A Theoretical Framework for Auxiliary-Loss-Free Load Balancing of Sparse Mixture-of-Experts in Large-Scale AI Models

X.Y. Han[*]
Chicago Booth
XY.Han@chicagobooth.edu

Yuan Zhong[*]
Chicago Booth
Yuan.Zhong@chicagobooth.edu

December 3, 2025

Abstract

In large-scale AI training, Sparse Mixture-of-Experts (s-MoE) layers enable scaling by activating only a small subset of experts per token. An operational challenge in this design is load balancing: routing tokens to minimize the number of idle experts, which is important for the efficient utilization of (costly) GPUs. We provide a theoretical framework for analyzing the Auxiliary-Loss-Free Load Balancing (ALF-LB) procedure — proposed by DeepSeek's Wang et al. (2024) — by casting it as a one-step-per-iteration primal-dual method for an assignment problem. First, in a stylized deterministic setting, our framework yields several insightful structural properties: (i) a monotonic improvement of a Lagrangian objective, (ii) a preference rule that moves tokens from overloaded to underloaded experts, and (iii) an approximate-balancing guarantee. Then, we incorporate the stochastic and dynamic nature of AI training using a generalized online optimization formulation. In the online setting, we derive a strong convexity property of the objective that leads to a logarithmic expected regret bound under certain step-size choices. Additionally, we present real experiments on 1B-parameter DeepSeekMoE models to complement our theoretical findings. Together, these results build a principled framework for analyzing the Auxiliary-Loss-Free Load Balancing of s-MoE in AI models.

1 Introduction: s-MoEs and Load Balancing in AI Training

Scaling architecture size has been the dominant driver of modern AI performance, with larger models consistently achieving better results (Kaplan et al., 2020; Hoffmann et al., 2022; Epoch AI, 2023). However, AI development has reached the point where the scaling of computation becomes prohibitively expensive due to hardware and energy constraints (Strubell et al., 2019; Thompson et al., 2020; Sevilla et al., 2022). Adapting to the cost and hardware limitations of scaling, researchers

[*] Authors listed alphabetically.

have turned to sparse Mixture-of-Experts (s-MoE) architectures (Shazeer et al., 2017), which are sparse realizations within the mixture-of-experts (MoE) paradigm codified by Jacobs et al. (1991).

In modern large-scale AI architectures, s-MoE layers — consisting of several parallel subnetworks ("experts") controlled by a "sparse gate" or *router* that selects data to route to them — have largely replaced single submodules through which all data must pass. In these s-MoEs, for each input, the sparse-gating component selects a strict subset of experts (hence "sparse") to apply to that input. Thus, only a small subcomponent of an AI architecture is activated to process each piece of input data — allowing models to have significantly more parameters while keeping inference and training costs manageable. As a testament to s-MoEs' utility, recent releases of OpenAI's GPT (Achiam et al., 2023), Google's Gemini (Gemini Team et al., 2024), and DeepSeek (DeepSeek-AI, 2025) have all leveraged s-MoE designs to improve efficiency and maintain performance scaling.

However, a crucial aspect of s-MoE design — load balancing (controlling "how many inputs per expert") — is mostly developed using trial-and-error motivated by heuristic insights (see Section 1.2). Learning to precisely and mathematically balance the load across experts, which reduces monetary losses from idle GPUs, could lead to enormous monetary savings for AI training.

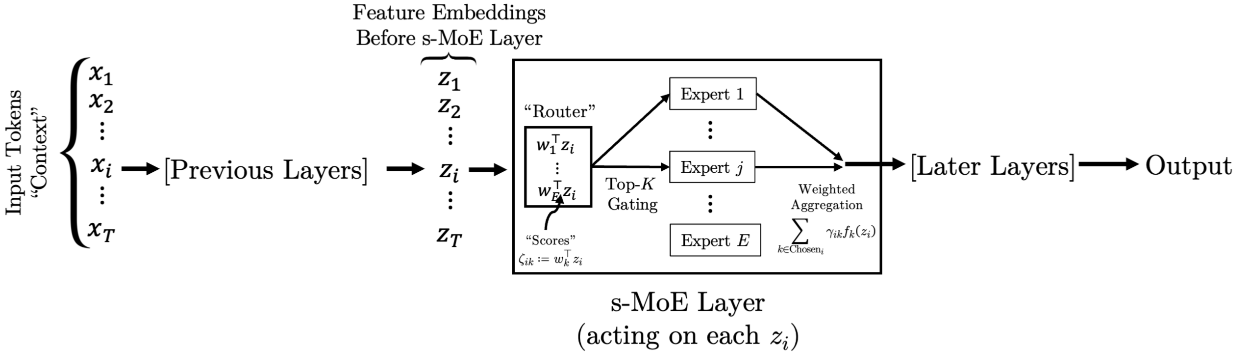


Figure 1: Schematic of a naïve s-MoE layer without load balancing.

1.1 Naïve s-MoE Layers Without Load Balancing

Figure 1 describes the "naïve" setup for s-MoE layers within transformer-based AI models. In particular, the input is a series of *token embeddings* $x_1, x_2, ..., x_T$ where each x_i is a high-dimensional vector corresponding (in language models) to a language unit such as "Hel", "lo", "world", etc. or (in vision models) a patch within an image. Each piece of input data (a sentence, an image patch, etc.) is decomposed into constituent tokens; each token is mapped to its vector embedding x_i; and those embeddings are input into the AI model. The entire tuple of vectors $\{x_i\}_{i=1}^T$ is called the *context* and T is the *context length*.

