\mathbf{J}_t = \{(\Lambda_\tau, \Phi_\tau) : \tau = 1, \ldots, T\}$. By definition, $\theta \in B_t^k$ implies $z_{[k]} \le \theta < z_{[k+1]}$, and thus, for any $\theta \in B_t^k$,

$$\begin{aligned} L_t(\theta) = \sum_{\tau=1}^{t} l_t(\theta) &= \sum_{\tau: z_\tau < z_{[k+1]}} l_\tau(\theta) + \sum_{\tau: z_\tau \ge z_{[k+1]}} l_\tau(\theta) \\ &= \sum_{\tau: z_\tau \le z_{[k]}} l_\tau(\theta) + \sum_{\tau: z_\tau \ge z_{[k+1]}} \Phi_\tau \\ &= \sum_{\tau: z_\tau \le z_{[k]}} \Lambda_\tau + \sum_{\tau: z_\tau \ge z_{[k+1]}} \Phi_\tau. \end{aligned} \tag{7}$$

In the third step above, we have used the fact that there is no confidence metric value observed between $z_{[k]}$ and $z_{[k+1]}$. Otherwise, we will have additional partitions between $z_{[k]}$ and $z_{[k+1]}$ from definition (6). The result follows as (7) is independent of θ. □

An important consequence of Lemma 1 is that, in round t, it is sufficient for an online algorithm to choose a threshold from the set $\mathcal{B}_t = \{z_{[1]}, z_{[2]}, \ldots, z_{[N_t]}\}$, reducing the number of experts to choose from $|\mathcal{Z}|$ to N_t. In other words, the experts in round t are the distinct confidence metric values observed till time t. Thus, in the sequel, we use *a new expert is revealed* when a new (distinct) confidence metric value is observed in round t. Also, for any realization of $\{\mathbf{J}, \mathbf{Z}\}$, from Lemma 1 we have

$$\min_{\theta \in \mathcal{Z}} L_T(\theta) = \min_{\theta \in \mathcal{B}_T} L_T(\theta). \tag{8}$$

Corollary 1. *Let a distinct confidence metric value z_{t+1} is revealed in round $t+1$ and $z_{[k]} < z_{t+1} < z_{[k+1]}$ for some $k \in \{0, 1, \ldots, N_t\}$, then $L_t(z_{t+1}) = L_t(z_{[k]})$.*

Proof. In round t, z_{t+1} was not revealed, and from the hypothesis, we have $z_{t+1} \in B_t^k$. Therefore, from Lemma 1, choosing threshold $\theta = z_{t+1}$ for first t rounds results in the same cumulative loss as that of $\theta = z_{[k]}$. □

Note that Lemma 1 and Corollary 1 are true for any realization of $\{Z_1, \ldots, Z_t\}$ and any t. From these results, we conclude that, in the HIL problem, each new expert revealed in a round has a cumulative loss equal to the cumulative loss of an existing expert, which we refer to by *parent expert*. After the new expert is revealed, its cumulative loss in the subsequent rounds deviates from the cumulative loss of its parent expert. This setting falls under *branching experts with perfect clones* problem studied in [14], except that in the HIL problem, the additional challenge is that the loss Φ_t is unknown when the local inference is accepted. The authors in [14] provided a lower bound $\Omega(\sqrt{TN_T})$ for the branching experts problem. It also applies to the HIL problem and is stated in the lemma below.

Lemma 2 (Theorem 11 [14]). *Any online algorithm has regret $\Omega(\sqrt{TN_T})$ for the HIL problem.*

Therefore, HIL problem will have a sub-linear regret bound only if N_T is finite or it grows sub-linearly with T. Next, we present Hedge-HI and show that its regret has the same dependency on N_T to become sub-linear.

5 THE HEDGE-HI ALGORITHM

5.1 Algorithm Design

Hedge-HI builds upon the classical Hedge algorithm [13] that solves the PEA problem with finite set of N experts. Hedge uses a learning rate $\eta \ge 0$, assigns weight $w_t(i) = e^{-\eta L_{t-1}(i)}$ for each expert i based on the observed cumulative loss, and chooses expert i with probability $p_i = w_t(i)/W_t$, where $W_t = \sum_{\tau=1}^{t} w_\tau(i)$. For a suitable choice of η, Hedge has $O(\sqrt{T \ln N})$ regret. In HIL, as a consequence of Lemma 1 and Corollary 1, in any round t, it is sufficient to choose the distinct confidence metric values (experts) as the thresholds from the set $\mathcal{B}_t$. But, unlike the standard PEA problem, the expert set grows with t as new confidence metric values may be revealed in different rounds. Therefore, we extend the Hedge algorithm for the growing number of expert setting. In particular, we use a new total weight parameter $\hat{W}_t$ to account for the new expert revealed.

A second challenge is that the loss Φ_t is only revealed for the offloaded inference as $h_r(x_t)$ will be unknown locally. To address this situation, we employ the exploration-exploitation trade-off to obtain the ground truth and estimate the loss of our algorithm. This is achieved by offloading a subset of the samples where a decision to accept the local ML inference is made. The Hedge algorithm is further modified as follows: each round generates a Bernoulli random variable, ζ_t, with a probability of ϵ. When $\zeta_t = 1$, the sample is offloaded to the server, irrespective of the initial decision based on the chosen expert's threshold. These additional samples facilitate the device to estimate Φ_t accurately. Specifically, we estimate $l_t(\theta)$ using $\tilde{l}_t(\theta)$, given by

$$\tilde{l}_t(\theta) = \begin{cases} 0 & \text{if } Z_t \geq \theta \text{ and } \zeta_t = 0 \text{ (accept } h_l(x_t)) \\ \frac{\Phi_t}{\epsilon} & \text{if } Z_t \geq \theta \text{ and } \zeta_t = 1 \text{ (offload } x_t) \\ \Lambda_t & \text{if } Z_t < \theta \text{ (offload } x_t) \,. \end{cases}$$

For any Z_t and $\theta \in \mathcal{Z}$, we have,

$$\begin{aligned} \mathbb{E}_\zeta[\tilde{l}_t(\theta)] &= \mathbb{1}(Z_t \geq \theta)\mathbb{E}_\zeta[\mathbb{1}(\zeta_t = 1)]\frac{\Phi_t}{\epsilon} + \mathbb{1}(Z_t < \theta)\Lambda_t \\ &= \mathbb{1}(Z_t \geq \theta)\Phi_t + \mathbb{1}(Z_t < \theta)\Lambda_t = l_t(\theta). \end{aligned} \tag{9}$$

Thus, the proposed estimated loss $\tilde{l}_t(\theta)$ is an unbiased estimator of $l_t(\theta)$. We use $\tilde{l}_t(\theta)$ in our algorithm, and the weights update rule would be $w_{t+1}(\theta_t) = w_t(\theta_t)e^{-\eta\tilde{l}_t(\theta_t)}$. The steps of Hedge-HI are presented in Algorithm 1, and we denote it by $\boldsymbol{\pi}^{\mathrm{H}}$.

