Carbon-Neutralizing Edge AI Inference for Data Streams via Model Control and Allowance Trading

Yining Zhang[1], Lei Jiao[2], Konglin Zhu[1], Yuedong Xu[3], Lin Zhang[1]
[1]Beijing University of Posts and Telecommunications, China [2]University of Oregon, USA [3]Fudan University, China

***Abstract*—To make edge AI inference carbon-neutral, we perform a comprehensive mathematical and algorithmic study on the complex online management of AI model selection and placement with carbon allowance trading. This work is non-trivial due to the critical challenges such as the unknown stochastic distributions and arrivals of inference data, the exploration-exploitation trade-off with model switching cost, and the uncertain, time-varying allowance prices and system environments. We first model a long-term stochastic cost optimization problem to capture these challenges. Then, we design a novel learning-centric decomposition-based online algorithmic framework which, on the one hand, samples and places the models repeatedly to minimize the expected inference loss with bounded model switches, and on the other hand, buys and sells carbon allowances cost-efficiently in real time toward carbon neutrality without relying on future allowance prices and system emissions. We further formally prove multiple performance guarantees of our algorithms in terms of sub-linear regret and fit. Finally, we conduct trace-driven evaluations to confirm the substantial advantages of our approach compared to baselines and state-of-the-arts in practice.**

I. Introduction

Edge AI inference [1], [2] often entails the deployment of AI models across edge computing infrastructures, such as micro data centers or server clusters co-located with cellular base stations or WiFi access points. Strategically positioned in proximity to end users, these infrastructures enable localized execution of inference tasks on user-submitted requests. This highly decentralized edge-based approach marks a transformative shift in AI service delivery, offering distinct advantages over traditional cloud-hosted solutions such as ultra-low network latency, traffic localization, and enhanced privacy preservation by confining user data within local access networks.

Achieving carbon neutrality in edge AI inference has become imperative, as these systems incur significant environmental costs [3] due to their substantial energy consumption. Within an AI model's lifecycle, the inference phase dominates the total carbon footprint, often surpassing training emissions. Recent studies indicate that inference processes contribute to 80%∼90% of a model's total energy consumption and associated emissions [4]. Consequently, decarbonizing edge AI inference is critical, necessitating innovations that offset operational emissions while maintaining system efficiency.

One widely-adopted approach to realizing carbon neutrality is leveraging *cap and trade* programs [5], [6]. That is, the carbon source, e.g., an AI service, firstly obtains an initial cap in terms of *carbon allowances* from the government for its carbon emission. Then, the carbon source can join the carbon market to sell spare allowances to the market if its carbon emission remains below the cap, and purchase additional allowances from the market if its carbon emission exceeds the allowances being held. Similar policies or mechanisms have been established in many jurisdictions across the world, e.g., China, European Union, and California in the U.S. [7]–[9].

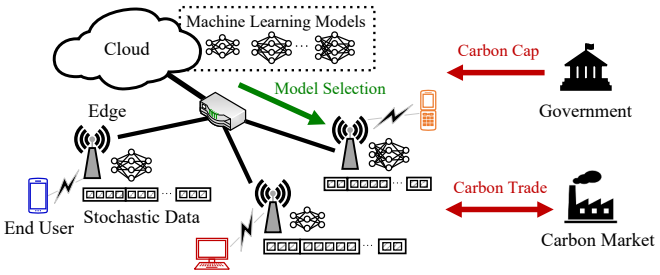

Fig. 1: System scenario

In fact, achieving carbon neutrality in this manner entails the complex management of both edge AI inference itself and carbon allowance trading, as in Fig. 1, which is non-trivial due to multiple fundamental and unique challenges as follows.

First, edge AI inference often needs to handle data streams, where the data samples themselves and the quantity of such dynamic data samples inherently follow unknown stochastic distributions [10]. To produce high-quality inference results, selecting the best models to deploy on the edges is a stochastic optimization problem that optimizes inference loss and computation overhead over the entire distributions in expectation. Yet, we only observe samples of the loss by conducting inference via the selected models upon concrete data—using samples to optimize unknown expectations is not straightforward.

Second, the lack of knowledge about model quality, data distributions, and edge platform performance requires continuous exploration and exploitation of different models, incurring model switching cost such as the communication delay in transferring the new models [11], [12]. Selecting the same model consistently avoids model switching cost, but may fail to explore other potentially better models; conversely, frequent model-selection decision changes may lead to the faster arrival at the optimum, but can cause excessive switching cost. We need to dynamically strike the balance between exploiting the best model so far and exploring a new model whose performance we do not know yet. Heterogeneous and uncertain time-varying system environments further complicate this issue.

Third, buying and selling carbon allowances cost-efficiently to maintain the long-term carbon neutrality requires cautious online decision-making repeatedly. As time goes, the carbon

allowance prices in the market typically fluctuate [8], [13], [14], and the carbon footprint to offset for the edge AI inference also changes. Then, the dilemma is that purchasing vast carbon allowances now could be unnecessary if future prices drop or system emissions decrease, but buying inadequate allowances now could force the system to buy more later, even if prices then become higher. Because future allowance prices and system emissions are unknown in prior, it is never easy to trade carbon allowances on the fly for long-term benefit.

Existing research falls insufficient in addressing the aforementioned challenges. Those works on model selection and inference in typical cloud-edge settings [15]–[20] often overlook carbon footprint and data stochasticity. Those about carbon and energy of AI services [21]–[25] either focus on reducing carbon emissions without carbon neutrality, or never consider dynamic carbon markets and online allowance trading. Finally, research on switching cost for cloud and edge systems [11], [12], [26]–[28] has not investigated our complicated stochastic optimization with long-term constraints, as described next. See Section VI for our detailed discussions on the related work.

In this paper, we present a rigorous modeling and algorithmic study on operating and carbon-neutralizing a distributed edge AI inference service. We make multiple contributions:

We first model a long-term stochastic optimization problem to optimize the total cost of the edge AI inference service, featuring the expected inference loss over dynamic data streams, the model hosting and switching cost over time, and the cumulative expense of buying and selling carbon allowances, while enforcing carbon neutrality. Our formulation controls AI model placements in the edge network and carbon allowance trading with the carbon market, and captures arbitrarily unknown stochastic distributions of data samples and arrivals and arbitrarily time-varying carbon allowance prices.

We then propose our algorithmic insights based on which we design a novel "learning-centric" polynomial-time solution framework to solve this problem online. Our approach decomposes the original problem into two subproblems and solves them respectively at each individual time slot. For the first subproblem of model selection and placement, we design a switching-aware *bandit learning* algorithm [29], [30]. Unlike conventional bandits, our algorithm controls the switching cost by dividing the time horizon into blocks of increasing length and changing model selections only at block boundaries, and overcomes the stochastic uncertainty by repeatedly sampling the models based on the unbiased estimations of the inference loss to balance exploration and exploitation. For the second subproblem of carbon allowance trading, we devise an *online learning* algorithm [31], [32] via a convex-concave reformulation with rectified online primal-dual steps, which decides carbon allowances to purchase and sell in real time for the long-term carbon neutralization without relying on future carbon allowance price and carbon emission information.

We further perform rigorous theoretical analysis for our proposed algorithms. For the first subproblem, we prove that the *regret* [33]–[35] on each edge, i.e., the difference between the expectation of inference loss and model hosting cost incurred by our approach and that incurred by the single best model at hindsight, plus model switching cost over time, only grows sub-linearly along with time. For the second subproblem, we also prove the sub-linear growth of *regret* and *fit* [36]–[38]. That is, both the time-averaged optimality gap between the cumulative carbon allowance trading expense of our approach and that of a sequence of instantaneous optimizations and the time-averaged long-term carbon neutrality violation incurred by our approach vanish progressively as time elapses. Based on these, we also derive the regret for our whole original problem via constructed redundant but useful intermediate terms.

Finally, we conduct experiments using MNIST and CIFAR-10 inference data [39], [40], London Underground user workload [41], EU Carbon Permit prices [8], and real-world data of edge server locations [42], inference latency [43], carbon rates [44] and energy consumption [45], with real-world deep neural networks [46], [47]. We compare our approach to different combinations of baselines such as random and greedy methods and state-of-the-arts such as Tsallis-INF [29] and UCB2 [48] for model selection and Lyapunov [24] for carbon trading. Our evaluations reveal the following results: (i) Compared to alternatives, our approach reduces the cumulative total cost on average by 21%$\sim$55%; (ii) Our approach performs the best consistently as the importance of the model switching cost, the carbon emission rate, or the initial carbon cap varies; (iii) Our approach incurs the lowest regret and the lowest fit for total cost minimization; (iv) Our approach achieves the highest inference accuracy using the selected models; (v) Our approach can execute effectively and finish within seconds.

II. Model and Problem Formulation

A. System Modeling

Cloud-Edge Inference System: We consider an AI service provider that owns and operates a cloud-edge system consisting of a cloud data center and a group of distributed and potentially heterogeneous "edges". Each edge is a micro data center or server cluster co-located with a cellular base station or WiFi access point in close proximity to the users. The users often connect to the edges via wireless networks, and the edges connect to the cloud via wired networks. We denote the set of edges as $\mathcal{I} = \{1, 2, \ldots, I\}$. We consider the system operating over a series of consecutive time slots $\mathcal{T} = \{1, 2, \ldots, T\}$. This service provider has a set of machine learning models in the cloud, and sends such models to the edges dynamically to conduct machine learning inference, as elaborated next.

Machine Learning Models: We denote the set of machine learning models hosted in the cloud as $\mathcal{N} = \{1, 2, \ldots, N\}$. For each model $n \in \mathcal{N}$, we use W_n to refer to its size, and use $v_{i,n}$ to refer to the computation cost of running this model on the edge i, e.g., computation latency, to conduct inference on a single "data sample" which will be defined next. We note that $v_{i,n}$ is *posterior*, i.e., it can only be observed after the model n is actually downloaded to the edge i and used to complete the inference. We denote the the communication cost, e.g., network delay, of downloading a model from the cloud to the edge i as u_i. For each model n, we also use $h_n(\cdot)$ to represent

its decision function, and without loss of generality, consider the squared loss as the inference loss function.

Stochastic Data Streams: Without loss of generality, all the data samples from the users can be considered as drawn from an unknown time-invariant stochastic distribution [49], [50]. That is, we have $(a, b) \sim \mathscr{D}$, where a is a random variable that represents the feature; b is a random variable that represents the ground-truth label; and $\mathscr{D}$ refers to the distribution. Thus, every single data sample in the system is a sample of (a, b). Accordingly, the loss incurred by using the model n to conduct the inference also follows some distribution $\mathscr{D}_n$, denoted as $l_n(a, b) = (h_n(a) - b)^2 \sim \mathscr{D}_n$. Hereafter, we simply write $l_n(a, b)$ as l_n when it is clear from the context.