Within the AI model, each of the original token embeddings x_i is transformed into *feature embeddings* by each of the AI model's layers. In Figure 1, to describe the action of some particular s-MoE layer, we use $\{z_i\}_{i=1}^T$ to denote the feature embeddings before that s-MoE layer.

When a feature embedding "enters" an s-MoE layer with E experts, we calculate an unnormalized affinity score $\zeta_{i,k}$ between z_i and the k-th expert — usually using an inner product: $\zeta_{i,k} := w_k^\top z_i$. These scores are then normalized, typically using the "softmax" function, into the *affinity scores*:

$$\gamma_{i,k} := \text{SoftMax}\left(\zeta_{i,k};\ \{\zeta_{i,k'}\}_{k'=1}^{E}\right) = \frac{\exp(\zeta_{i,k})}{\sum_{k'=1}^{E}\exp(\zeta_{i,k'})}$$

The router then selects the Top-K experts based on the K largest $\gamma_{i,k}$. The final step in an s-MoE layer is to aggregate the outputs of the selected experts. This is done by computing a weighted sum of the selected experts' outputs:

$$\sum_{k \in \text{ChosenExperts}_i} \gamma_{i,k} f_k(z_i), \tag{1}$$

where f_k represents the k-th expert. Note that the softmax is taken *before* the Top-K selection, which is typically the preferred order in recent s-MoEs (Dai et al., 2024; Riquelme et al., 2021). Moreover, the softmax is monotonic, so it is equivalent to choose the Top-K experts based on the K largest $\{\gamma_{i,k}\}_{k=1}^{E}$ for each i. This completes the description of the schematic in Figure 1.

1.2 Load Balancing of Experts: Background and Related Work

While the naïve routing method of choosing the top-K among $\{\gamma_{i,k}\}_{k=1}^{E}$ is conceptually simple, it could easily cause some experts to idle while others are overloaded. Since GPU time is expensive, such an imbalance could lead to significant monetary losses and inefficiencies in the training process.

Several fixes have been proposed. The most commonly adopted approach is adding an auxiliary "balancing loss" directly to the training loss penalizing the network parameters during training for inducing imbalanced token allocations (Fedus et al., 2022; Lepikhin et al., 2021; Shazeer et al., 2017). However, this method interferes with the gradient updates of the performance-focused component of the objective (see Section 2.2 of Wang et al. (2024) for a more detailed discussion).

Another approach by Lewis et al. (2021) approximately solves — via a truncated auction algorithm based on Bertsekas (1992) — an integer program that balances the load across experts in every training iteration (corresponding to one mini-batch of data). However, generating an AI model's outputs for even one single batch of data (a "forward pass") requires significant computation time and memory since it requires calculating matrix multiplications and non-linear transformations defined by millions to billions of parameters. If this is done during training, there is an additional computational and memory overhead for computing and storing the backpropagated gradients (the "backward pass"). Thus, it is inadvisable to spend additional time solving a multi-iterative subroutine (whether an auction algorithm or an integer program) for every s-MoE layer and every mini-batch.

To address this problem, DeepSeek's auxiliary-loss-free (ALF-LB) (Wang et al., 2024) procedure augments each expert with a bias p_k using a *single-shot* update (as opposed to a multi-step subroutine), nudging tokens toward underloaded experts — without interfering with training gradients as is done in works leveraging auxiliary balancing losses. Notably, ALF-LB was used to successfully train the recent DeepSeekV3 (DeepSeek-AI, 2024) models.

1.3 DeepSeek's ALF-LB Algorithm

The ALF-LB procedure (Wang et al., 2024) is as follows:

1. For each expert $k = 1, \dots, E$, initialize a scalar shift parameter p_k to be 0.
2. Perform a forward pass on a mini-batch. During the forward pass, route token i based on the experts with the highest shifted weights $\gamma_{ik} + p_k$.
3. Calculate the downstream network loss and update the main network parameters, treating the shifts $\{p_k\}$ as constants.
4. For each expert k, update its shift parameter as follows, where u is a small constant (e.g., 0.001):

$$p_k \leftarrow \begin{cases} p_k - u & \text{if expert } k \text{ had load } > L; \\ p_k + u & \text{if expert } k \text{ had load } < L; \\ p_k & \text{otherwise.} \end{cases} \tag{2}$$

5. Repeat steps 2-4 for each mini-batch of input data.

In the original publication, Wang et al. (2024) chose $u = 0.001$ and exhibited empirical benefits of this procedure on 1B to 3B parameter DeepSeekMoE models (Dai et al., 2024).

1.4 Contributions and Organization of Paper

Our main contribution is a rigorous theoretical framework for understanding and analyzing the ALF-LB procedure, with specific contributions detailed across different sections. First, in Section 2, we cast the ALF-LB procedure as a single-step primal-dual method for an assignment problem, connecting a state-of-the-art heuristic from large-scale AI to the operations research and primal-dual optimization literature for resource allocation such as those in Bertsekas (1992, 1998, 2008). However, the procedure we analyze differs from the aforementioned operations research problems since, as discussed in Section 1.2, the computational and memory requirements of performing a forward pass through an AI model do not allow for one to run multi-iterative procedures as subroutines with those forward passes. Instead, s-MoE balancing routines (such as ALF-LB) must be updated in a "one-shot" manner — with computationally-minimal, constant-time updates per forward pass.