5.2 Regret Analysis

The expected cumulative loss under $\boldsymbol{\pi}^H$ involves joint expectation involving ζ_t and the probability distributions p_t used in line 10 of Algorithm 1. Therefore, we have

$$\mathbb{E}^{\boldsymbol{\pi}^{\mathrm{H}}}[L_T(\boldsymbol{\pi}^{\mathrm{H}})] = \sum_{t=1}^{T}\sum_{\theta\in\mathcal{B}_t}\frac{w_t(\theta)}{\hat{W}_t}\mathbb{E}_\zeta[l_t(\theta)]. \tag{10}$$

Let $\mathbb{E}^{\boldsymbol{\pi}^{\mathrm{H}}}[\tilde{L}_T(\boldsymbol{\pi}^{\mathrm{H}})]$ denote the expected pseudo cumulative loss based on the estimated losses $\tilde{l}_t(\pi_t^{\mathrm{H}})$, given by

$$\mathbb{E}^{\boldsymbol{\pi}^{\mathrm{H}}}[\tilde{L}_T(\boldsymbol{\pi}^{\mathrm{H}})] = \sum_{t=1}^{T}\sum_{\theta\in\mathcal{B}_t}\frac{w_t(\theta)}{\hat{W}_t}\mathbb{E}_\zeta[\tilde{l}_t(\theta)]. \tag{11}$$

Lemma 3. *$\mathbb{E}^{\boldsymbol{\pi}^H}[L_T(\boldsymbol{\pi}^H)] - \mathbb{E}^{\boldsymbol{\pi}^H}[\tilde{L}_T(\boldsymbol{\pi}^H)] \leq \epsilon T$.*

Proof. If $\boldsymbol{\pi}^{\mathrm{H}}$ chooses θ in round t, the loss is given by

$$\begin{aligned} &l_t(\theta) = \mathbb{1}(Z_t \geq \theta)\mathbb{1}(\zeta_t = 0)\Phi_t + (1 - \mathbb{1}(Z_t \geq \theta)\mathbb{1}(\zeta_t = 0))\Lambda_t \\ &\Rightarrow \mathbb{E}_\zeta[l_t(\theta)] = \mathbb{1}(Z_t \geq \theta)(1-\epsilon)\Phi_t + (1-\mathbb{1}(Z_t \geq \theta)(1-\epsilon))\Lambda_t. \end{aligned} \tag{12}$$

From (9) and (12), we obtain

$$\mathbb{E}_\zeta[l_t(\pi_t^{\mathrm{H}})] - \mathbb{E}_\zeta[\tilde{l}_t(\theta)] = -\epsilon\Phi_t + \mathbb{1}(Z_t \geq \theta)\epsilon\Lambda_t \leq \epsilon. \tag{13}$$

The result follows by using (13), (10), and (11). □

Algorithm 1: Hedge-HI

```
1: Initialize: select any $\theta_0 \in \mathcal{Z}$. Set $\mathcal{B}_0 = \{\theta_0\}$, $n_0 = 0$ ,
   $w_1(\theta_0) = 1$, and $W_1 = 1$.
2: for each round $t = 1, 2, \ldots, T$ do
3:     if distinct confidence metric value $Z_t$ (new expert) is
       revealed then
4:         $n_t = n_{t-1} + 1$ and $\mathcal{B}_t = \mathcal{B}_{t-1} \cup \{Z_t\}$
5:         Initialize $\tilde{L}_t(Z_t)$ to the cumulative loss of $Z_t$'s parent.
6:         Compute new weight $w_t(Z_t) = e^{-\eta\tilde{L}_t(Z_t)}$, and
           $\hat{W}_t = W_t + w_t(Z_t)$
7:     else
8:         $\mathcal{B}_t = \mathcal{B}_{t-1}$ and $\hat{W}_t = W_t$
9:     end if
10:    Choose $\theta \in \mathcal{B}_t$ with probability $p_t = \frac{w_t(\theta)}{\hat{W}_t}$.
11:    if $Z_t <$ chosen threshold then
12:        Offload $x_t$, receive loss $\tilde{l}_t = \Lambda_t$.
13:    else
14:        Generate a Bernoulli random variable $\zeta_t$ with
           $\mathbb{P}(\zeta_t = 1) = \epsilon$.
15:        if $\zeta_t = 1$ then
16:            Offload $x_t$, receive loss $\tilde{l}_t = \frac{\Phi_t}{\epsilon}$.
17:        else
18:            Accept the inference $h_l(x_t)$, assign loss $\tilde{l}_t = 0$.
19:        end if
20:    end if
21:    Compute $\tilde{l}_t(\theta), \forall\theta \in \mathcal{B}_t$.
22:    Update weights $w_{t+1}(\theta) = e^{-\eta\tilde{l}_t(\theta)}w_t(\theta), \forall\theta \in \mathcal{B}_t$.
23:    Cumulative weight $W_{t+1} = \sum_{\theta\in\mathcal{B}_t} w_t(\theta)$.
24: end for
```

Theorem 1. *The regret of Hedge-HI satisfies $R_T(\boldsymbol{\pi}^H) \leq g(\epsilon, \eta)$, where*

$$g(\epsilon, \eta) = \left(\epsilon + \frac{\eta}{2\epsilon}\right)T + \frac{\mathbb{E}_Z[N_T]}{\eta}\log 2. \tag{14}$$