Each edge receives a stream of data samples that dynamically arrive from the users. Equivalently, this can be seen as an Independent and Identically Distributed (IID) stochastic process for each edge, where a data sample corresponds to a random variable that follows $\mathscr{D}$. We envisage that the number of the data samples or such random variables, which arrive at each edge i, denoted as M_i, follows another unknown time-invariant stochastic distribution. The average loss incurred on the edge i by using the model n to conduct the inference is $L_{i,n} = \frac{1}{M_i} \sum_{\upsilon=1}^{M_i} (h_n(a_\upsilon) - b_\upsilon)^2$, where M_i and $\{a_\upsilon, b_\upsilon, \forall \upsilon\}$ are all random variables and $(a_\upsilon, b_\upsilon) \sim \mathscr{D}, \forall \upsilon$.

Machine Learning Inference: At each time slot t, on each edge i, the machine learning inference workflow is illustrated in Fig. 2, also described as follows:

- *Step 1:* The system selects and downloads one and only one (replica of) a model from the cloud to the edge. The download operation occurs if the model selected for t is different from that for $t-1$; no download operation occurs if the model selected for t is the same as that for $t-1$, because the model already on the edge can be reused. Suppose the model n_i is selected for t.
- *Step 2:* M_i^t data samples arrive sequentially, where M_i^t is a sample of the random variable M_i. Then, repeat the following *Steps 2.1*~*2.3* for each $m_i \in \{1, 2, ..., M_i^t\}$:
  - *Step 2.1:* The edge receives the feature a_{m_i}.
  - *Step 2.2:* The edge uses the model n_i to conduct the inference to obtain the inferred label $h_{n_i}(a_{m_i})$, which is sent back to the corresponding user.
  - *Step 2.3:* The edge receives the ground-truth label b_{m_i} from the same user.
- *Step 3:* The edge then computes the incurred loss $L_{i,n_i}^t = \frac{1}{M_i^t} \sum_{m_i=1}^{M_i^t} (h_{n_i}(a_{m_i}) - b_{m_i})^2$ as a sample of the random variable L_{i,n_i}, and finds the computation cost v_{i,n_i} if such cost has not been observed for the model n_i.
- *Step 4:* L_{i,n_i}^t and v_{i,n_i} are collected by the system to potentially improve the model selection decision for the edge i at the next time slot, i.e., $t+1$.

Note that for each single data sample, the order of firstly receiving the feature, afterward conducting the inference, and finally receiving the ground-truth label is a general working pattern of lots of streamed AI inference systems such as phone soft keyboards [51] and ads recommendation systems [52].

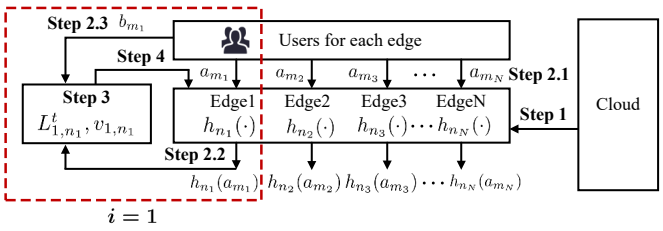

Fig. 2: Inference in a single time slot t

Carbon Allowance Trading: We consider the cap-and-trade program. That is, the cloud-edge system is pre-allocated a certain number of carbon allowances R, which are used as permits to cover carbon emissions; also, the cloud-edge system is engaged in the allowance trading with a carbon trading market to purchase allowances to offset excess carbon emissions or to sell surplus allowances. We use c^t and r^t to denote the buying price and the selling price of the carbon allowances at the time slot t, respectively. To quantify the carbon emission of the cloud-edge inference system, we multiply the total energy consumption of the inference process by the carbon emission rate ρ which is the amount of carbon emitted per unit energy consumption. We represent the energy consumption at the edge i at the time slot t when using the model n to conduct the inference as $E_{i,n}^t = \varphi_n M_i^t$, where φ_n denotes the energy for inferring one single data sample by the model n, and represent the energy consumption for transferring the model n from the cloud to the edge i as $F_{i,n} = \vartheta_i W_n$, where ϑ_i is the energy for transferring one unit size of the model to the edge i.

Control Decisions: The system makes the following control decisions dynamically at each time slot t: $x_{i,n}^t \in \{1, 0\}$, denoting whether or not to place the model n on the the edge i at the time slot t; $z^t \geq 0$, denoting the quantity of carbon allowances purchased by the system at the time slot t; and $w^t \geq 0$, denoting the quantity of carbon allowances sold by the system at the time slot t. We also introduce the auxiliary control variable $y_i^t \triangleq \mathbf{1}\{\sum_n n x_{i,n}^t \neq \sum_n n x_{i,n}^{t-1}\}$ to denote whether or not the model hosted on the edge i at the time slot t is different from that placed on the same edge at the time slot $t-1$, where $\mathbf{1}\{\cdot\}$ is the indicator function equal to 1 if the specified condition holds and 0 otherwise.

Cost of Machine Learning Inference: The cost incurred by conducting the inference at the time slot t refers to the expected inference loss over the entire data distribution, the computation cost of running the models, and the communication cost of downloading the models from the cloud to the edges: $\sum_i \sum_n x_{i,n}^t (\mathbb{E}_{l_n \sim \mathscr{D}_n}(l_n) + v_{i,n}) + \sum_i y_i^t u_i$.

Cost of Carbon Allowance Trading: The cost incurred by the carbon allowance trading for the cloud-edge inference system at the time slot t refers to the expense of purchasing carbon allowances minus the revenue obtained from selling carbon allowances: $z^t c^t - w^t r^t$.

B. Problem Formulation and Algorithmic Challenges

Total Cost Minimization: The total cost refers to the sum of the cost of machine learning inference and the cost of

carbon allowance trading over time. We formulate the total cost minimization problem $\mathbb{P}_0$ as follows:

$$\begin{aligned}
\mathbb{P}_0: \min\ & \mathcal{P} = \textstyle\sum_t \sum_i \sum_n x_{i,n}^t (\mathbb{E}_{l_n \sim \mathscr{D}_n}(l_n) + v_{i,n}) \\
& + \textstyle\sum_t \sum_i y_i^t u_i + \sum_t z^t c^t - \sum_t w^t r^t, && (1)\\
s.t.\quad & \textstyle\sum_n x_{i,n}^t = 1, \forall i, \forall t, && (1a)\\
& y_i^t = \mathbf{1}\{\textstyle\sum_n n x_{i,n}^t \neq \sum_n n x_{i,n}^{t-1}\}, \forall i, \forall t, && (1b)\\
& \textstyle\sum_t \sum_i \sum_n x_{i,n}^t \rho(E_{i,n}^t + y_i^t F_{i,n}) \\
& \leq R + \textstyle\sum_t z^t - \sum_t w^t, && (1c)\\
\text{var.}\quad & x_{i,n}^t, y_i^t \in \{0,1\}, z^t \geq 0, w^t \geq 0, \forall i, \forall n, \forall t.
\end{aligned}$$

The objective (1) minimizes the total cost over time. Constraint (1a) ensures that each edge hosts one and only one model at each time slot. Constraint (1b) captures the definition of the auxiliary variable. Constraint (1c) ensures the carbon neutrality in the long term, i.e., the cumulative carbon emission is fully covered by the cumulatively possessed carbon allowances.

Algorithmic Goal: Our goal is to design an algorithmic approach to solve the problem $\mathbb{P}_0$ online to produce the solution $\{\{\bar{x}_{i,n}^t, \forall i, \forall n\}, \{\bar{y}_i^t, \forall i\}, \bar{z}^t, \bar{w}^t, \forall t\}$[1], while provably bounding the "regret" as $\mathcal{P}(\{\{\bar{x}_{i,n}^t, \forall i, \forall n\}, \{\bar{y}_i^t, \forall i\}, \bar{z}^t, \bar{w}^t, \forall t\}) - \mathcal{P}^* \leq C$. $\mathcal{P}$ is the objective function of $\mathbb{P}_0$. $\mathcal{P}^*$ refers to the offline optimal objective value of $\mathbb{P}_0$. C is generally expected to be a parameterized constant, which further needs to be sublinear with respect to the length of the time horizon T.

Algorithmic Challenges: Solving $\mathbb{P}_0$ in an online manner to achieve our algorithmic goal in the above is non-trivial.

Stochastic Uncertainty: $\mathscr{D}_n$ is unknown and thus the expectation is not calculable. Also, at each t, we are not observing the sample of l_n, but the sample of $L_{i,n}$ for each i which involves another random variable M_i. Leveraging the latter to minimize the expectation of the former is challenging.

Switching Cost: The existence of $\sum_t \sum_i y_i^t u_i$ couples every pair of adjacent time slots, and restricts the choices for $x_{i,n}^t$ at each t. Whatever value $x_{i,n}^t$ takes, it impacts the switching cost between t and $t+1$; yet, when deciding $x_{i,n}^t$, what will occur at $t+1$ is unknown as the time slot $t+1$ has not arrived.

Long-Term Constraint: Constraint (1c) entails deciding z^t and w^t at each t to make the constraint hold in the long term. Any values that z^t and w^t take at t without considering c^t and r^t for future time slots beyond t can lead to suboptimum of $\sum_t (z^t c^t - w^t r^t)$. But c^t and r^t beyond t are unknown at t. This constraint is also nonconvex due to the multiplication of $x_{i,n}^t$ and y_i^t.

III. Online Algorithm Design

A. Algorithm Rationale with Problem Decomposition

To address the aforementioned algorithmic challenges and achieve our algorithmic goal, we propose to firstly decompose the original problem $\mathbb{P}_0$ into two subproblems $\mathbb{P}_1$ and $\mathbb{P}_2$.

Problem $\mathbb{P}_1$***:*** We present the problem $\mathbb{P}_1$ as follows.

$$\begin{aligned}
\mathbb{P}_1: \min\ & \textstyle\sum_t \sum_i \sum_n x_{i,n}^t (\mathbb{E}_{l_n \sim \mathscr{D}_n}(l_n) + v_{i,n}) \\
& + \textstyle\sum_t \sum_i y_i^t u_i, && (2)\\
s.t.\quad & \textstyle\sum_n x_{i,n}^t = 1, \forall i, \forall t, && (2a)\\
& y_i^t = \mathbf{1}\{\textstyle\sum_n n x_{i,n}^t \neq \sum_n n x_{i,n}^{t-1}\}, \forall i, \forall t, && (2b)\\
& x_{i,n}^t, y_i^t \in \{0,1\}, \forall i, \forall n, \forall t.
\end{aligned}$$

$\mathbb{P}_1$ involves the control variables $\{\{x_{i,n}^t, \forall i, \forall n\}, \{y_i^t, \forall i\}, \forall t\}$ only. Constraints (2a) and (2b) are from Constraints (1a) and (1b) of $\mathbb{P}_0$, respectively.