Then, in Section 4, we analyze this procedure in a stylized deterministic setting and establish several insightful structural properties: (i) a monotonic improvement of a Lagrangian objective (Theorem 1), (ii) a preference rule that moves tokens from overloaded to underloaded experts (Theorem 2), and (iii) a guarantee that expert loads converge to a bounded deviation from perfect balance. Finally, in Section 5, we extend our analysis to a more realistic online, stochastic setting by establishing a strong convexity property of the expected dual objective (Section 5.6) and using it to derive a logarithmic regret bound for the ALF-LB procedure (Theorem 13).

1.4.1 Related Work in Online Resource Allocation

It is insightful to compare this paper to another recent line of work at the intersection of AI implementation and operations research: the online resource allocation in AI infrastructures (see Zhang et al. (2024) and citations therein) where computational jobs arrive in an online, stochastic manner at, say, a large data center and must be optimally scheduled to the best server for the job. In comparison, in the s-MoE balancing problem, for each forward pass, every s-MoE layer must process batches of tokens that *all arrive at once*. These tokens must all pass through that s-MoE layer before they can collectively move to the next s-MoE layer — an example for reference would be DeepSeekV3 (DeepSeek-AI, 2024), which contains 61 s-MoE layers. Thus, unlike the case studied in Zhang et al. (2024), the allocation in MoE load balancing must be done immediately and in a computationally-minimal manner. (See Section 1.2 for additional details.)

Another related line of works is Balseiro et al. (2020, 2021); Agrawal and Devanur (2014); Jenatton et al. (2016) (see Balseiro et al. (2021, Section 1.2) and citations therein) that design and analyze primal-dual methods for solving online resource allocation problems by formulating them as online stochastic convex programs or regularized allocation problems. These prior works utilize dual descent and mirror descent techniques to manage global resource constraints which can be formulated as load balancing. However, the algorithms proposed in those works often require solving auxiliary optimization sub-routines (such as linear programs or non-trivial projections) during their updates. As mentioned in Section 1.2, in the context of s-MoE training, multi-iterative subroutines are computationally impracticable because routing must occur in every s-MoE layer of the AI architecture during already-computationally-expensive forward passes, which does not allow for the extra overhead of solving auxiliary sub-routines at every s-MoE layer. Thus, in comparison, our paper instead analyzes a "single-shot" update framework, built specially to encompass DeepSeek's ALF-LB procedure (Wang et al., 2024), that updates the load balancing parameters with negligible effect on the speed of the forward pass.

2 A Primal-Dual Framework for Optimal Load Balancing

We establish a rigorous mathematical framework to understand existing load balancing heuristics, particularly the DeepSeek ALF-LB method. In the remainder of the paper, for simplicity, we will refer to the normalized affinity scores γ_{ik} as the "affinity scores" and adopt the convention of using them both for routing and aggregation.

2.1 Allocation Problem: Integer Program and Relaxation

Consider the exact-balancing primal problem for assigning T tokens to E experts. As a starting point, we make the following assumptions and stylizations:

- The number of tokens multiplied by the sparsity, KT, is exactly divisible by the number of experts E, so the perfectly balanced load is $L = KT/E$.

- The affinity scores γ_{ik} are constant from iteration to iteration[1].

Hence, the target load is $L := KT/E$ and perfect balance is characterized by the solution of the following integer program (IP):

$$\begin{aligned} \max_{\{x_{ik}\}} \ & \sum_{i,k} \gamma_{ik} x_{ik} \\ \text{s.t.} & \sum_{k} x_{ik} = K \quad \forall i = 1, \dots, T \\ & \sum_{i} x_{ik} = L \ \forall k = 1, \dots, E \\ & x_{ik} \in \{0,1\} \quad \forall i, k. \end{aligned} \tag{3}$$

In practice, it is typically inadvisable (in terms of both time and memory requirements) to solve an IP for every MoE layer and on each individual mini-batch of data[2].

Instead, we first relax the IP to a linear program (LP) by replacing the integer constraint $x_{ik} \in \{0,1\}$ with $x_{ik} \geq 0$. It is routine to show that the IP and the LP relaxation have the same optimal value. The Lagrangian of the LP relaxation is

$$\begin{aligned} \mathcal{L}(x, y, p) &= \sum_{i,k} \gamma_{ik} x_{ik} + \sum_{i} y_i \left(K - \sum_{k} x_{ik} \right) + \sum_{k} p_k \left(\sum_{i} x_{ik} - L \right) \\ &= \sum_{i,k} \left(\gamma_{ik} + p_k - y_i \right) x_{ik} + K \sum_{i} y_i - L \sum_{k} p_k. \end{aligned} \tag{4}$$

The corresponding dual problem is

$$\begin{aligned} \min_{\{y_i\},\{p_k\}} \ & \sum_{i} y_i - L \sum_{k} p_k \\ \text{s.t.} \ & y_i - p_k \geq \gamma_{ik} \quad \forall i, k. \end{aligned}$$

However, even solving this LP relaxation to completion every iteration would still be too slow and often memory-infeasible.