Proof. Let $\tilde{L}_T(\theta)$ denote the sample-path pseudo cumulative based on the estimated losses for using θ in T rounds. First, we obtain a lower bound for $\log\frac{W_{T+1}}{W_1}$:

$$\begin{aligned} \log\frac{W_{T+1}}{W_1} = \log W_{T+1} &= \log\sum_{\theta\in\mathcal{B}_T} e^{-\eta\sum_{i=1}^{T}\tilde{l}_i(\theta)} \\ &\geq \log\max_{\theta\in\mathcal{B}_T} e^{-\eta\tilde{L}_T(\theta)} \\ &\geq \max_{\theta\in\mathcal{B}_T}\log e^{-\eta\tilde{L}_T(\theta)} = -\min_{\theta\in\mathcal{B}_T}\eta\tilde{L}_T(\theta). \end{aligned}$$

Taking expectation with respect to the Bernoulli distribution of ζ on both sides, leveraging the property that the expectation of the minimum is bounded above by the minimum of the expectations and (9), we obtain

$$\mathbb{E}_\zeta\left[\log\frac{W_{T+1}}{W_1}\right] \geq -\min_{\theta\in\mathcal{B}_T}\eta\mathbb{E}_\zeta[\tilde{L}_T(\theta)] = -\min_{\theta\in\mathcal{B}_T}\eta L_T(\theta). \tag{15}$$

Next, we obtain an upper bound for $\log \frac{W_{T+1}}{W_1}$.

$$\log \frac{W_{T+1}}{W_1} = \log \prod_{t=1}^{T} \frac{W_{t+1}}{W_t} = \sum_{t=1}^{T} \log \frac{W_{t+1}}{\hat{W}_t} + \sum_{t=1}^{T} \log \frac{\hat{W}_t}{W_t}. \quad (16)$$

We note that $\hat{W}_t$ (defined in line 6, Algorithm 1) normalizes the weights w_t. Therefore, we have

$$\begin{aligned}\log \frac{W_{t+1}}{\hat{W}_t} &= \log \sum_{\theta \in \mathcal{B}_t} \frac{w_t(\theta)}{\hat{W}_t} e^{-\eta \tilde{l}_t(\theta)} \\ &\leq \log \sum_{\theta \in \mathcal{B}_t} \frac{w_t(\theta)}{\hat{W}_t} \left(1 - \eta \tilde{l}_t(\theta) + \frac{(\eta \tilde{l}_t(\theta))^2}{2}\right) \\ &= \log \left[1 + \sum_{\theta \in \mathcal{B}_t} \frac{w_t(\theta)}{\hat{W}_t} \left(-\eta \tilde{l}_t(\theta) + \frac{(\eta \tilde{l}_t(\theta))^2}{2}\right)\right] \\ &\leq \sum_{\theta \in \mathcal{B}_t} \frac{w_t(\theta)}{\hat{W}_t} \left(-\eta \tilde{l}_t(\theta) + \frac{(\eta \tilde{l}_t(\theta))^2}{2}\right) \\ &\leq \sum_{\theta \in \mathcal{B}_t} \frac{w_t(\theta)}{\hat{W}_t} \left(-\eta \tilde{l}_t(\theta) + \frac{\eta^2 \tilde{l}_t(\theta)}{2\epsilon}\right). \quad (17)\end{aligned}$$

In the second step above, we have used $e^{-x} \leq 1 - x + x^2/2$. It is easy to show that the function $-\eta \tilde{l}_t(\theta) + (\eta \tilde{l}_t(\theta))^2/2 \geq -0.5$ for any value of $\eta \tilde{l}_t(\theta)$, and thus, in the third step we have used $\log(1 + x) \leq x$. In the last step, we have used $\tilde{l}(\theta) \in [0, 1/\epsilon]$.

Let E_t denote the event of a new expert arrival in round t. If $E_t = 1$, then in Algorithm 1, line 6, we have $\hat{W}_t = W_t + w_t(Z_t)$, else $\hat{W}_t = W_t$. Thus,

$$\frac{\hat{W}_t}{W_t} = \begin{cases} 1 & \text{if } E_t = 0, \\ 1 + \frac{w_t(Z_t)}{W_t} & \text{if } E_t = 1. \end{cases} \quad (18)$$

Therefore, from (16), we have

$$\begin{aligned}\log \frac{W_{T+1}}{W_1} &= \sum_{t=1}^{T} \log \frac{W_{t+1}}{\hat{W}_t} + \sum_{t=1}^{T} \log \frac{\hat{W}_t}{W_t} \\ &\leq \sum_{t=1}^{T} \log \frac{W_{t+1}}{\hat{W}_t} + \sum_{t=1}^{T} \mathbb{1}(E_t = 1) \log\left(1 + \frac{w_t(Z_t)}{W_t}\right) \quad (19) \\ &\leq \sum_{t=1}^{T} \log \frac{W_{t+1}}{\hat{W}_t} + \sum_{t=1}^{T} \mathbb{1}(E_t = 1) \log\left(1 + \frac{e^{-\eta \tilde{L}_t(Z_t)}}{W_t}\right) \\ &\leq \sum_{t=1}^{T} \log \frac{W_{t+1}}{\hat{W}_t} + N_T \log 2 \\ &\leq \sum_{t=1}^{T} \sum_{\theta \in \mathcal{B}_t} \frac{w_t(\theta)}{\hat{W}_t} \left(-\eta \tilde{l}_t(\theta) + \frac{\eta^2 \tilde{l}_t(\theta)}{2\epsilon}\right) + N_T \log 2. \quad (20)\end{aligned}$$

We have used (18) in the second step; $e^{-\eta \tilde{L}_t(Z_t)} << W_t$ and the number of events $E_t = 1$ till time T is N_T in the fourth step; and (17) in the last step. Now, taking expectation with respect to the Bernoulli distribution of ζ on both sides of (20), we obtain

$$\begin{aligned}\mathbb{E}_\zeta \left[\log \frac{W_{T+1}}{W_1}\right] &\leq -\eta \sum_{t=1}^{T} \sum_{\theta \in \mathcal{B}_t} \frac{w_t}{\hat{W}_t} \mathbb{E}_\zeta [\tilde{l}_t(\theta)] \\ &\quad + \sum_{t=1}^{T} \sum_{\theta \in \mathcal{B}_t} \frac{w_t}{\hat{W}_t} \frac{\eta^2 \mathbb{E}_\zeta [\tilde{l}_t(\theta)]}{2\epsilon} + N_T \log 2 \\ &\leq -\eta \mathbb{E}^{\pi^H} [\tilde{L}_T(\theta)] + \frac{\eta^2}{2\epsilon} T + N_T \log 2 \quad (21)\end{aligned}$$

In the second step above, we have used (11) for the first term, and $\mathbb{E}_\zeta [\tilde{l}_t(\theta)] = l_t(\theta) \leq 1$, from (9), in the second term. In the last step, we have used Lemma 3.