Problem $\mathbb{P}_2$***:*** We present the problem $\mathbb{P}_2$ as follows. We introduce some additional notations to facilitate the design of our algorithms later. We denote $f^t(\mathbf{Z}^t) = z^t c^t - w^t r^t$, and also $g^t(\mathbf{Z}^t) = \sum_i \sum_n (\bar{x}_{i,n}^t \rho(E_{i,n}^t + \bar{y}_i^t F_{i,n})) - \frac{R}{T} - z^t + w^t$, where $\mathbf{Z}^t$ represents z^t and w^t collectively, and the values $\{\{\bar{x}_{i,n}^t, \forall i, \forall n\}, \{\bar{y}_i^t, \forall i\}\}$ are solved from $\mathbb{P}_1$ at t.

$$\begin{aligned}
\mathbb{P}_2: \min & \textstyle\sum_t f^t(\mathbf{Z}^t), && (3)\\
s.t.\quad & \textstyle\sum_t g^t(\mathbf{Z}^t) \leq 0, && (3a)\\
& \mathbf{Z}^t \in \mathcal{X} = \{z^t, w^t | z^t \geq 0, w^t \geq 0, \forall t\}.
\end{aligned}$$

$\mathbb{P}_2$ involves the control variables $\{z^t, w^t, \forall t\}$ only, which takes $\{\{\bar{x}_{i,n}^t, \forall i, \forall n\}, \{\bar{y}_i^t, \forall i\}\}$ as the input at each t. Constraint (3a) is from Constraint (1c) of $\mathbb{P}_0$. Note that in this case (3a) becomes a linear constraint.

Algorithm Rationale: Our decomposition is motivated by our idea of containing the algorithmic challenges in different subproblems and overcoming them separately.

- For $\mathbb{P}_1$, we note $\mathbb{E}_{l_n \sim \mathscr{D}_n}(l_n) + v_{i,n} = \mathbb{E}_{l'_{i,n} \sim \mathscr{D}'_{i,n}}(l'_{i,n})$, where $l'_{i,n} \triangleq l_n + v_{i,n}$ is a new random variable for each i and n and informally, the corresponding new distribution can be written as $\mathscr{D}'_{i,n} = \mathscr{D}_n + v_{i,n}$. Therefore, we can actually treat $\mathbb{P}_1$ through *bandit learning* for each single i, respectively, as Constraints (2a) and (2b) can be readily further decomposed for each i. What is needed here is to cautiously leverage the sample of $L_{i,n}$ at each time slot for $l'_{i,n}$, while taking care of the switching cost.
- For $\mathbb{P}_2$, note that, if the values $\{\{\bar{x}_{i,n}^t, \forall i, \forall n\}, \{\bar{y}_i^t, \forall i\}\}$ are obtained from $\mathbb{P}_1$ at each t, we can then focus on obtaining $\{\bar{z}^t, \bar{w}^t, \forall t\}$ by decomposing $\mathbb{P}_2$ into a series of one-shot problems, denoted as $\{\mathbb{P}_2^t, \forall t\}$, and treating them through *online learning* to overcome the long-term constraint (3a).

Our algorithms, with their design insights and time complexity analysis, are elaborated in the next two sections for $\mathbb{P}_1$ and $\mathbb{P}_2$, respectively.

B. Model Selection via Switching-Aware Bandit Learning

Unfortunately, $\mathbb{P}_1$ for each edge i is not a standard Multi-Armed Bandit (MAB) problem with the models as the "arms". Solving it is not straightforward. Yet, we have two insights:

- *Insight 1:* Unlike exploration vs. exploitation in conventional MAB, switching cost is now embedded whenever we change from one model to another. To prevent excess switching cost, we need to explicitly restrict the model-selection decision changes.

[1]In this paper, we use notations like $x_{i,n}^t$ to represent the *decision variables* and notations like $\bar{x}_{i,n}^t$ to represent the *values* of the corresponding variables.

- *Insight 2:* Even though $l'_{i,n} = l_n + v_{i,n}$ and the random variable l_n differs from the random variable $L_{i,n}$, $\forall i$ as the latter depends on M_i, we can actually use $L^t_{i,n} + v_{i,n}$ as a sample at the time slot t for $l'_{i,n}$. That is, the random variable of the arriving data samples, i.e., M_i, does not matter here, as formally shown in Appendix A.

Simultaneously motivated by our two insights as the above, for each edge i, we divide the time horizon T into a sequence of blocks $\{B_{i,k}\}$, where $k \geq 1$, with the block length $|B_{i,k}|$ representing the number of time slots contained. The key is that we keep choosing the same model for all the time slots within each block, and only allow model changes across block boundaries. We now define K_i as the smallest integer which satisfies $\sum_{k=1}^{K_i} |B_{i,k}| \geq T$, and truncate the last block so that the cumulative lengths of the first K_i blocks sum up to T exactly. Then, the total number of model switches on the edge i can be bounded by K_i, i.e., $\sum_t y^t_i \leq K_i$. By appropriately setting the block lengths, together with other parameters such as learning rates, we will be able to upper-bound the regret.

Our Algorithm 1, for each block at each edge, samples a model at the beginning of the block and maintains this decision within the block. Line 3 calculates the probabilities to be used for sampling, which is based on online mirror descent with Tsallis entropy regularization [29]. $\Delta = \{p_{i,k,n}, \forall n | p_{i,k,n} \geq 0 \text{ and } \sum_n p_{i,k,n} = 1\}$ is the probability simplex. Line 4 is the actual sampling step. We define J^t_i as the model selected for the edge i at the time slot t, and $J_{i,k}$ as the model selected for the edge i in the block k. Thus, we have $J^t_i = J_{i,k}, \forall t \in B_{i,k}$. Based on this, Lines 5 and 6 update the control decisions. In Line 7, we observe $c_{i,k,J_{i,k}} = \sum_{t \in B_{i,k}} (L^t_{i,J_{i,k}} + v_{i,J_{i,k}})$, which is the cumulative inference loss incurred by the selected model $J_{i,k}$ in the block k. In Line 8, as we do not observe the complete loss vector $\boldsymbol{c}_{i,k}$ in the bandit setting, we construct an unbiased estimator $\hat{\boldsymbol{c}}_{i,k}$ via importance sampling [29]. It is unbiased, due to $\mathbb{E}[\hat{\boldsymbol{c}}_{i,k}] = \sum_n \hat{c}_{i,k,n} p_{i,k,n} = \sum_n \frac{\mathbf{1}(J_{i,k}=n) c_{i,k,n}}{p_{i,k,n}} p_{i,k,n} = \sum_n \mathbf{1}(J_{i,k} = n) c_{i,k,n} = \mathbb{E}[\boldsymbol{c}_{i,k}]$. Finally, the cumulative total loss suffered by each model n over the blocks is updated in Line 9.

Complexity Analysis: Algorithm 1 takes $O(K_i \cdot (\log(1/\epsilon) + 4N + \log N) + 3T)$. In each of the K_i iterations, Line 3 is solvable by existing optimization solvers. For example, it takes $O(\log(1/\epsilon) + N)$ to firstly find an ϵ-accurate solution via the Brent method [53] and then obtain the values of $p_{i,k,n}$, $\forall n \in \mathcal{N}$. Line 4 can be implemented as weighted sampling via binary search, which takes $O(N + \log N)$. It is then obvious to count in the complexities of Lines 5∼9 to obtain the overall complexity of Algorithm 1.

C. Carbon Trading via Long-Term-Aware Online Learning

- *Insight 3:* Our insight for addressing the long-term constraint in $\mathbb{P}_2$ is to remove that long-term constraint by absorbing it into the objective via Lagrange relaxation and then design an online learning (a.k.a. online convex optimization) algorithm to solve the transformed problem while upper-bounding the cumulative violation of the original long-term constraint.

Algorithm 1: Online Model Selection Algorithm, $\forall i$

Input: Learning rates $\eta_{i,1} \geq \eta_{i,2} \geq \cdots \geq \eta_{i,K_i} > 0$; block lengths $|B_{i,1}|, |B_{i,2}|, \ldots, |B_{i,K_i}|$.

1 **initialize** $\hat{C}_{i,0}(n) = 0, \forall n$; $\bar{x}^t_{i,n} = 0$, $\forall n$, $\forall t$;

2 **for** $k = 1, 2, \ldots, K_i$ **do**

3 $\quad \{p_{i,k,n}, \forall n\} = \arg\min_{\{p_{i,k,n}, \forall n\} \in \Delta} \{\sum_n p_{i,k,n} \hat{C}_{i,k-1}(n) - \sum_n \frac{4\sqrt{p_{i,k,n}} - 2p_{i,k,n}}{\eta_{i,k}}\}$;

4 $\quad$ Select a model as $J_{i,k}$ using probabilities $p_{i,k,n}$,$\forall n$;

5 $\quad \bar{x}^t_{i,J_{i,k}} = 1$, $J^t_i = J_{i,k}$, $\forall t \in B_{i,k}$;

6 $\quad \bar{y}^t_i = \begin{cases} 1, & \textbf{if } J^t_i \neq J^{t-1}_i \\ 0, & \textbf{if } J^t_i = J^{t-1}_i \end{cases}, \forall t \in B_{i,k}$;

7 $\quad$ Observe $c_{i,k,J_{i,k}} = \sum_{t \in B_{i,k}} (L^t_{i,J_{i,k}} + v_{i,J_{i,k}})$;

8 $\quad \hat{c}_{i,k,n} = \begin{cases} \frac{c_{i,k,n}}{p_{i,k,n}}, & \textbf{if } J_{i,k} = n \\ 0, & \textbf{if } J_{i,k} \neq n \end{cases}, \forall n$;

9 $\quad \hat{C}_{i,k}(n) = \hat{C}_{i,k-1}(n) + \hat{c}_{i,k,n}, \forall n$;

Algorithm 2: Online Carbon Trading Algorithm

Input: Initial decision $\bar{\mathbf{Z}}^0$; $\lambda^1 = 0$; step sizes γ_1, γ_2.