2.2 Deriving ALF-LB from Primal-Dual Principles

We show that DeepSeek ALF-LB (Section 1.3) can be formulated as a primal-dual procedure that performs a single-shot update per iteration for finding a critical point of the Lagrangian (4). For conciseness, we introduce the following notation for the *load* of the k-th expert at iteration n:

[1] This is a stylized assumption for the initial analysis in this section only. Later, in Section 5, we will consider the case where the affinity scores are new stochastic realizations from some distribution every iteration.

[2] One notable exception is the BASE layer heuristic invented by Lewis et al. (2021) which aims to approximately solve the IP using a truncated auction algorithm modeled after Bertsekas (1992).

$A_k^{(n)} := \sum_i x_{ik}^{(n)}$. Indexing each training iteration with n, consider the following primal-dual scheme:

$$\textbf{Dual Update: } p_k^{(n+1)} \leftarrow p_k^{(n)} + \epsilon_k^{(n)} \left(L - A_k^{(n)}\right) \qquad \forall\, k. \tag{5}$$

$$\textbf{Primal Update: } x_{ik}^{(n+1)} \leftarrow \begin{cases} 1 & \text{if } k \in \mathrm{TopKInd}_{k'}\left(\gamma_{ik'}^{(n+1)} + p_{k'}^{(n+1)}\right) \\ 0 & \text{otherwise} \end{cases} \qquad \forall\, i, k, \tag{6}$$

where $\left\{\epsilon_k^{(n)}\right\}$ are step-sizes and $\mathrm{TopKInd}_{k'}(\cdot)$ gives the indices that would induce the K-largest arguments. The Primal Update enforces $\sum_k x_{ik} = K$. Maximizing $\sum_{i,k}(\gamma_{ik} + p_k - y_i)x_{ik}$ subject to this constraint is equivalent to choosing the top K values of $\gamma_{ik} + p_k$ for each i, regardless of y_i. Thus we can simplify the Lagrangian by dropping the y_i terms, which gives:

$$\mathcal{L}(x, p) = \sum_{i,k} (\gamma_{ik} + p_k)\, x_{ik} - L \sum_k p_k. \tag{7}$$

Within this setup, the original DeepSeek ALF-LB update from Wang et al. (2024) described in (2) corresponds to the step-size

$$\epsilon_k^{(n)} = \frac{u}{\left|L - A_k^{(n)}\right|}. \qquad \textbf{(DeepSeek ALF-LB Step-Size)} \tag{8}$$

Figure 2 illustrates the convergence behavior of this primal-dual scheme during the training of real 1B-parameter DeepSeekMoE models (Dai et al., 2024) with varying $\epsilon_k^{(n)}$ step-size choices. More experimental details are provided in Section 3.

3 Experimental Setup and Observations

3.1 Experimental Setup

In all experiments in this paper (Figures 2-4), we train 1B-parameter DeepSeekMoE models (Dai et al., 2024) for 100K steps on the next-token prediction task on the Salesforce WikiText-103 dataset (Merity et al., 2016) with the cross-entropy loss. The text data is tokenized using the GPT-2 tokenizer (Radford et al., 2019).

Here, we will provide only a brief description of the DeepSeekMoE architecture for completeness and refer to Dai et al. (2024) for more in-depth details: The DeepSeekMoE architecture follows the paradigmatic practice of stacking decoder-only transformer layers (Vaswani et al., 2017) into a full large language model. In its simplest form, the transformer layer contains several sub-layers — among them a multi-headed attention sub-layer and a multi-layer perceptron (MLP) sub-layer. For our setting of interest, modern s-MoE architectures (Shazeer et al., 2017; Jiang et al., 2024; Dai et al., 2024) replace the MLP sub-layer of each transformer layer with an s-MoE sub-layer described in Sections 1.1-1.2, where each parallel expert is typically a separate MLP. Additionally, the DeepSeekMoE architecture (Dai et al., 2024) is specifically characterized by its use of "granular