The result follows from (15) and (21) and taking expectation with respect to $\mathbf{Z}$. □

The bound in Theorem 1 is valid without taking expectation with respect to expert arrivals $\mathbf{Z}$. Thus, the regret bound is valid for adversarial losses and any sequence of expert arrivals. The bound neatly captures the effect of ϵ on the regret. Note that the term ϵT is a direct consequence of offloading the sample at round t, when $\zeta = 1$. We minimize this bound and determine the parameters that make the bound sublinear in T.

Lemma 4. *The function $g(\epsilon, \eta)$ defined in (14) has the global minimum at (ϵ^*, η^*), where $\epsilon^* = \sqrt{\eta/2}$ and $\eta^* = \left(2(\mathbb{E}_\mathbf{Z}[N_T])^2 \log^2 2/T^2\right)^{\frac{1}{3}}$.*

Proof. The proof is deferred to [1] due to space limitation. □

The following corollary is a direct consequence of Theorem 1 and Lemma 4.

Corollary 2. *Hedge-HI achieves the following regret bound for $\epsilon^* = \sqrt{\eta/2}$ and $\eta^* = \left(2(\mathbb{E}_\mathbf{Z}[N_T])^2 \log^2 2/T^2\right)^{\frac{1}{3}}$:*

$$R_T(\boldsymbol{\pi}^H) \leq 3T^{\frac{2}{3}} \left(\mathbb{E}_\mathbf{Z}[N_T] \log 2/2\right)^{\frac{1}{3}}. \quad (22)$$

From Corollary 2, we infer that if $\mathbb{E}_\mathbf{Z}[N_T]$ is finite or sub-linear in T, then Hedge-HI has sub-linear regret. It would be interesting to find sub-linear regret bounds even if $\mathbb{E}_\mathbf{Z}[N_T]$ grows linearly with T. We show this is possible under a mild assumption in the next section. The computational complexity of Hedge-HI in round t is dominated by lines 10 and 22 in Algorithm 1 involving an iteration over the set $\mathcal{B}_t$. Thus, its computational complexity in round t is $O(t)$ since $|\mathcal{B}_t| = N_t \leq t$.

6 HEDGE-HI WITH SINGLE RESTART

In this section, we propose Hedge-HI-Restart. We prove that, with high probability, it has a sub-linear regret, even if $\mathbb{E}_\mathbf{Z}[N_T] = O(T)$, under the following assumption.

Assumption 1. *In any round t, a confidence metric value equal to an optimal threshold θ^* is revealed with fixed non-zero probability, i.e., $\mathbb{P}(Z_t = \theta^*) = \nu$, for some $\nu \in (0, 1]$.*

The assumption is true, for example, in scenarios where $|\mathcal{Z}|$ is large albeit a finite value, and the Z_t values are observed uniformly at random as a result of uniformly sampling x_t from $\mathcal{X}$. Let $q_\tau = \mathbb{P}(\theta^* \in \mathcal{B}_\tau)$ denote the probability that the best expert θ^* is revealed in the first τ slots. Recall that Z_t are i.i.d. We have

$$q_\tau = 1 - [1 - \nu]^\tau = 1 - \left[1 - \frac{\tau\nu}{\tau}\right]^\tau \geq 1 - e^{-\tau\nu}. \quad (23)$$

In the last step above, we have used $(1 - x/n)^n < e^{-x}$. Note that as τ increases, q_τ, the probability that θ^* is revealed, increases. The idea behind Hedge-HI-Restart is to wait for a sufficient number of rounds τ such that θ^* is revealed with a high enough probability and then freeze the set of experts to be utilized in Hedge-HI.

Algorithm 2: Hedge-HI-Restart (with parameter τ)

```
1: For $t = 1, \ldots, \tau$, use Hedge-HI
2: For $t > \tau$, use Hedge-HI only using the experts from $\mathcal{B}_\tau$ and
   resetting the weights $w_{\tau+1}(\theta) = 1$, for all $\theta \in \mathcal{B}_\tau$.
```

We present Hedge-HI-Restart with parameter τ in Algorithm 2 and use $\boldsymbol{\pi}^{\mathrm{HR}}$ to denote it. We note that $\boldsymbol{\pi}^{\mathrm{HR}}$ uses Hedge-HI for the first τ slots, for some $\tau > 0$. At the start of round $\tau + 1$, it resets the weights, and for all $t \geq \tau + 1$, it uses Hedge-HI with the experts revealed till round τ, i.e., without adding the new experts revealed after round τ. Later, as part of the regret analysis, we will compute the value of τ that results in a sub-linear regret bound.

Theorem 2. *The regret $R_T(\boldsymbol{\pi}^{HR})$ of the Hedge-HI-Restart satisfies,*

$$R_T(\boldsymbol{\pi}^{HR}) \leq \epsilon T + \frac{\eta T}{2\epsilon} + \frac{(2e^{\eta\tau} + 1)\log(\mathbb{E}_Z[N_\tau])}{\eta}, \qquad w.p.\ q_\tau.$$

Proof. Proof of the theorem is given in Appendix A. □

Note that the bound in Theorem 2 is valid for adversarial losses and stochastic expert arrivals. From (23), for the probability q_t to approach 1, we need to choose $\tau = \Omega(T^\alpha)$, for some $\alpha > 0$. However, from Theorem 2, we observe that the third term in the regret bound increases exponentially in τ. Therefore, to obtain a sublinear regret bound, we impose an upper bound on τ, specifically, $\tau \leq \frac{c}{\eta}$, for some constant $c > 0$. We present the sublinear regret bound in the following corollary (proof is deferred to [1]).