1 **for** $t = 1, 2, \ldots, T$ **do**

2 $\quad$ Obtain $\bar{\boldsymbol{x}}^t$, $\bar{\boldsymbol{y}}^t$ produced by **Algorithm 1**;

3 $\quad$ Update $\bar{\mathbf{Z}}^t$ according to (4);

4 $\quad$ Given $\bar{\boldsymbol{x}}^t$, $\bar{\boldsymbol{y}}^t$, observe $f^t(\bar{\mathbf{Z}}^t)$ and $g^t(\bar{\mathbf{Z}}^t)$;

5 $\quad$ Update λ^{t+1} according to (5);

Our Algorithm 2 proceeds as follows. We firstly note that solving $\mathbb{P}_2$ is equivalent to solving its convex-concave version:

$$\min_{\mathbf{Z}^t \in \mathcal{X}} \max_{\lambda} \textstyle\sum_t \mathcal{L}^t(\mathbf{Z}^t, \lambda) = \sum_t (f^t(\mathbf{Z}^t) + \lambda g^t(\mathbf{Z}^t)),$$

where λ is the Lagrange multiplier. Using this, we can solve $\mathbb{P}_2$ in an online primal-dual manner. That is, at each time slot t, we obtain $\bar{\mathbf{Z}}^t$ as the minimizer of the following problem.

$$\mathbb{P}^t_2 : \min_{\mathbf{Z}^t \in \mathcal{X}} \nabla f^{t-1}(\bar{\mathbf{Z}}^{t-1})(\mathbf{Z}^t - \bar{\mathbf{Z}}^{t-1}) + \lambda^t g^{t-1}(\mathbf{Z}^t) + \frac{\|\mathbf{Z}^t - \bar{\mathbf{Z}}^{t-1}\|^2}{2\gamma_2}, \tag{4}$$

Given $\bar{\mathbf{Z}}^t$, we prepare the dual variable for the next time slot $t+1$:

$$\lambda^{t+1} = [\lambda^t + \gamma_1 \nabla_\lambda \mathcal{L}^t(\bar{\mathbf{Z}}^t, \lambda^t)]^+ = [\lambda^t + \gamma_1 g^t(\bar{\mathbf{Z}}^t)]^+. \tag{5}$$

$\gamma_1 > 0$ and $\gamma_2 > 0$ are predefined step sizes; $\nabla f^{t-1}(\bar{\mathbf{Z}}^{t-1})$ is the gradient of $f^{t-1}(\cdot)$ at $\bar{\mathbf{Z}}^{t-1}$; $\nabla_\lambda \mathcal{L}^t(\bar{\mathbf{Z}}^t, \lambda^t)$ is the gradient of $\mathcal{L}^t(\bar{\mathbf{Z}}^t, \lambda)$ at λ^t; and $[\cdot]^+ = \max\{\cdot, 0\}$.

We highlight that, to obtain $\bar{\mathbf{Z}}^t$ at each t, we only need the inputs until (and even excluding) t, which is advantageous as no future information beyond t is required. In fact, in such alternate descent-ascent steps, while the dual ascent step is standard, the primal descent step is not; in contrast, the primal descent step directly uses and thus penalizes the constraint function, rather than its first-order approximation, while also adopting a regularization and proximal term. We will show later that we obtain provable performance using this approach.

Complexity Analysis: Algorithm 2 takes $O(TA^2 \log(1/\epsilon) + T)$. The key is to solve $\mathbb{P}_2^t$ at each $t \in \mathcal{T}$, which can be done through any standard convex optimization solver. For instance, the interior-point method obtains an ϵ-accurate solution in $O(A^2 \log(1/\epsilon))$, where $A = 2$ is the number of the decision variables in our case [54].

IV. Theoretical Analysis

By Theorem 1, we define and characterize the regret plus the cumulative switching cost for each edge in the problem $\mathbb{P}_1$, where the regret refers to the gap between the expectation of the inference loss and the model hosting cost incurred by our approach and that incurred by the single best model at hindsight. The regret turns out to be sub-linear, i.e., its growth is even slower than the progression of time. This is a good result in general. Our analysis aligns with lots of existing work in the sense that the switching cost is not placed into the regret; having it in the regret for further analysis could be of independent interest.

Theorem 1 *Define the regret for $\mathbb{P}_1$ with regard to the single best model n_i^* for the edge i as $\mathrm{Reg}_{1,i}^T = \mathbb{E}[\sum_t (l_{J_i^t} + v_{i,J_i^t})] - \mathbb{E}[\sum_t (l_{n_i^*} + v_{i,n_i^*})]$, where $n_i^* \in \arg\min_n \mathbb{E}[\sum_t (l_n + v_{i,n})]$. Then, with $\eta_{i,k} = \frac{2}{d_{i,k}+1}\sqrt{\frac{2}{k}}$ as the learning rates and $|B_{i,k}| = \max\{\lceil d_{i,k} \rceil, 1\}$ as the block lengths, where $d_{i,k} = \frac{3u_i}{2}\sqrt{\frac{k}{N}}$, and also $\Delta_{i,n} = \mathbb{E}[l_n + v_{i,n}] - \min_n \mathbb{E}[l_n + v_{i,n}]$ as the suboptimality gap of the model n for the edge i, we have the following result from Algorithm 1:*

$$\mathrm{Reg}_{1,i}^T + \textstyle\sum_t u_i \bar{y}_i^t \leq \mathcal{O}((u_i N)^{\frac{2}{3}} T^{\frac{1}{3}} + u_i^2 + \ln T) \cdot \sum_{n \neq n_i^*} \frac{1}{\Delta_{i,n}}$$

Proof. See Appendix A. □

By Theorem 2, we define and characterize the regret and the fit for the problem $\mathbb{P}_2$. While the regret captures the difference between the objective value incurred by our approach and the sum of the series of one-shot optimums, the fit reflects the violation of Constraint (3a). Recall that (3a) has been absorbed into the objective in our approach, so it is important to quantify the violation of this constraint. Our analysis demonstrates that both the regret and the fit grow only sub-linearly.

Theorem 2 *The regret and the fit for $\mathbb{P}_2$ satisfy*

$$\mathrm{Reg}_2^T := \textstyle\sum_{t=1}^T f^t(\bar{\mathbf{Z}}^t) - \sum_{t=1}^T f^t(\bar{\mathbf{Z}}^{t*}) \leq \mathcal{O}(T^{\frac{2}{3}}),$$

$$\mathrm{Fit}^T := \|[\textstyle\sum_{t=1}^T \mathbf{g}^t(\bar{\mathbf{Z}}^t)]^+\| \leq \mathcal{O}(T^{\frac{2}{3}}),$$

where for each t, $\bar{\mathbf{Z}}^t$ represents the output of Algorithm 2; $\bar{\mathbf{Z}}^{t*} \in \arg\min_{\mathbf{Z} \in \mathcal{X}^t} f^t(\mathbf{Z})$; and $\mathcal{X}^t := \{\mathbf{Z} | g^t(\mathbf{Z}) \leq 0; z^t \geq 0, w^t \geq 0\}, \forall t$.

Proof. See Appendix B. □

By Theorem 3, we analyze the regret for the original problem $\mathbb{P}_0$. We note that the proof here is also non-trivial. Directly adding up the results of Theorems 1 and 2 is incorrect, as their corresponding optimums are in different senses. In fact, proving Theorem 3 requires the careful decomposition of the total regret into multiple components, with constructed intermediate terms to assist and complete the derivations.

Theorem 3 *The regret for $\mathbb{P}_0$ with respect to T via jointly using Algorithms 1 and 2 is*

$$\overline{\mathcal{P}} - \mathcal{P}^* \leq \mathcal{O}(T^{\frac{1}{3}} + \ln T) + \mathcal{O}(T^{\frac{2}{3}}) + \Omega_1,$$

where Ω_1 is a constant independent of T.

Proof. See Appendix C. □

V. Experimental Study

A. Experimental Settings

Datasets and Models: We adopt the MNIST [39] and the CIFAR-10 [40] datasets. We consider three types of models for MNIST: (i) the Convolutional Neural Network (CNN) with two 3×3 convolutional layers (32 or 64 channels) with ReLU activation, each of them followed by a 2×2 max pooling layer, a fully-connected layer, and a softmax output layer; (ii) the LeNet-5 [46]; and (iii) the Multilayer Perceptron (MLP) with two fully-connected layers. We consider three types of models for CIFAR-10: (i) the CNN with two 3×3 convolutional layers (64 or 128 channels), with other settings the same as that for the MNIST data; (ii) the LeNet-5; and (iii) the MobileNet V1 [47]. We consider 2 models for each type and thus 6 models for each dataset. We easily obtain the actual size of each model.

Cloud, Edge, and Inference Workload: We envisage that the cloud and the edges are deployed at the real-world cellular base stations in Australia [42]. We choose the first site in this dataset, i.e., a base station in Northern Territory, as the cloud location, and choose 10∼50 other sites as the edge locations. We use the real-world geographical distance to estimate the network delay. We select the top 10∼50 stations with the highest passenger counts of London's 268 underground stations [41], and use such dynamic passenger counts to represent the inference workload fluctuations for each edge. The passenger data are measured for every 15 minutes on a Thursday and a Friday in 2020. Thus, we consider a two-day period of 160 time slots in our experiments. For each edge, we randomly sample 8000 data points from the test data of our two datasets and use those as the incoming data streams, respectively.

Carbon and Energy: We use the computation latency in $[25, 150]$ ms [43], [55]. The buying price of carbon allowances is randomly taken from the prices of the EU Carbon Permits [8] from March 2023 to March 2024, i.e., $[5.9, 10.9]$ cent/kg, and the selling price of carbon allowances is set to 90% of the buying price [56]. The initial cap of carbon emissions is set to 500. The carbon emission rate per unit energy consumption is 500 g/kWh [44]. The energy consumed for conducting the inference on a single data sample is $[6, 10] \times 10^{-8}$ kWh [45], [55]. The energy consumed for sending one unit size of the model from the cloud to the edge is 1.02×10^{-16} kWh [57].

Algorithms: We compare our proposed approach in this paper to multiple combinations of different existing algorithms.