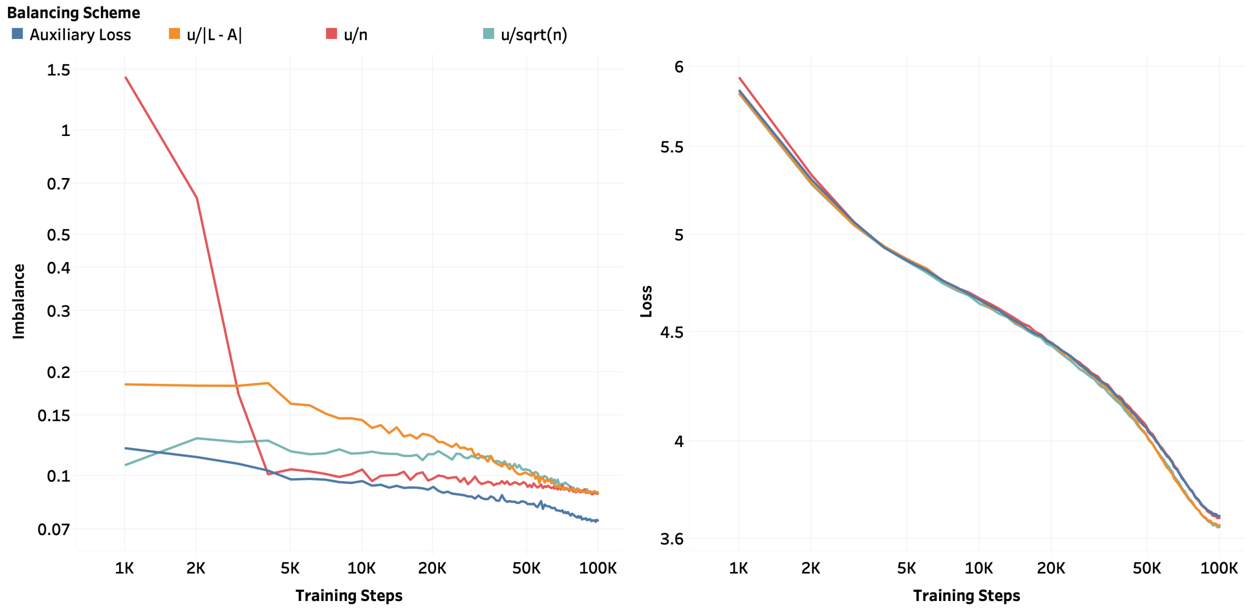


Figure 2: Validation set load imbalance and loss during the training of a 1B-parameter DeepSeekMoE model. Section 3 gives experiment details. *Left:* We measure the imbalance as the average load deviation from the target load $L = KT/E$ across all experts in the DeepSeekMoE-1B architecture. *Right:* We measure the loss on the validation set.

segmentation" (using narrower experts but increasing the total number of experts) and the inclusion of two "shared experts" that are always chosen by the gate[3].

The architectural parameters of the 1B-parameter DeepSeekMoE models in our experiments are the same as those described in Wang et al. (2024, Table 5). For consistency with Wang et al. (2024), we also use $E = 64$ experts with sparsity level $K = 6$. During training, we optimize all 1B parameters within the transformer backbone and prediction head of the DeepSeekMoE architectures starting from random initializations. Each model was trained on 8xH100/H200 GPUs with a batch size of 64 sequences/batch and 4096 tokens/sequence (so, $T \approx 262K$). To optimize the models, we use the AdamW (Loshchilov and Hutter, 2019) optimizer.

Balancing Schemes. In our experiments, we compare three choices of the k-th expert step-sizes at iteration n (denoted $\epsilon_k^{(n)}$, see Section 2.2) in the ALF-LB balancing scheme framework. In particular, given some balancing hyperparameter u, we compare the following schemes:

- $\epsilon_k^{(n)} = \frac{u}{L - A_k^{(n)}}$ (Original DeepSeek ALF-LB from Wang et al. (2024))
- $\epsilon_k^{(n)} = \frac{u}{n}$
- $\epsilon_k^{(n)} = \frac{u}{\sqrt{n}}$

Additionally, we include a comparison with a fourth scheme that trains with an auxiliary loss

[3] We will not include the shared experts within the theoretical framework presented in this paper because the shared experts represent a fixed computational load that does not require dynamic balancing. Additionally, omitting the shared experts from our theoretical formulation leads to cleaner and more concise analyses.

(Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022; Wang et al., 2024). We calculate the auxiliary loss with the method described in Wang et al. (2024, Section 2.2). The auxiliary loss is multiplied by a "trade-off parameter" that we will, for consistency, also denote by u and then added to the main cross-entropy loss.

Hyperparameter Search. For each of the four scheduling schemes, we conducted hyperparameter search over the following hyperparameters:

- balancing constants $u \in \{1e{-}4, 1e{-}3, 1e{-}2, 1e{-}1, 1, 10\}$,
- learning rates $\texttt{lr} \in \{1e{-}5, 1e{-}4, 1e{-}3\}$, and
- weight decay $\texttt{wd} \in \{0.01, 0.1, 0.001\}$.

Thus, we trained $4 \times 6 \times 3 \times 3 = 216$ separate 1B-parameter DeepSeekMoE models to conduct this search. Then, for each of the four scheduling schemes, we select the hyperparameter setting that achieves the best cross-entropy loss on a held-out validation set to be shown in the experimental plots and tables in this paper. We found that

- $\texttt{lr} = 1e{-}4$ and $\texttt{wd} = 1e{-}1$ consistently led to the best validation loss across all settings;
- the u/n and auxiliary loss scheduling schemes performed the best with parameter $u = 1$; and
- the $u/|L - A_k^{(n)}|$ and $u/\sqrt{n}$ scheduling schemes performed the best with $u = 1e{-}3$.

3.2 Experimental Observations

We make some interesting empirical observations from our experiments that are of separate interest from the theoretical framework proposed in this paper.

Firstly, we found that, for the original "constant update" scheme considered by Wang et al. (2024) (which corresponds to $u/|L - A_k^{(n)}|$ in our formalization), our hyperparameter search also yielded $u = 1e{-}3$ to be the optimal balancing constant, which corroborates the same observation from Wang et al. (2024).