Corollary 3. *For any constant $c > 0$, let $c_1 = \sqrt{2}(2^c + 1)$ and choose $\eta = (c_1 \log(\mathbb{E}_Z[N_T])/T)^{\frac{2}{3}}$ and $\epsilon = \sqrt{\eta/2}$. Then, for any $\alpha \in (0, 1)$, Hedge-HI-Restart has $O\left(T^{\frac{2}{3}}(\log(\mathbb{E}_Z[N_T]))^{\frac{1}{3}}\right)$ regret, with probability at least $1 - e^{-\nu\tau}$, where*

$$\tau = c\,(T/c_1\mathbb{E}[\log N_T])^{\frac{2\alpha}{3}}\,. \tag{24}$$

From Corollary 3, we observe that the regret bound of Hedge-HI-Restart scales with $(\log(\mathbb{E}_Z[N_T]))^{\frac{1}{3}}$ which is a significant improvement over the regret bound of Hedge-HI (cf. Corollary 2). Further, the regret bound is sub-linear even if $N_T = T$, i.e., a new confidence metric value is revealed in each round. This is possible when the confidence metric space $\mathcal{Z}$ is continuous.

Similar to Hedge-HI, the computational complexity of Hedge-HI-Restart is given by $O(t)$. However, for $t > \tau$, we have $t \leq \tau$ as no new experts are added to $\mathcal{B}_\tau$. Therefore, the computation complexity of Hedge-HI-Restart is $O(\min(t, \tau))$. Contrast this with the computational complexity of the $O(t)$ per inference for the HI algorithm in [22]. One may choose small τ by choosing a small value for α in (24). However, this reduces the rate at which probability in (23) goes to 1. In our simulations, we choose $\alpha = 0.75$ for which τ is in the order of $\sqrt{T/\log(\mathbb{E}_Z[N_T])}$ and demonstrate that Hedge-HI-Restart has significantly lower runtime than the alternatives.

Remark: Though Assumption 1 is key to proving the regret bound for Hedge-HI-Restart, we study its performance without the assumption in simulations.

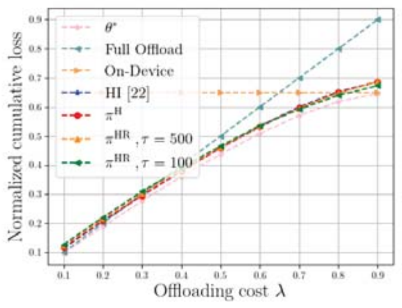
7 NUMERICAL RESULTS

In this section, we evaluate the performance and the runtime of Hedge-HI and Hedge-HI-Restart , denoted by $\boldsymbol{\pi}^{\mathrm{H}}$ and $\boldsymbol{\pi}^{\mathrm{HR}}$, against four algorithms: 1) On-Device, which accepts all the local DL inferences; 2) Full Offload, which offloads all the local DL inferences; 3) θ^*, the optimal fixed-threshold algorithm; 4) the HIL algorithm [22]. We conducted simulations for $T = 10000$ rounds, setting ϵ and η to the values as discussed in Lemma 4 and Corollary 3. We average the results over 100 runs of each simulation that generates 100 different realizations of input sample sequences.

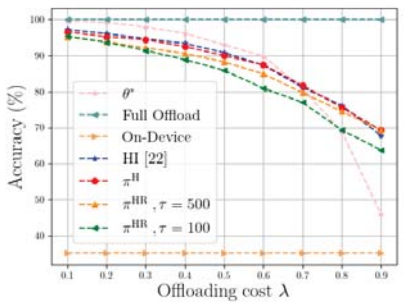
Dataset, DL model, and losses: We considered the widely-studied image classification application and used the dataset ImageNet, which has 1000 classes and 50000 images in its validation dataset. We also used other well-known datasets Imagenette and Imagewoof, subsets of ImageNet, containing 10 classes and around 400 images per class each. However, results for these datasets follow similar trends shown below and can be found in the technical report [1]. We classify the images using an 8-bit quantized MobileNet tflite model with a width parameter of 0.25 [30]. The accuracy of this model for classifying the validation dataset of ImageNet dataset is 35%, which is the accuracy for On-Device. The model outputs 8-bit scores, thus $|\mathcal{Z}| = 256$, and the number of distinct soft-max values in T rounds $N_T \leq 256$. The model is 500 KB, and we deploy it on a Raspberry Pi with 4 CPU cores, 1.5 GHz frequency, and 8 GB RAM. We consider that the remote DL model has 100% accuracy. As a result, Full Offload has 100 % accuracy. When the local DL inference is accepted, the cost Φ_t equals 1 if the inference is incorrect and 0 otherwise. The loss for offloading Λ_t is a constant cost equal to λ across slots. We observe similar trends for Λ_t generated from a uniform distribution and are not presented due to redundancy.

Performance comparison. In Figure 2, we compare the performance of the algorithms in terms of normalized cumulative loss, accuracy, and the number of offloaded images. We present $\boldsymbol{\pi}^{\mathrm{HR}}$ with two restart times, $\tau = 100, 500$, obtained by substituting $T = 10000$ and $N_T = 256$ in $\sqrt{6T/\log(\mathbb{E}_Z[N_T])}$ and $\sqrt{140T/\log(\mathbb{E}_Z[N_T])}$, respectively, chosen according to the discussion at the end of Section 6. Observe that in terms of cumulative loss, the HI-based approaches outperform Full Offload and On-Device for almost the entire range of values of λ. Further, the cumulative loss of $\boldsymbol{\pi}^{\mathrm{H}}$ and $\boldsymbol{\pi}^{\mathrm{HR}}$ (for $\tau = 100$ and $\tau = 500$) is comparable to that of the optimal fixed threshold policy even though these policies are online policies and do not have information about the correctness of the inference of the local-DL for samples that are not offloaded. Though the proposed algorithms and HI from [22] have similar performance, we will show shortly that our algorithms, grounded in the structural property discussed in Section 4, demonstrate enhanced efficiency, particularly when considering the runtime per inference.