For model selection, we consider the following methods: (i) `Random`, where each edge selects a model randomly at

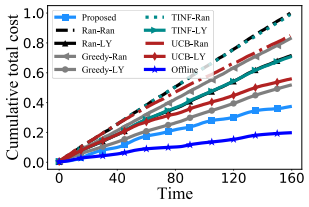

Fig. 3: Cumulative total cost

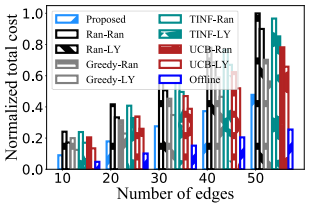

Fig. 4: Total cost

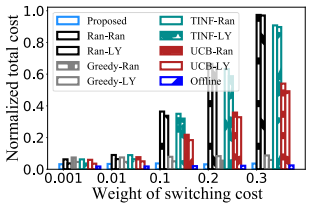

Fig. 5: Impact of switching cost

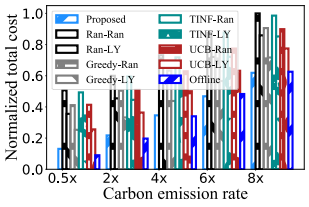

Fig. 6: Impact of carbon rate

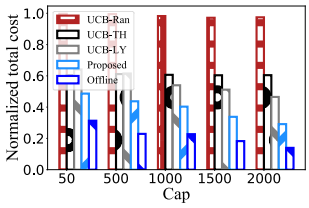

Fig. 7: Impact of cap

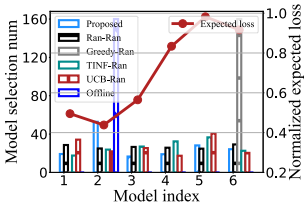

Fig. 8: Model selection

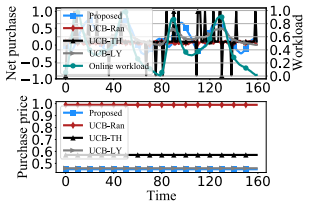

Fig. 9: Carbon trading

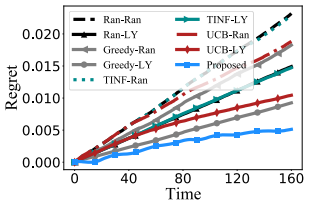

Fig. 10: Regret

each time slot; (ii) `Greedy`, where each edge selects the model with the lowest energy consumption at each time slot; (iii) `Tsallis-INF`, a state-of-the-art bandit algorithm [29], which does not consider switching cost; and (iv) `UCB2`, a state-of-the-art bandit algorithm [30], [48], which upper-bounds the switching cost.

For carbon trading, we consider the following methods: (i) `Random`, where the quantity of carbon allowances bought and sold at each time slot is random; (ii) `Threshold`, where, at each t, a fixed quantity is bought when c^t is below some value and a fixed quantity is sold when r^t is above some value; and (iii) `Lyapunov`, a state-of-the-art method that solves time-averaged stochastic optimizations using virtual queues and drift-plus-penalty mechanisms [22]–[24].

We combine these algorithms for model selection and for carbon trading, and denote all the combined approaches as `Ran-Ran`, `Ran-TH`, `Ran-LY`, `Greedy-Ran`, `Greedy-TH`, `Greedy-LY`, `TINF-Ran`, `TINF-TH`, `TINF-LY`, `UCB-Ran`, `UCB-TH`, and `UCB-LY`, respectively. For visualization clarity, we may not show all these algorithms in the result figures.

We also consider the offline optimum, written as `Offline`. `Offline` chooses the model with the minimum expectation of the inference loss in terms of the posterior average for each edge, where we use the sample mean of the inference loss obtained from the entire test dataset as an approximation to the expectation of the unknown underlying distribution. Keeping using the best model on each edge, `Offline` solves carbon trading by the Gurobi [58] solver, assuming that all the inputs in the entire time horizon are known in advance beforehand.

B. Experimental Results

All our experimental results represent the average of 10 runs due to the randomness in our algorithms.

Fig. 3 illustrates the normalized cumulative total cost in real time for different algorithms with 10 edges. Our proposed approach exhibits a slower trend compared to all others, and is the closest to the offline optimum.

Fig. 4 presents the normalized total cost of the different algorithms. As the system scales up in terms of the number of the edges, our approach always incurs the lowest cost. Compared to `Ran-Ran`, `Ran-LY`, `Greedy-Ran`, `Greedy-LY`, `TINF-Ran`, `TINF-LY`, `UCB-Ran`, and `UCB-LY`, our approach reduces the total cost on average by 55%, 46%, 41%, 21%, 55%, 45%, 45% and 30%, respectively.

Fig. 5 visualizes the total cost as the weight associated to switching cost increases. While the total cost of the other algorithms increases significantly, the total cost of our approach remains almost unchanged. This is because as the weight of switching cost grows, the block length of our algorithm also increases, reducing the number of switches. `Greedy` ranks just below ours, because it consistently selects the model with the lowest energy consumption, resulting in minimal switching. On each edge, `Offline` incurs switching cost at the initial time slot and no switching cost thereafter.

Fig. 6 depicts the impact of the carbon emission rate on the total cost. As such rate increases, the carbon emission rises, leading to the purchase of more carbon allowances to achieve carbon neutrality and thus increasing the total cost. Compared to the other algorithms except `Offline`, our approach still has the minimum total cost. As the carbon emission rate grows, the total cost of our algorithm is lower than that of `Offline`. This is because `Offline` fully satisfies the carbon neutrality constraint without any violation, while our algorithm allows instantaneous violations and tries to satisfy it in the long run.

Fig. 7 demonstrates the impact of the initial carbon cap on the total cost. As the cap increases, the quantity of carbon allowances that need to be purchased decreases, leading to a reduction in the total cost of our approach, `Offline`, and `UCB-LY`, while the total cost of ours remains the lowest except for `Offline`. The total costs of `UCB-Ran` and `UCB-TH` do not decrease with the increasing cap as their carbon trading decisions are not related to the cap.

Fig. 8 shows the relationship between the number of model selections and the corresponding expected loss. This figure uses only one edge randomly, and records the number of times each model is selected for it. `Offline` selects the model with the minimum loss, while `Greedy` selects the model with the lowest energy. In our approach, as the expected loss decreases,

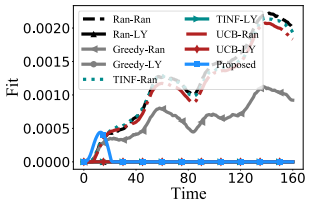

Fig. 11: Fit

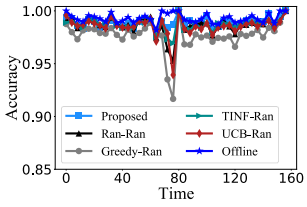

Fig. 12: Accuracy of MNIST

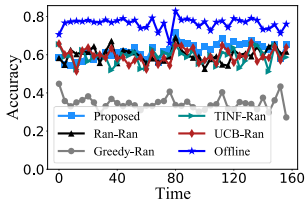

Fig. 13: Accuracy of CIFAR-10

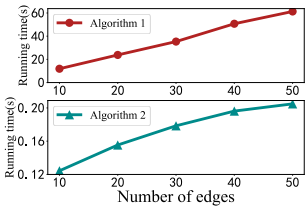

Fig. 14: Algorithm running time

the frequency of selecting the corresponding model increases.

Fig. 9 shows the relationship between the carbon allowance trading volume and the inference workload of data streams. The former refers to the normalized net value of purchase. The net purchase of our approach varies with the workload, because as workload increases, the carbon emissions incurred by inference increase, resulting in the purchase of more allowances. Yet, the net purchase of `UCB-Ran` and `UCB-TH` is only related to the trading price and does not consider workload. This figure also compares the normalized unit cost of carbon purchase, and our approach outperforms all others.

Fig. 10 and 11 exhibit the regret and the fit for $\mathbb{P}_0$ incurred by different algorithms as the length of the time horizon varies. Our approach outperforms others in regret. Although the fit of ours starts with a non-zero value initially, it quickly decreases to zero. This figure also confirms our theoretical results that the regret and the fit grow only sub-linearly with time.

Fig. 12 and 13 evaluate the inference accuracy at each time slot over the entire streams of the MNIST and the CIFAR-10 data, respectively. The accuracy of `Greedy-Ran` is the worst, because it only considers the energy consumption of models, without other factors, when making control decisions. `UCB-Ran` and `TINF-Ran` perform similarly to our approach, while ours is closer to `Offline`.

Fig. 14 depicts the execution time per time slot for each of our proposed algorithms on a commodity computer with a 3.2-GHz AMD Ryzen 7 5800H CPU and 16-GB memory. As the number of edges reaches 50, Algorithm 1 finishes in 61.32 seconds, and Algorithm 2 finishes in 0.21 seconds, performing very well compared to the single time slot of 15 minutes.

VI. Related Work

We discuss related work in different categories, and for each category, we collectively highlight the shortcomings.

Cloud/Edge Model Selection and Inference: Bai et. al. [15] constructed a Deep Neural Network (DNN) model ensembles based on the features of inference tasks. Zhao et. al. [16] jointly considered configuration adaptation, model selection, and resource provisioning to achieve scalable edge DNN inference serving. Lu et. al. [17] utilized end users' quality of experience feedback to guide DNN selections. Li et. al. [18] and Lim et. al. [19] studied DNN model partitioning and collaborative inference in mobile networks. Jin et. al. [20] minimized the model accuracy loss at the edge via appropriate model placements considering inference query queues.

These works explore model selection and inference in cloud-edge scenarios, but only consider inference loss or system cost often upon concrete data samples, neglecting carbon emissions generated during the inference processes. None of them have considered the stochasticity of online data streams.

AI/ML Carbon Footprint: Su et. al. [21] minimized the inference accuracy loss under the long-term carbon emission cap. Ma et. al. [22] considered the carbon emissions generated by cloud-edge inference and achieved carbon neutrality through purchasing carbon credits. Bian et. al. [23] considered the carbon footprint generated by training AI models and made data center selection decisions within a carbon footprint budget. Yang et. al. [24] proposed a carbon-intensity-based scheduling policy, which reduced the cumulative carbon emissions of AI model training tasks. Zhang et. al. [25] utilized multi-agent reinforcement learning to reduce carbon emissions from AI-Generated Content model training.

This group of research either focuses on reducing carbon footprint or emissions of AI/ML services without considering carbon neutrality, or neglects the carbon market with dynamic carbon allowance prices and online carbon trading decisions. Data stochasticity is not typically incorporated, either.

Bandit Learning with Switching Cost: Steiger et. al. [11] introduced a block-based algorithm to address the bandit problem with arm selection constraints and switching costs in the cloud-edge scenario. Shi et. al. [12] considered a bandit learning problem regarding model selection with the switching cost of model downloading in an edge AI setting. Huang et. al. [26] solved an adversarial bandit problem with switching cost via a block-based algorithm. Appavoo et. al. [27] utilized the bandit techniques to solve the wireless network selection problem with network switching costs. Zhu et. al. [28] formulated the task offloading problem as a delayed bandit feedback problem with the cost of switching the offloading target.

None of these studies have modeled the specific multi-model edge AI inference scenario upon stochastic data streams, not to mention the carbon footprint or neutrality. Bandit learning has not been employed with online learning to jointly solve long-term stochastic optimization as in our work.