Secondly, Table 1 reports the final validation loss and overall imbalance of the different balancing schemes at the end of training. Observe that the $u/\sqrt{n}$ scheme achieves the lowest validation loss (best predictive performance) but the highest imbalance (worst computational efficiency); in contrast, the auxiliary loss approach (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022) achieves the lowest imbalance (best computational efficiency) but the highest validation loss (worst predictive performance). The u/n scheme (which we will analyze in Section 5 through the lens of online optimization) and Wang et al. (2024)'s original $u/|L - A_k^{(n)}|$ scheme achieve a balance between validation loss and imbalance — with u/n achieving slightly better balance and $u/|L - A_k^{(n)}|$ achieving slightly better predictive performance.

4 Convergence Analysis for the Deterministic Case

To start, we present theoretical guarantees for the convergence of the procedure described by (5) and (6), assuming fixed scores γ_{ik}. For simplicity, we consider the case where $K = 1$ for Section 4

Table 1: Comparison of cross-entropy loss on validation data and overall imbalance at the end of training for different scheduling schemes. Experiment details in Section 3.

Balancing Scheme	**Validation Loss**	**Overall Imbalance**
Auxiliary Loss	3.68999	**0.07443**
$u/\lvert L - A_k^{(n)}\rvert$	3.65369	0.08928
u/n	3.68228	0.08893
$u/\sqrt{n}$	**3.64642**	0.08961

only. We will later consider the case where the scores are stochastic and for general K in Section 5.

4.1 Monotonicity of the Lagrangian

Towards showing the convergence of this procedure, we will show that the Lagrangian decreases with every iteration. Additionally, we define the *assignment function* $\alpha_n(i) := \arg\max_{k'}\left(\gamma_{ik'} + p_{k'}^{(n)}\right)$ that gives the assigned expert to token i at iteration n. The *switching benefit* is defined as

$$b^{(n+1)}(i) = \left(\gamma_{i\alpha_{n+1}(i)} + p_{\alpha_{n+1}(i)}^{(n+1)}\right) - \left(\gamma_{i\alpha_n(i)} + p_{\alpha_n(i)}^{(n+1)}\right), \tag{9}$$

which captures the benefit of switching to expert $\alpha_{n+1}(i)$ rather than staying with $\alpha_n(i)$. Note that $b^{(n+1)}(i) \geq 0$ by definition of the primal update.

Theorem 1. ***(Change in Lagrangian)*** *Using the procedure described in Steps (5)-(6) (with $K = 1$), the following holds for the Lagrangian (7):*

$$\mathcal{L}\left(x^{(n+1)}, p^{(n+1)}\right) - \mathcal{L}\left(x^{(n)}, p^{(n)}\right) = \sum_i b^{(n+1)}(i) - \sum_k \epsilon_k^{(n)}\left(A_k^{(n)} - L\right)^2.$$

Proof. By analyzing the change in the two terms of the Lagrangian (7) over one iteration, we have:

$$\begin{aligned}
&\mathcal{L}\left(x^{(n+1)}, p^{(n+1)}\right) - \mathcal{L}\left(x^{(n)}, p^{(n)}\right)\\
&\quad= \sum_{i,k}\left(\gamma_{ik} + p_k^{(n+1)}\right)x_{ik}^{(n+1)} - \sum_{i,k}\left(\gamma_{ik} + p_k^{(n)}\right)x_{ik}^{(n)} - L\sum_k(p_k^{(n+1)} - p_k^{(n)})\\
&\quad= \sum_i\left[\left(\gamma_{i\alpha_{n+1}(i)} + p_{\alpha_{n+1}(i)}^{(n+1)}\right) - \left(\gamma_{i\alpha_n(i)} + p_{\alpha_n(i)}^{(n)}\right)\right] - L\sum_k \epsilon_k^{(n)}(L - A_k^{(n)})\\
&\quad= \sum_i\left[b^{(n+1)}(i) + \left(\gamma_{i\alpha_n(i)} + p_{\alpha_n(i)}^{(n+1)}\right) - \left(\gamma_{i\alpha_n(i)} + p_{\alpha_n(i)}^{(n)}\right)\right] - L\sum_k \epsilon_k^{(n)}(L - A_k^{(n)})\\
&\quad= \sum_i b^{(n+1)}(i) + \sum_i\left(p_{\alpha_n(i)}^{(n+1)} - p_{\alpha_n(i)}^{(n)}\right) - L\sum_k \epsilon_k^{(n)}(L - A_k^{(n)})\\
&\quad= \sum_i b^{(n+1)}(i) + \sum_i \epsilon_{\alpha_n(i)}^{(n)}(L - A_{\alpha_n(i)}^{(n)}) - L\sum_k \epsilon_k^{(n)}(L - A_k^{(n)})\\
&\quad= \sum_i b^{(n+1)}(i) + \sum_k A_k^{(n)}\epsilon_k^{(n)}(L - A_k^{(n)}) - L\sum_k \epsilon_k^{(n)}(L - A_k^{(n)})
\end{aligned}$$

$$= \sum_i b^{(n+1)}(i) + \sum_k \epsilon_k^{(n)}(A_k^{(n)} - L)(L - A_k^{(n)})$$

$$= \sum_i b^{(n+1)}(i) - \sum_k \epsilon_k^{(n)}(A_k^{(n)} - L)^2.$$

□

This theorem shows that the improvement in the Lagrangian is the difference between the total switching benefit and the squared sum of load imbalances (weighted by step-sizes). We now specialize the analysis to the DeepSeek ALF-LB step-size choice.