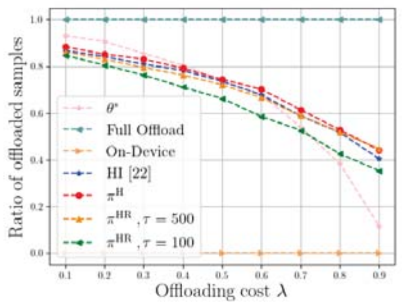
Runtime comparison: Figure 3(a) presents the runtime comparison between the algorithms for the ImageNet dataset. As expected, since the expert set can grow only up to round τ in $\boldsymbol{\pi}^{\mathrm{HR}}$ algorithms, the time per inference increases up to round τ and stabilizes thereafter. We note that the time per inference for HI in [22] and $\boldsymbol{\pi}^{\mathrm{H}}$ also saturates once all the 256 distinct maximum soft-max values are revealed. Observe that $\boldsymbol{\pi}^{\mathrm{HR}}$ has a runtime half that of HI [22]. To



(a) Normalized cumulative loss

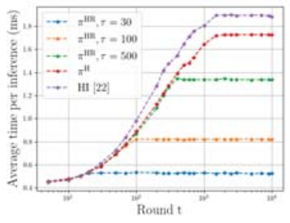


(b) Accuracy of classification

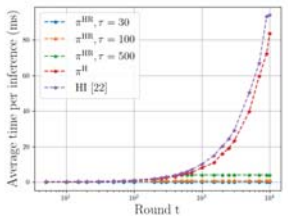


(c) Ratio of offloaded samples

Figure 2: Performance comparison between different algorithms for ImageNet dataset classification for varying λ.



(a) ImageNet dataset.



(b) Synthetic data.

Figure 3: Comparison of average time per inference for varying t under different algorithms.

further study the runtime performance of the algorithms, we synthetically generated a new confidence metric value in each round and presented the runtimes in Figure 3(b). As expected, the runtime per inference for HI in [22] and $\boldsymbol{\pi}^{\mathrm{H}}$ keeps growing with t, while that of $\boldsymbol{\pi}^{\mathrm{HR}}$ saturates after round τ, resulting in a notable difference of two orders of magnitude. Note that this difference is bound to increase if we run the algorithms on resource-constrained IoT devices instead of Raspberry Pi. We thus conclude that $\boldsymbol{\pi}^{\mathrm{HR}}$ has a significantly lower response time and is more conducive to implementing HIL on resource-constrained devices than other HI-based algorithms while having comparable loss performance.

8 CONCLUSION

HI is a promising approach for distributed inference which selectively offloads the data samples for which the local DL inference is likely incorrect. Our objective is to learn an optimal threshold for the confidence metric that minimizes the losses for accepting incorrect local DL inference and the losses for offloading. We proposed two novel algorithms Hedge-HI and Hedge-HI-Restart and proved sub-linear regret bounds for both algorithms. The asymptotic runtime of Hedge-HI-Restart is $O(\sqrt{T})$, as compared to $O(T)$ for its alternatives, which is prohibitive for resource-constrained devices. We present numerical results using the ImageNet dataset; we found that Hedge-HI-Restart achieves a cumulative loss close to the optimal threshold policy while providing an order of magnitude lower runtime than the alternatives.

ACKNOWLEDGEMENT

This work is supported in part by the European Union through MSCA-PF project "DIME: Distributed Inference for Energy-efficient Monitoring at the Network Edge" under Grant 101062011, and in part by SERB grant on Leveraging Edge Resources for Service Hosting and an SCPP grant from IIT Bombay, India.

REFERENCES

[1] Ghina Al-Atat, Puranjay Datta, Sharayu Moharir, and Jaya Prakash Champati. 2024. Regret Bounds for Online Learning for Hierarchical Inference. https://drive.google.com/file/d/1j0_cY-oCaTsLh_i9yFDbgHjScBKTmvFF/view.
[2] Ghina Al-Atat, Andrea Fresa, Adarsh Behera, Vishnu Moothedath, James Gross, and Jaya Champati. 2023. The case for hierarchical deep learning inference at the network edge. In *Proc. Workshop on Networked AI Systems, MobiSyS*. 1–6.
[3] Colby Banbury et al. 2021. MLPerf Tiny Benchmark. *Proc. NeurIPS (Systems Track on Datasets and Benchmarks)* (2021).
[4] Adarsh Prasad Behera, Roberto Morabito, Joerg Widmer, and Jaya Prakash Champati. 2023. Improved Decision Module Selection for Hierarchical Inference in Resource-Constrained Edge Devices. In *Proc. ACM Mobicom*. 1–3.
[5] Hasan Burhan Beytur, Ahmet Gunhan Aydin, Gustavo de Veciana, and Haris Vikalo. 2024. Optimization of Offloading Policies for Accuracy-Delay Tradeoffs in Hierarchical Inference. In *Proc. IEEE INFOCOM*. 1989–1998.
[6] Nicolo Cesa-Bianchi and Gábor Lugosi. 2006. *Prediction, learning, and games*. Cambridge university press.
[7] Yanjiao Chen, Baolin Zheng, Zihan Zhang, Qian Wang, Chao Shen, and Qian Zhang. 2020. Deep learning on mobile and embedded devices: State-of-the-art, challenges, and future directions. *Comput. Surveys* 53, 4 (2020), 1–37.
[8] Gemma Team (Google DeepMind). 2024. Gemma: Open Models Based on Gemini Research and Technology. arXiv:2403.08295 [cs.CL]
[9] Lei Deng, Guoqi Li, Song Han, Luping Shi, and Yuan Xie. 2020. Model compression and hardware acceleration for neural networks: A comprehensive survey. *Proc. IEEE* 108, 4 (2020), 485–532.
[10] Chongwu Dong, Sheng Hu, Xi Chen, and Wushao Wen. 2021. Joint optimization with DNN partitioning and resource allocation in mobile edge computing. *IEEE Transactions on Network and Service Management* 18, 4 (2021), 3973–3986.
[11] Maryam Ebrahimi, Alexandre da Silva Veith, Moshe Gabel, and Eyal de Lara. 2022. Combining DNN partitioning and early exit. In *Proceedings of the 5th International Workshop on Edge Systems, Analytics and Networking*. 25–30.
[12] Andrea Fresa and Jaya Prakash Champati. 2023. Offloading Algorithms for Maximizing Inference Accuracy on Edge Device in an Edge Intelligence System. *IEEE Transactions on Parallel and Distributed Systems* 34, 7 (2023), 2025–2039.
[13] Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences* 55, 1 (1997), 119–139.
[14] Eyal Gofer, Nicolo Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. 2013. Regret minimization for branching experts. In *Proc. COLT*. 618–638.
[15] Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv:1610.02136* (2016).
[16] Chenghao Hu and Baochun Li. 2022. Distributed Inference with Deep Learning Models across Heterogeneous Edge Devices. In *Proc. IEEE INFOCOM*. 330–339.
[17] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. 2017. Neurosurgeon: Collaborative intelligence between