VII. Conclusion and Future Work

Edge AI inference is an indispensable component of edge AI and also a significant contributor to the carbon footprint of the latter. While it is undoubtedly crucial to make edge AI inference carbon-neutral, the data stochasticity, intertwined with the model switching cost, and the interaction with the carbon market through dynamic trading of carbon allowances have been unfortunately overlooked so far. This paper bridges the gap. We have conducted a novel modeling and algorithmic

study of this problem, featuring the fusion of bandit learning and online learning techniques for long-term stochastic optimization, and have covered all aspects of the algorithm design, the formal analysis, and the empirical experiments.

Several exciting future directions could emerge from this work. First, while our current approach treats carbon allowance prices as exogenous, integrating price prediction models could further optimize trading strategies. Second, we plan to extend our framework to support Large Language Models (LLMs) at the edge, addressing their high computational demands and memory footprint via quantization-aware carbon or energy control. Finally, deploying our system in real-world cloud-edge environments would validate its robustness and scalability.

ACKNOWLEDGMENT

This work was supported by the National Key Research and Development Program of China (2023YFB2704500), by the Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing, and by the U.S. National Science Foundation (CNS-2047719 and CNS-2225949). Corresponding authors are Lei Jiao (ljiao2@uoregon.edu) and Konglin Zhu (klzhu@bupt.edu.cn).

APPENDIX

A. Proof of Theorem 1

We structure this proof into three steps as follows.

Step I: We first define

$$\Phi_k(C)=\max_{p\in\Delta}\{\langle p,C\rangle+\textstyle\sum_n \frac{4\sqrt{p_{i,k,n}}-2p_{i,k,n}}{\eta_{i,k}}\}.$$

Note that the probabilities $p_{i,k}$ that are used to draw the model $J_{i,k}$ for the block B_k satisfy $p_{i,k}=\nabla\Phi_k(-\tilde{C}_{i,k-1})$. Thus, the block-based regret regarding $L_{i,n}$ when $L^t_{i,n}$ is observed can be written as

$$\begin{aligned}&R_i^T=\mathbb{E}[\textstyle\sum_{k=1}^{K_i} c_{i,k,J_{i,k}}]-\min_n \mathbb{E}[\sum_{k=1}^{K_i} c_{i,k,n}]\\&=\mathbb{E}[\textstyle\sum_{k=1}^{K_i} c_{i,k,J_{i,k}}+\Phi_k(-\tilde{C}_{i,k})-\Phi_k(-\tilde{C}_{i,k-1})]\\&+\mathbb{E}[\textstyle\sum_{k=1}^{K_i}\Phi_k(-\tilde{C}_{i,k-1})-\Phi_k(-\tilde{C}_{i,k})-c_{i,k,n_i^*}],\end{aligned}$$

where the first term is the stability term; the second term is the penalty term; and n_i^* is the best arm for the edge i.

Then, we introduce a bound of the *cumulative switching cost* for any fixed n_i^*. There is always a switch at the round 1. For subsequent rounds, when there is a switch at round k, at least one of $J_{i,k-1}$ or $J_{i,k}$ is not equal to n_i^*, where $J_{i,k}$ is the index of the model downloaded by the edge i at the block k. We have

$$\mathbb{P}(J_{i,k-1}\neq J_{i,k})\leq\textstyle\sum_{n\neq n_i^*}\mathbb{P}(J_{i,k-1}=n)+\mathbb{P}(J_{i,k}=n),$$

and the cumulative switching cost for the edge i satisfies

$$\begin{aligned}&\textstyle\sum_t u_i\bar{y}_i^t=u_i+\sum_{k=2}^{K_i}u_i\mathbb{P}(J_{i,k-1}\neq J_{i,k})\\&\leq u_i+\textstyle\sum_{k=2}^{K_i}u_i(\sum_{n\neq n_i^*}\mathbb{P}(J_{i,k-1}=n)+\mathbb{P}(J_{i,k}=n))\\&\leq u_i+\textstyle\sum_{k=1}^{K_i}\sum_{n\neq n_i^*}2u_i\mathbb{P}(J_{i,k}=n)\\&=u_i+\textstyle\sum_{k=1}^{K_i}\sum_{n\neq n_i^*}2u_i\mathbb{E}[p_{i,k,n}].\end{aligned}$$

Combining the results mentioned above, we can apply such results to blocks. We first calculate an upper bound on the number of blocks K_i. Let $K_i^*=N^{\frac{1}{3}}(T/u_i)^{\frac{2}{3}}$. Observe that

$$\begin{aligned}&\textstyle\sum_{k=1}^{\lfloor K_i^*\rfloor+1}|B_{i,k}|\geq\sum_{k=1}^{\lfloor K_i^*\rfloor+1}\frac{3u_i\sqrt{k}}{2\sqrt{N}}\geq\int_0^{\lfloor K_i^*\rfloor+1}\frac{3u_i\sqrt{k}}{2\sqrt{N}}\\&\geq\textstyle\int_0^{K_i^*}\frac{3u_i\sqrt{k}}{2\sqrt{N}}=\frac{u_i}{\sqrt{N}}(K_i^*)^{\frac{3}{2}}\geq T.\end{aligned}$$

Thus, we can upper-bound K_i by $N^{\frac{1}{3}}(T/u_i)^{\frac{2}{3}}+1$. Then we bound $\eta_{i,k}|B_{i,k}|^2$ for all $k\leq K_i$ as

$$\tfrac{\eta_{i,k}}{2}|B_{i,k}|^2\leq\tfrac{\sqrt{2}}{\sqrt{k}}(\tfrac{3u_i\sqrt{k}}{2\sqrt{N}}+1)\leq\tfrac{3u_i}{\sqrt{2N}}+\tfrac{\sqrt{2}}{\sqrt{k}}.$$

Regarding the *stability term*, we use Lemma 1 in [59] and the previous result to bound $\frac{\eta_{i,k}}{2}|B_{i,k}|^2$. Then we see that the stability term is upper-bounded by

$$\begin{aligned}&\textstyle\sum_{k=1}^{K_i}(\frac{3\sqrt{2}u_i}{2\sqrt{N}}+\frac{\sqrt{2}}{\sqrt{k}})\sum_{n\neq n_i^*}(\sqrt{\mathbb{E}[p_{i,k,n}]}+2.5\mathbb{E}[p_{i,k,n}])\\&+\textstyle\sum_{k=1}^{K_{i,0}}(\frac{3\sqrt{2}u_i}{2\sqrt{N}}+\frac{\sqrt{2}}{\sqrt{k}}),\end{aligned}$$

where $K_{i,0}=128$ for $k\geq K_{i,0}$, $\eta_{i,k}\,|B_{i,k}|\leq\frac{1}{4}$.

Regarding the *penalty term*, we first bound the difference between the inverses of two consecutive learning rates as

$$\begin{aligned}&\eta_{i,k}^{-1}-\eta_{i,k-1}^{-1}\\&=(\tfrac{3u_i\sqrt{k}}{2\sqrt{N}}+1)\tfrac{\sqrt{k}}{2\sqrt{2}}-(\tfrac{3u_i\sqrt{k-1}}{2\sqrt{N}}+1)\tfrac{\sqrt{k-1}}{2\sqrt{2}}\\&=\tfrac{3\sqrt{2}u_i}{8\sqrt{N}}+\tfrac{\sqrt{k}-\sqrt{k-1}}{2\sqrt{2}}\\&\leq\tfrac{3\sqrt{2}u_i}{8\sqrt{N}}+\tfrac{\sqrt{2}}{4\sqrt{k}}.\end{aligned}$$

Then we use Lemma 2 in [59] to bound the penalty term as

$$\textstyle\sum_{k=1}^{K_i}(\frac{3\sqrt{2}u_i}{2\sqrt{N}}+\frac{\sqrt{2}}{\sqrt{k}})\sum_{n\neq n_i^*}(\sqrt{\mathbb{E}[p_{i,k,n}]}-\frac{1}{2}\mathbb{E}[p_{i,k,n}])+1.$$

Summing up these two bounds of the stability term and the penalty term, and noting that for all i,k,n, $\mathbb{E}[p_{i,k,n}]\leq\sqrt{\mathbb{E}[p_{i,k,n}]}$, we have

$$\begin{aligned}&R_i^T\leq\textstyle\sum_{k=1}^{K_i}((\frac{6\sqrt{2}u}{\sqrt{N}}+\frac{4\sqrt{2}}{\sqrt{k}})\sum_{n\neq n_i^*}(\sqrt{\mathbb{E}[p_{i,k,n}]}))\\&+\tfrac{3\sqrt{2}u_i}{2\sqrt{N}}K_{i,0}+2\sqrt{2K_{i,0}}+1.\end{aligned}$$

Then we use the self-bounding technique [29], which states that if $L\leq R\leq U$, then $R\leq 2U-L$. For the lower bound L, we use the following identity for the regret as

$$R_i^T=\textstyle\sum_{k=1}^{K_i}|B_{i,k}|\sum_{n\neq n_i^*}\Delta_{i,n}\mathbb{E}\,[p_{i,k,n}],$$

where $B_{i,K}$ is truncated, so that $|B_1|+\ldots+|B_{i,K}|=T$. Using the previous expression for the upper bound U, we get

$$\begin{aligned}&R_i^T\leq\textstyle\sum_{k=1}^{K_i}(\frac{12\sqrt{2}u_i}{\sqrt{N}}+\frac{8\sqrt{2}}{\sqrt{k}})\sum_{n\neq n_i^*}(\sqrt{\mathbb{E}[p_{i,k,n}]})\\&-\textstyle\sum_{k=1}^{K_i}|B_{i,k}|\sum_{n\neq n_i^*}\Delta_{i,n}\mathbb{E}[p_{i,k,n}]+\frac{544u_i}{\sqrt{N}}+66=C_1.\end{aligned}$$

Step II: With previous results, we know $R_i^T\leq C_1$. Since the first term of $\sum_t\mathbb{E}[L^t_{i,J_i^t}+v_{i,J_i^t}]-T\cdot\mathbb{E}[L^t_{i,n_i^*}+v_{i,n_i^*}]$ and R_i^T are both solved by our Algorithm 1 and the second term selects the best arm n_i^* for all time slots, we have

$$\textstyle\sum_t\mathbb{E}[L^t_{i,J_i^t}+v_{i,J_i^t}]-T\cdot\mathbb{E}[L^t_{i,n_i^*}+v_{i,n_i^*}]=R_i^T\leq C_1.$$