4.2 Analysis of the DeepSeek Step-Size

Theorem 2. *Assume (A) we use the procedure (5)-(6) with the DeepSeek step-size $\epsilon_k^{(n)} = u/|L - A_k^{(n)}|$, (B) token i switched experts between iterations n and $n+1$, and (C) there are no ties in bias-shifted scores. Then, the following must hold:*

1. *Token i must have switched to a strictly lower designation in the ordering: Overloaded $\succ$ Balanced $\succ$ Underloaded.*
2. *The switching benefit of token i is bounded: $0 < b_i^{(n+1)} < 2u$.*
3. *The iteration-n score difference is bounded: $-2u < \gamma_{i\alpha_{n+1}(i)} + p^{(n)}_{\alpha_{n+1}(i)} - \left(\gamma_{i\alpha_n(i)} + p^{(n)}_{\alpha_n(i)}\right) < 0$.*

Proof. Since scores are not tied, the switching benefit $b_i^{(n+1)}$ must be strictly positive, and $\gamma_{i\alpha_{n+1}(i)} + p^{(n)}_{\alpha_{n+1}(i)} < \gamma_{i\alpha_n(i)} + p^{(n)}_{\alpha_n(i)}$. From the definition of $b_i^{(n+1)}$ and the dual update, we have

$$b_i^{(n+1)} = \left(\gamma_{i\alpha_{n+1}(i)} + p^{(n)}_{\alpha_{n+1}(i)}\right) - \left(\gamma_{i\alpha_n(i)} + p^{(n)}_{\alpha_n(i)}\right) + u \cdot \text{Sign}\left(L - A^{(n)}_{\alpha_{n+1}(i)}\right) - u \cdot \text{Sign}\left(L - A^{(n)}_{\alpha_n(i)}\right).$$

For $b_i^{(n+1)} > 0$, it must be that $u \cdot \text{Sign}\left(L - A^{(n)}_{\alpha_{n+1}(i)}\right) - u \cdot \text{Sign}\left(L - A^{(n)}_{\alpha_n(i)}\right) > 0$. This only happens if token i moves from an expert that is more loaded (relative to L) to one that is less loaded, which implies $\text{Sign}\left(L - A^{(n)}_{\alpha_{n+1}(i)}\right) > \text{Sign}\left(L - A^{(n)}_{\alpha_n(i)}\right)$, proving (1). The sign difference term is either u or $2u$. The bounds in (2) and (3) follow from this observation and the strict inequalities. □

Corollary 3. *(DeepSeek Lagrangian) With the DeepSeek step-size, the Lagrangian change in Theorem 1 simplifies to*

$$\mathcal{L}\left(x^{(n+1)}, p^{(n+1)}\right) - \mathcal{L}\left(x^{(n)}, p^{(n)}\right) = \sum_i b^{(n+1)}(i) - u \sum_k \left|A_k^{(n)} - L\right|.$$

Furthermore, letting $\mathcal{S}^{(n+1)}$ be the set of tokens that switched experts,

$$\mathcal{L}\left(x^{(n+1)}, p^{(n+1)}\right) - \mathcal{L}\left(x^{(n)}, p^{(n)}\right) < u\left[2\left|\mathcal{S}^{(n+1)}\right| - \sum_k \left|A_k^{(n)} - L\right|\right].$$

Proof. Follows directly from combining Theorems 1 and 2, and using the strict inequality from the no-ties assumption. □

From this, we can show that if the imbalance pattern (which experts are overloaded vs. underloaded) does not "flip" between iterations, the Lagrangian strictly decreases.

Proposition 4. *Suppose that at iteration n, a set of experts K_1 has load $\geq L$ and a set K_2 has load $\leq L$. If at iteration $n+1$, the same sets of experts remain in their respective loading states (i.e., experts in K_1 still have load $\geq L$, and experts in K_2 still have load $\leq L$), then*

$$\mathcal{L}\left(x^{(n+1)}, p^{(n+1)}\right) - \mathcal{L}\left(x^{(n)}, p^{(n)}\right) < 0.$$

Proof. By Theorem 2, tokens can only switch from an expert in K_1 to an expert in K_2. This implies that $|\mathcal{S}^{(n+1)}| = \sum_{k \in K_1}(A_k^{(n)} - A_k^{(n+1)})$. Since $A_k^{(n+1)} \geq L$ for $k \in K_1$, we have $|\mathcal{S}^{(n+1)}| \leq \sum_{k \in K_1}(A_k^{(n)} - L)$. Also, we know that $2\sum_{k \in K_1}(A_k^{(n)} - L) = \sum_k |A_k^{(n)} - L|$. Thus, $2|\mathcal{S}^{(n+1)}| \leq \sum_k |A_k^{(n)} - L|$. The result then follows from Corollary 3. □

4.3 Preference Analysis and Guarantee of Balance

We can reframe the routing decision in terms of preferences. Token i prefers expert k' over k at iteration $n+1$ if $\gamma_{ik'} + p_{k'}^{(n+1)} > \gamma_{ik} + p_k^{(n+1)}$. Let the score gap be $\mathfrak{G}\gamma_{k'k}^i := \gamma_{ik'} - \gamma_{ik}$ and the bias gap be $\mathfrak{G}p_{kk'}^{(n+1)} := p_k^{(n+1)} - p_{k'}^{(n+1)}$. The preference is equivalent to $\mathfrak{G}\gamma_{k'k}^i > \mathfrak{G}p_{kk'}^{(n+1)}$.