the cloud and mobile edge. *ACM SIGARCH Comp. Arch. News* 45, 1 (2017).

[18] Nicholas D Lane, Sourav Bhattacharya, Akhil Mathur, Petko Georgiev, Claudio Forlivesi, and Fahim Kawsar. 2017. Squeezing deep learning into mobile and embedded devices. *IEEE Pervasive Computing* 16, 3 (2017), 82–88.

[19] En Li, Liekang Zeng, Zhi Zhou, and Xu Chen. 2020. Edge AI: On-Demand Accelerating Deep Neural Network Inference via Edge Computing. *IEEE Transactions on Wireless Communications* 19, 1 (2020), 447–457.

[20] Pavel Mach and Zdenek Becvar. 2017. Mobile Edge Computing: A Survey on Architecture and Computation Offloading. *IEEE Communications Surveys Tutorials* 19, 3 (2017), 1628–1656.

[21] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, and Mohsen Guizani. 2018. Deep learning for IoT big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials* 20, 4 (2018), 2923–2960.

[22] Vishnu Narayanan Moothedath, Jaya Prakash Champati, and James Gross. 2024. Getting the Best Out of Both Worlds: Algorithms for Hierarchical Inference at the Edge. *IEEE Tran. on Machine Learning in Comm. and Netw.* 2 (2024), 280–297.

[23] Ivana Nikoloska and Nikola Zlatanov. 2021. Data Selection Scheme for Energy Efficient Supervised Learning at IoT Nodes. *IEEE Comm. Let.* 25, 3 (2021), 859–863.

[24] Samuel S. Ogden and Tian Guo. 2020. MDINFERENCE: Balancing Inference Accuracy and Latency for Mobile Applications. In *Proc. IEEE IC2E*. 28–39.

[25] Ramon Sanchez-Iborra and Antonio F Skarmeta. 2020. Tinyml-enabled frugal smart objects: Challenges and opportunities. *IEEE Circuits and Systems Magazine* 20, 3 (2020), 4–18.

[26] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proc. IEEE CVPR*. 4510–4520.

[27] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Caceres, and Nigel Davies. 2009. The case for vm-based cloudlets in mobile computing. *IEEE pervasive Computing* 8, 4 (2009), 14–23.

[28] Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proc. ICML*. PMLR, 6105–6114.

[29] Surat Teerapittayanon, Bradley McDanel, and H.T. Kung. 2016. BranchyNet: Fast inference via early exiting from deep neural networks. In *Proc. ICPR*. 2464–2469.

[30] TensorFlow. 2024. TFLite Models. https://www.tensorflow.org/lite/models.

[31] Junjue et al. Wang. 2018. Bandwidth-efficient live video analytics for drones via edge computing. In *Proc. ACM/IEEE SEC*. 159–173.

[32] Yi-Shan Wu, Yi-Te Hong, and Chi-Jen Lu. 2021. Lifelong Learning with Branching Experts. In *Asian Conference on Machine Learning*. PMLR, 1161–1175.

[33] Dianlei Xu, Tong Li, Yong Li, Xiang Su, Sasu Tarkoma, Tao Jiang, Jon Crowcroft, and Pan Hui. 2021. Edge intelligence: Empowering intelligence to the edge of network. *Proc. IEEE* 109, 11 (2021), 1778–1837.

A PROOF OF THEOREM 2

For $\hat{t} > t$, let $L_{t,\hat{t}}(\theta)$ and $L_{t,\hat{t}}(\boldsymbol{\pi}^{\text{HR}})$ denote the cumulative loss in the slots $\{t+1, \dots, \hat{t}\}$ under fixed-threshold θ and the Hedge-HI-Restart, respectively, and are given by

$$L_{t,\hat{t}}(\theta) = \sum_{r=t+1}^{\hat{t}} l_r(\theta), \quad \text{and} \quad L_{t,\hat{t}}(\boldsymbol{\pi}^{\text{HR}}) = \sum_{r=t+1}^{\hat{t}} l_r(\pi_t^{\text{HR}}).$$

$L_T(\theta^*)$ is the minimum sample path cumulative loss that can be achieved for a given realization of $\{\mathbf{J}, \mathbf{Z}\}$. Since $\theta^* \in \mathcal{B}_\tau$, we have

$$\begin{aligned} L_T(\theta^*) &= \min_{\theta \in \mathcal{B}_\tau} L_T(\theta) = L_\tau(\theta^*) + L_{\tau,T}(\theta^*) \\ &= \min_{\theta \in \mathcal{B}_\tau} L_\tau(\theta) + \min_{\theta \in \mathcal{B}_\tau} L_{\tau,T}(\theta), \text{ w.p. } q_\tau. \quad (25) \end{aligned}$$

The following lemma proves a new bound for Hedge-HI that is equally valid for Hedge-HI-Restart.