Step III: Note that

$$\mathbb{E}(\tfrac{1}{D}\textstyle\sum_{i=1}^{D}X_i\,|D=d)=\mathbb{E}(\frac{1}{d}\sum_{i=1}^{d}X_i)=\frac{1}{d}\sum_{i=1}^{d}\mathbb{E}(X_i)=\mu$$

$$\mathbb{E}(\tfrac{1}{D}\sum_{i=1}^{D} X_i) = \mathbb{E}(\mathbb{E}(\tfrac{1}{D}\sum_{i=1}^{D} X_i \,|D)) = \mathbb{E}(\mu) = \mathbb{E}(X),$$

where $X_i, \forall i$ all have the mean μ, and D is a random integer, with D independent of X_i. We know $L_{i,J_i^t} = \frac{1}{M_i}\sum_{\nu=1}^{M_i} l_{J_i^t,\nu}$, and then we have

$$\mathbb{E}[L_{i,J_i^t}] = \mathbb{E}[\tfrac{1}{M_i}\sum_{\nu=1}^{M_i} l_{J_i^t,\nu}] = \mathbb{E}[l_{J_i^t}],$$

$$\mathbb{E}[L_{i,n_i^*}] = \mathbb{E}[\tfrac{1}{M_i}\sum_{\nu=1}^{M_i} l_{n_i^*,\nu}] = \mathbb{E}[l_{n_i^*}].$$

The regret of $\mathbb{P}_1$ for the edge i can be obtained as

$$\begin{aligned}&\mathrm{Reg}_{1,i}^T = \textstyle\sum_t \mathbb{E}[l_{J_i^t} + v_{i,J_i^t}] - T\cdot\mathbb{E}[l_{n_i^*} + v_{i,n_i^*}]\\ &= \textstyle\sum_t \mathbb{E}[L_{i,J_i^t} + v_{i,J_i^t}] - T\cdot\mathbb{E}[L_{i,n_i^*} + v_{i,n_i^*}] \le C_1.\end{aligned}$$

Considering $\sum_t u_i\bar{y}_i^t$ and using the techniques in [59], we can optimize the bound and obtain the result in Theorem 1.

B. Proof of Theorem 2

Our theorem relies on a group of common assumptions [32], [33] that are widely adopted and easily satisfied: (1) The function $f^t(\mathbf{Z})$ has bounded gradients on $\mathcal{X}$, i.e., $\|\nabla f^t(\mathbf{Z})\| \le F$, $\forall \mathbf{Z} \in \mathcal{X}$; and $g^t(\mathbf{Z})$ is bounded on $\mathcal{X}$, i.e., $\|g^t(\mathbf{Z})\| \le G, \forall \mathbf{Z} \in \mathcal{X}, \forall t$; (2) The radius of the convex feasible set $\mathcal{X}$ is bounded, i.e., $\|\mathbf{Z}_1 - \mathbf{Z}_2\| \le R, \forall \mathbf{Z}_1, \mathbf{Z}_2 \in \mathcal{X}$; (3) There exists a constant $\delta > 0$ and an interior point $\widehat{\mathbf{Z}} \in \mathcal{X}$, such that $g^t(\widehat{\mathbf{Z}}^t) \le -\delta\mathbf{1}, \forall t$; (4) The slack constant δ satisfies $\delta > V(\mathbf{g})$, where the point-wise maximal variation of the consecutive constraints is denoted as $V(\mathbf{g}) = \max_t \max_{\mathbf{Z}\in\mathcal{X}} \|[g^{t+1}(\mathbf{Z}) - g^t(\mathbf{Z})]^+\|$.

We bound the regret and the fit of $\mathbb{P}_2$ [32]. By setting proper step sizes, we express these bounds as sub-linear functions of T. Setting $\gamma = \eta = \max\{\sqrt{\frac{V(\{\widetilde{\boldsymbol{x}}^{t*}\}_{t=1}^T)}{T}}, \sqrt{\frac{V(\{\mathbf{g}^t\}_{t=1}^T)}{T}}\}$, we have

$$\mathrm{Reg}^T = \mathcal{O}(\max\{\sqrt{V(\{\widetilde{\boldsymbol{x}}^{t*}\}_{t=1}^T)T}, \sqrt{V(\{\mathbf{g}^t\}_{t=1}^T)T}\}),$$

$$\mathrm{Fit}^{\mathrm{T}} \le \frac{\|\bar{\lambda}\|}{\eta} = \mathcal{O}(\max\{\tfrac{T}{V(\{\widetilde{\boldsymbol{x}}^{t*}\}_{t=1}^T)}, \tfrac{T}{V(\{\mathbf{g}^t\}_{t=1}^T)}\}).$$

Afterwards, if we further set $\gamma = \eta = \mathcal{O}(T^{-\frac{1}{3}})$, we have $\mathrm{Reg}^T = \mathcal{O}(\max\{V(\{\widetilde{\boldsymbol{x}}^{t*}\}_{t=1}^T)T^{\frac{1}{3}}, V(\{\mathbf{g}^t\}_{t=1}^T)T^{\frac{1}{3}}, T^{\frac{2}{3}}\})$ and $\mathrm{Fit}^{\mathrm{T}} = \mathcal{O}(T^{\frac{2}{3}})$. The sub-linear regret and the sub-linear fit of $\mathcal{O}(T^{\frac{2}{3}})$ can be achieved if we have $V(\{\widetilde{\boldsymbol{x}}^{t*}\}_{t=1}^T) \in \boldsymbol{o}(T^{\frac{2}{3}})$ and $V(\{\mathbf{g}^t\}_{t=1}^T) \in \boldsymbol{o}(T^{\frac{2}{3}})$.

C. Proof of Theorem 3

The regret of $\mathbb{P}_0$ is $regret = \overline{\mathcal{P}} - \mathcal{P}^* = \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \bar{z}^t, \bar{w}^t) - \mathcal{P}(x_{i,n}^{t*}, y_i^{t*}, z^{t*}, w^{t*})$, where $\bar{x}_{i,n}^t, \bar{y}_i^t, \bar{z}^t, \bar{w}^t$ are decisions obtained from our proposed algorithms and $x_{i,n}^{t*}, y_i^{t*}, z^{t*}, w^{t*}$ are the optimal decisions of $\mathbb{P}_0$. The regret can be split as

$$\begin{aligned}regret = {}& \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \bar{z}^t, \bar{w}^t) - \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \hat{z}^{t*}, \hat{w}^{t*})\\ &+ \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \hat{z}^{t*}, \hat{w}^{t*}) - \mathcal{P}(x_{i,n}^{t*}, y_i^{t*}, \hat{z}^{t*}, \hat{w}^{t*})\\ &+ \mathcal{P}(x_{i,n}^{t*}, y_i^{t*}, \hat{z}^{t*}, \hat{w}^{t*}) - \mathcal{P}(x_{i,n}^{t*}, y_i^{t*}, z^{t*}, w^{t*}),\end{aligned}$$

where $\hat{z}^{t*}, \hat{w}^{t*}$ are the optimums given $\bar{x}_{i,n}^t, \bar{y}_i^t$ as inputs.

The first two terms in $regret$ imply the difference on the objective under various z^t, w^t, which is exactly the objective of our subproblem $\mathbb{P}_2$, i.e., $\sum_t f^t(z^t, w^t) - \sum_t f^{t*} \le \mathcal{O}(T^{\frac{2}{3}})$. Then, we have

$$\mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \bar{z}^t, \bar{w}^t) - \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \hat{z}^{t*}, \hat{w}^{t*}) \le \mathcal{O}(T^{\frac{2}{3}}).$$

According to Theorem 1, we have

$$\begin{aligned}&\mathrm{Reg}_1^T + \textstyle\sum_t\sum_i u_i\bar{y}_i^t\\ &= \sum_{t,i,n}(\bar{x}_{i,n}^t - \tilde{x}_{i,n}^{t*})(\mathbb{E}_{l_n\sim\mathscr{D}_n}(l_n) + v_{i,n}) + \sum_{t,i} u_i(\bar{y}_i^t - \tilde{y}_i^{t*})\\ &= \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t) - \mathcal{P}(\tilde{x}_{i,n}^{t*}, \tilde{y}_i^{t*}) \le \mathcal{O}(T^{\frac{1}{3}} + \ln T),\end{aligned}$$

where $\tilde{x}_{i,n}^{t*}, \tilde{y}_i^{t*}$ are the optimal decisions of $\mathbb{P}_1$. We know $\tilde{x}_{i,n}^{t*} = \tilde{x}_{i,n}^*$ and $y_i^{t*} = \tilde{y}_i^{t*} = 0$ for all t. Further, we know $x_{i,n}^{t*}$ is the optimum of $\mathbb{P}_0$, which is the decision made subject to Constraint (1c), while $\tilde{x}_{i,n}^{t*}$ is the decision made in the entire feasible region. Thus, it is possible to find $\tilde{x}_{i,n}^{t*}$ such that

$$\sum_{t,i,n} \tilde{x}_{i,n}^{t*}(\mathbb{E}_{l_n\sim\mathscr{D}_n}(l_n) + v_{i,n}) \le \sum_{t,i,n} x_{i,n}^{t*}(\mathbb{E}_{l_n\sim\mathscr{D}_n}(l_n) + v_{i,n}),$$

and thus make $\mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t) - \mathcal{P}(x_{i,n}^{t*}, y_i^{t*}) \le \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t) - \mathcal{P}(\tilde{x}_{i,n}^{t*}, \tilde{y}_i^{t*})$ hold. Then, the next two terms in $regret$ satisfy

$$\begin{aligned}&\mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \hat{z}^{t*}, \hat{w}^{t*}) - \mathcal{P}(x_{i,n}^{t*}, y_i^{t*}, \hat{z}^{t*}, \hat{w}^{t*})\\ &\le \mathcal{P}(\bar{x}_{i,n}^t, \bar{y}_i^t, \hat{z}^{t*}, \hat{w}^{t*}) - \mathcal{P}(\tilde{x}_{i,n}^{t*}, \tilde{y}_i^{t*}, \hat{z}^{t*}, \hat{w}^{t*})\\ &\le \mathcal{O}(T^{\frac{1}{3}} + \ln T).\end{aligned}$$

For the last two terms in $regret$, we highlight that we denote $\mathcal{P}(x_{i,n}^{t*}, y_i^{t*}, \hat{z}^{t*}, \hat{w}^{t*}) - \mathcal{P}(x_{i,n}^{t*}, y_i^{t*}, z^{t*}, w^{t*}) = \Omega_1$, where Ω_1 is a constant. When the input data are given, i.e., for a specific instance of our problem, $\bar{x}_{i,n}^t$ and $\bar{y}_i^t$ are fixed values, which leads to $\hat{z}^{t*}$ and $\hat{w}^{t*}$ being fixed values. The difference between $\mathcal{P}(\hat{z}^{t*}, \hat{w}^{t*})$ and $\mathcal{P}(z^{t*}, w^{t*})$ is thus a constant. If we take the expectation over multiple repetitions, the result will still be a constant. Combining all of the above, we have

$$regret \le \mathcal{O}(T^{\frac{2}{3}}) + \mathcal{O}(T^{\frac{1}{3}} + \ln T) + \Omega_1.$$

REFERENCES

[1] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7457–7469, 2020.

[2] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738–1762, 2019.

[3] K. B. Letaief, Y. Shi, J. Lu, and J. Lu, "Edge artificial intelligence for 6g: Vision, enabling technologies, and applications," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 5–36, 2021.

[4] D. Patterson, J. Gonzalez, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. So, M. Texier, and J. Dean, "Carbon emissions and large neural network training," *arXiv preprint arXiv:2104.10350*, 2021.

[5] Z. Cao, X. Zhou, H. Hu, Z. Wang, and Y. Wen, "Toward a systematic survey for carbon neutral data centers," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 895–936, 2022.

[6] J. Carl and D. Fedor, "Tracking global carbon revenues: A survey of carbon taxes versus cap-and-trade in the real world," *Energy Policy*, vol. 96, pp. 50–77, 2016.

[7] "The beijing carbon emissions trading platform." [Online]. Available: https://www.bjets.com.cn/

[8] "Eu carbon permits." [Online]. Available: https://tradingeconomics.com/commodity/carbon

[9] "California cap-and-trade program." [Online]. Available: https://ww2.arb.ca.gov/our-work/programs/cap-and-trade-program

[10] J. P. Miller, R. Taori, A. Raghunathan, S. Sagawa, P. W. Koh, V. Shankar, P. Liang, Y. Carmon, and L. Schmidt, "Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization," in *ICML*, 2021.

[11] J. Steiger, B. Li, B. Ji, and N. Lu, "Constrained bandit learning with switching costs for wireless networks," in *IEEE INFOCOM*, 2023.

[12] M. Shi, X. Lin, and L. Jiao, "Power-of-2-arms for bandit learning with switching costs," in *ACM MobiHoc*, 2022.

[13] "Carbon future price." [Online]. Available: https://cn.investing.com/commodities/carbon-emissions

[14] "China carbon emissions permit trading." [Online]. Available: https://www.cneeex.com/zhhq/quotshown.html

[15] Y. Bai, L. Chen, M. Abdel-Mottaleb, and J. Xu, "Automated ensemble for deep learning inference on edge computing platforms," *IEEE Internet of Things Journal*, vol. 9, no. 6, pp. 4202–4213, 2021.

[16] K. Zhao, Z. Zhou, X. Chen, R. Zhou, X. Zhang, S. Yu, and D. Wu, "Edgeadaptor: Online configuration adaption, model selection and resource provisioning for edge dnn inference serving at scale," *IEEE Transactions on Mobile Computing*, vol. 22, no. 10, pp. 5870–5886, 2023.

[17] B. Lu, J. Yang, J. Xu, and S. Ren, "Improving qoe of deep neural network inference on edge devices: A bandit approach," *IEEE Internet of Things Journal*, vol. 9, no. 21, pp. 21 409–21 420, 2022.

[18] Y. Li, Z. Liu, Z. Kou, Y. Wang, G. Zhang, Y. Li, and Y. Sun, "Real-time adaptive partition and resource allocation for multi-user end-cloud inference collaboration in mobile environment," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 13 076–13 094, 2024.

[19] J.-A. Lim, J. Lee, J. Kwak, and Y. Kim, "Cutting-edge inference: Dynamic dnn model partitioning and resource scaling for mobile ai," *IEEE Transactions on Services Computing*, vol. 17, no. 6, pp. 3300–3316, 2024.

[20] Y. Jin, L. Jiao, Z. Qian, S. Zhang, N. Chen, S. Lu, and X. Wang, "Provisioning edge inference as a service via online learning," in *IEEE SECON*, 2020.

[21] S. Su, Z. Zhou, T. Ouyang, R. Zhou, and X. Chen, "Learning to be green: Carbon-aware online control for edge intelligence with colocated learning and inference," in *IEEE ICDCS*, 2023.

[22] H. Ma, Z. Zhou, X. Zhang, and X. Chen, "Toward carbon-neutral edge computing: Greening edge ai by harnessing spot and future carbon markets," *IEEE Internet of Things Journal*, vol. 10, no. 18, pp. 16 637–16 649, 2023.

[23] J. Bian, L. Wang, S. Ren, and J. Xu, "Cafe: Carbon-aware federated learning in geographically distributed data centers," in *ACM e-Energy*, 2024.

[24] C.-S. Yang, C.-C. Huang-Fu, and I.-K. Fu, "Carbon-neutralized task scheduling for green computing networks," in *IEEE GLOBECOM*, 2022.

[25] S. Zhang, M. Xu, W. Y. B. Lim, and D. Niyato, "Sustainable aigc workload scheduling of geo-distributed data centers: A multi-agent reinforcement learning approach," in *IEEE GLOBECOM*, 2023.

[26] Y. Huang, Q. Liu, and J. Xu, "Adversarial combinatorial bandits with switching cost and arm selection constraints," in *IEEE INFOCOM*, 2024.

[27] A. M. Appavoo, S. Gilbert, and K.-L. Tan, "Shrewd selection speeds surfing: Use smart exp3!" in *IEEE ICDCS*, 2018.

[28] Z. Zhu, T. Liu, Y. Yang, and X. Luo, "Blot: Bandit learning-based offloading of tasks in fog-enabled networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2636–2649, 2019.

[29] J. Zimmert and Y. Seldin, "Tsallis-inf: An optimal algorithm for stochastic and adversarial bandits," *Journal of Machine Learning Research*, vol. 22, no. 1, pp. 1310–1358, 2021.

[30] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, pp. 235–256, 2002.

[31] M. Mahdavi, R. Jin, and T. Yang, "Trading regret for efficiency: online convex optimization with long term constraints," *Journal of Machine Learning Research*, vol. 13, no. 1, pp. 2503–2528, 2012.

[32] T. Chen, Q. Ling, and G. B. Giannakis, "An online convex optimization approach to proactive network resource allocation," *IEEE Transactions on Signal Processing*, vol. 65, no. 24, pp. 6350–6364, 2017.

[33] A. Koppel, F. Y. Jakubiec, and A. Ribeiro, "A saddle point algorithm for networked online convex optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 19, pp. 5149–5164, 2015.

[34] E. C. Hall and R. M. Willett, "Online convex optimization in dynamic environments," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 647–662, 2015.

[35] L. Pu, J. Shi, X. Yuan, X. Chen, L. Jiao, T. Zhang, and J. Xu, "Ems: Erasure-coded multi-source streaming for uhd videos within cloud native 5g networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 2, pp. 1472–1487, 2023.

[36] S. Paternain and A. Ribeiro, "Online learning of feasible strategies in unknown environments," *IEEE Transactions on Automatic Control*, vol. 62, no. 6, pp. 2807–2822, 2016.

[37] T. Chen, Q. Ling, Y. Shen, and G. B. Giannakis, "Heterogeneous online learning for "thing-adaptive" fog computing in iot," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4328–4341, 2018.

[38] X. Li, R. Zhou, Y.-J. A. Zhang, L. Jiao, and Z. Li, "Smart vehicular communication via 5g mmwaves," *Computer Networks*, vol. 172, p. 107173, 2020.

[39] "Mnist database." [Online]. Available: https://yann.lecun.com/exdb/mnist/

[40] "Cifar-10 dataset." [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html

[41] "Our open data - transport for london," 2020. [Online]. Available: https://tfl.gov.uk/info-for/open-data-users/our-open-data/

[42] P. Lai, Q. He, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, and Y. Yang, "Optimal edge user allocation in edge computing with variable sized vector bin packing," in *ICSOC*, 2018.

[43] L. L. Zhang, S. Han, J. Wei, N. Zheng, T. Cao, Y. Yang, and Y. Liu, "Nn-meter: Towards accurate latency prediction of deep-learning model inference on diverse edge devices," in *ACM MobiSys*, 2021.

[44] Z. Zhou, F. Liu, R. Zou, J. Liu, H. Xu, and H. Jin, "Carbon-aware online control of geo-distributed cloud services," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 9, pp. 2506–2519, 2015.

[45] R. Desislavov, F. Martínez-Plumed, and J. Hernández-Orallo, "Compute and energy consumption trends in deep learning inference," *arXiv preprint arXiv:2109.05472*, 2021.

[46] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[47] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[48] T. Le, C. Szepesvari, and R. Zheng, "Sequential learning for multi-channel wireless network monitoring with channel switching costs," *IEEE Transactions on Signal Processing*, vol. 62, no. 22, pp. 5919–5929, 2014.

[49] L. Chamon and A. Ribeiro, "Probably approximately correct constrained learning," in *NeurIPS*, 2020.

[50] M. E. Ahsen and M. Vidyasagar, "An approach to one-bit compressed sensing based on probably approximately correct learning theory," *Journal of Machine Learning Research*, vol. 20, no. 11, pp. 1–23, 2019.

[51] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[52] X. Zhao, C. Gu, H. Zhang, X. Yang, X. Liu, J. Tang, and H. Liu, "Dear: Deep reinforcement learning for online advertising impression in recommender systems," in *AAAI*, 2021.

[53] R. P. Brent, *Algorithms for minimization without derivatives*. Courier Corporation, 2013.

[54] S. Mizuno and F. Jarre, "Global and polynomial-time convergence of an infeasible-interior-point algorithm using inexact computation," *Mathematical Programming*, vol. 84, no. 1, pp. 105–122, 1999.

[55] C. Xia, J. Zhao, H. Cui, X. Feng, and J. Xue, "Dnntune: Automatic benchmarking dnn models for mobile-cloud computing," *ACM Transactions on Architecture and Code Optimization*, vol. 16, no. 4, pp. 1–26, 2019.

[56] C. Mu, T. Ding, S. Zhu, O. Han, P. Du, F. Li, and P. Siano, "A decentralized market model for a microgrid with carbon emission rights," *IEEE Transactions on Smart Grid*, vol. 14, no. 2, pp. 1388–1402, 2022.

[57] X. Long, J. Wu, and L. Chen, "Energy-efficient offloading in mobile edge computing with edge-cloud collaboration," in *ICA3PP*, 2018.

[58] "The leader in decision intelligence technology." [Online]. Available: https://www.gurobi.com/

[59] C. Rouyer, Y. Seldin, and N. Cesa-Bianchi, "An algorithm for stochastic and adversarial bandits with switching costs," in *ICML*, 2021.