Lemma 5. *Given τ, under Hedge-HI (or Hedge-HI-Restart), we have*

$$\mathbb{E}^{\boldsymbol{\pi}^H}[L_\tau(\boldsymbol{\pi}^H)] - \min_{\theta \in \mathcal{B}_\tau} L_\tau(\theta) \le \epsilon\tau + \frac{\eta\tau}{2\epsilon} + \frac{2e^{\eta\tau}\log N_\tau}{\eta}.$$

Proof. The proof structure mirrors that of our proof in Theorem 1 so we highlight only its altered parts. We first write the lower bound in (15) for $\mathbb{E}_\zeta[\log \frac{W_{\tau+1}}{W_1}]$:

$$\mathbb{E}_\zeta\left[\log \frac{W_{\tau+1}}{W_1}\right] \ge -\eta \min_{\theta \in \mathcal{B}_\tau} L_\tau(\theta). \quad (26)$$

The rest of the analysis involves computing an upper bound for $\log \frac{W_{\tau+1}}{W_1}$ in terms of $L_\tau(\boldsymbol{\pi}^{\text{HR}})$. In any round t, if a new expert z_t is revealed, then we can write

$$\begin{aligned} \frac{w_t(z_t)}{W_t} &= \frac{e^{-\eta L_t(z_t)}}{\sum_{\theta \in \mathcal{B}_{t-1}} e^{-\eta L_t(\theta)}} \le \frac{1}{N_{t-1}}(e^{-\eta \sum_{j \in \mathcal{B}_{t-1}} L_t(\theta)})^{-1/N_{t-1}} \\ &\le \frac{1}{N_{t-1}}(e^{-\eta t N_{t-1}})^{-1/N_{t-1}} = \frac{e^{\eta t}}{N_{t-1}}. \quad (27) \end{aligned}$$

Therefore,

$$\begin{aligned} \log \frac{W_{\tau+1}}{W_1} &= \sum_{t=1}^{\tau} \log \frac{W_{t+1}}{\hat{W}_t} + \sum_{t=1}^{\tau} \mathbb{1}(E_t = 1) \log\left(1 + \frac{w_t(z_t)}{W_t}\right) \\ &\le \sum_{t=1}^{\tau} \log \frac{W_{t+1}}{\hat{W}_t} + \sum_{t=1}^{\tau} \mathbb{1}(E_t = 1) \frac{e^{\eta t}}{N_{t-1}} \\ &\le \sum_{t=1}^{\tau} \log \frac{W_{t+1}}{\hat{W}_t} + e^{\eta\tau} \sum_{n=1}^{N_\tau} \frac{1}{n} \\ &\le \sum_{t=1}^{\tau} \sum_{\theta \in \mathcal{B}_t} \frac{w_t}{\hat{W}_t}\left(-\eta \tilde{l}_t(\theta) + \frac{\eta^2 \tilde{l}_t(\theta)}{2\epsilon}\right) + e^{\eta\tau}(\log N_\tau + 1). \quad (28) \end{aligned}$$

We have used (19) in the first step; $\log(1+x) < x$ and (27) in the second step; the number of events $E_t = 1$ till time τ is N_τ in the third step and (17) in the last step. By taking expectation with respect to the Bernoulli distribution of ζ on both sides of (28) and mirroring the steps to get (21), we obtain:

$$\mathbb{E}_\zeta\left[\log \frac{W_{\tau+1}}{W_1}\right] \le -\eta \mathbb{E}^{\boldsymbol{\pi}^{\text{H}}}[L_\tau(\boldsymbol{\pi}^{\text{H}})] + \eta\epsilon\tau + \frac{\eta^2\tau}{2\epsilon} + 2e^{\eta\tau}\log N_\tau. \quad (29)$$

From (26) and (29), we obtain the result. □

For $t > \tau$, Hedge-HI-Restart uses Hedge-HI with the set of experts from $\mathcal{B}_\tau$. Since θ^* belongs to $\mathcal{B}_\tau$ with probability q_τ, with this probability, the regret bound of Hedge-HI applies to the losses $L_{\tau,T}(\pi^{\text{HR}})$ and $L^*_{\tau,T}(\mathcal{B}_\tau)$ for the horizon $T - \tau$. This is stated in the following lemma (proof is deferred to [1]).

Lemma 6. *Under Hedge-HI-Restart, we have*

$$\mathbb{E}^{\boldsymbol{\pi}^{HR}}[L_{\tau,T}(\boldsymbol{\pi}^{HR})] - \min_{\theta \in \mathcal{B}_\tau} L_{\tau,T}(\theta) \le \epsilon(T - \tau) + \frac{\eta(T-\tau)}{2\epsilon} + \frac{\log N_\tau}{\eta}.$$

Summing the inequalities in Lemmas 5 and 6, we obtain

$$\begin{aligned} &\mathbb{E}^{\boldsymbol{\pi}^{\text{HR}}}[L_T(\boldsymbol{\pi}^{\text{HR}})] - [\min_{\theta \in \mathcal{B}_\tau} L_\tau(\theta) + \min_{\theta \in \mathcal{B}_\tau} L_{\tau,T}(\theta)] \\ &\le \epsilon T + \frac{\eta T}{2\epsilon} + \frac{(2e^{\eta\tau} + 1)\log N_\tau}{\eta} \\ \Rightarrow &\mathbb{E}^{\boldsymbol{\pi}^{\text{HR}}}_{\mathbf{Z}}[L_T(\boldsymbol{\pi}^{\text{HR}})] - \mathbb{E}_{\mathbf{Z}}[\min_{\theta \in \mathcal{B}_\tau} L_T(\theta)] \\ &\le \epsilon T + \frac{\eta T}{2\epsilon} + \frac{(2e^{\eta\tau} + 1)\log \mathbb{E}_{\mathbf{Z}}[N_\tau]}{\eta}, \quad \text{w.p. } q_\tau. \end{aligned}$$

In the last step above, we have used (25) and $\mathbb{E}_{\mathbf{Z}}[\log N_\tau] \le \log \mathbb{E}_{\mathbf{Z}}[N_\tau]$. The result follows from the fact that

$$\mathbb{E}_{\mathbf{Z}}[\min_{\theta \in \mathcal{B}_\tau} L_T(\theta)] \le \min_{\theta \in \mathcal{B}_\tau} \mathbb{E}_{\mathbf{Z}}[L_T(\theta)].$$