Proposition 5. *Assume two tokens i and j concurrently switched from the same origin expert α_n to the same destination expert α_{n+1}. Then, their score gaps relative to those experts satisfy*

$$\left|\mathfrak{G}\gamma_{\alpha_{n+1}\alpha_n}^i - \mathfrak{G}\gamma_{\alpha_{n+1}\alpha_n}^j\right| < 2u.$$

Proof. For token i to switch, its score gap must lie in a specific interval determined by the bias gaps: $\mathfrak{G}p_{\alpha_n\alpha_{n+1}}^{(n)} + \texttt{sign\_diff} \cdot u < \mathfrak{G}\gamma_{\alpha_{n+1}\alpha_n}^i < \mathfrak{G}p_{\alpha_n\alpha_{n+1}}^{(n)}$, where `sign_diff` is between -2 and -1. The same holds for token j. Since the bounds are independent of the token and the interval has length at most $2u$, the result follows. □

This proposition implies that if we choose the step parameter u to be smaller than half the minimum difference between any two score gaps, i.e., $u < \bar{u}$ where $\bar{u} = \frac{1}{2}\min_{k,k',i,j}\left|\mathfrak{G}\gamma_{kk'}^i - \mathfrak{G}\gamma_{kk'}^j\right|$, then token movements become unique.

Corollary 6. *In the DeepSeek implementation, if we choose a constant step-size satisfying $u < \bar{u}$, then no two tokens will move between the same pair of origin and destination experts in two consecutive iterations. As a consequence, an expert's load cannot change by more than $(E-1)$ tokens between two consecutive iterations.*

Proof. A direct consequence of Proposition 5. □

Finally, we guarantee that DeepSeek ALF-LB must achieve an approximately balanced state.

Theorem 7. ***(Guarantee of Approximate Balancing)*** *When applying the DeepSeek ALF-LB procedure using a constant step parameter $u < \bar{u}$, the load of all experts must converge to the range $[L - (E-1), L + (E-1)]$. Moreover, once an expert's load enters that range, it remains in that range.*

Proof. If perfect balance is achieved, the shifts p_k stop changing and the algorithm terminates. Otherwise, there will always be an overloaded and an underloaded expert. If an expert k's load is above $L + (E-1)$, its shift p_k will decrease by u each iteration, causing it to shed tokens. By Corollary 6, its load can change by at most $(E-1)$ per iteration. So, its load must eventually enter the range $[L, L+(E-1)]$ without overshooting below L. A symmetric argument holds if an expert's load is below $L-(E-1)$. Once an expert's load enters the range $[L-(E-1), L+(E-1)]$, it cannot escape. For example, if an expert's load is $L + j$ for $1 \leq j \leq E-1$, it is overloaded and can only lose tokens. It can lose at most $E-1$ tokens. So its load will remain above $L+j-(E-1)$, which is greater than $L - E$. A similar argument holds for underloaded experts. Therefore, all expert loads will enter and remain within the stated range. □

When the number of tokens is much larger than the number of experts ($T \gg E$), the deviation of $(E-1)$ from the perfectly balanced load $L = KT/E$ is negligible. This completes the theoretical demonstration of why the DeepSeek ALF-LB procedure leads to the desirable balancing behavior seen experimentally.

5 Stochastic Analysis via Online Optimization

In practice, the affinity scores $\gamma_{ik}^{(n)}$ evolve during training. Thus, in this section, we generalize the previously considered Lagrangian (7) by assuming that the scores $\gamma_{ik}^{(n)}$ are *stochastic* and drawn from expert-dependent distributions. In particular, at iteration n, we assume a random affinity score $\gamma_{ik}^{(n)} \in (0,1)$ is observed for each token $i \in \{1, \dots, T\}$ and expert $k \in \{1, \dots, E\}$. The algorithm updates a shift vector $p^{(n)} \in \mathbb{R}^E$ and, for each token i, selects the K experts with the largest values of $\gamma_{ik}^{(n)} + p_k^{(n)}$, where $p_k^{(n)}$ is the k-th coordinate of $p^{(n)}$.

Using the notation of Section 2.2, we note that this section will consider a step-size choice of $\epsilon_k^{(n)} = u/n$ instead of the $u/|L - A_k^{(n)}|$ step-size chosen by Wang et al. (2024). This is because, when affinity scores become stochastic and time-varying, analyzing the coordinate-dependent $u/|L - A_k^{(n)}|$ step-size sequence becomes technically intricate. In contrast, the diminishing and coordinate-independent u/n step-size connects directly with ideas from online convex analysis (Hazan, 2016, Section 3.3.1) leading to cleaner theoretical insights. This adjustment maintains experimental relevance and practicality: Figure 2 and Table 1 demonstrate that using the u/n step-size is comparable in effectiveness as using the original $u/|L - A_k^{(n)}|$ step-size. In fact, Table 1 shows that the u/n step-size leads to a slightly *better* load balancing performance than the $u/|L - A_k^{(n)}|$ step-size at the cost of a slightly higher validation loss.

5.1 Notation

For any vector $z \in \mathbb{R}^E$, define $\mathrm{TopKInd}(z) \subseteq \{1, \dots, E\}$ to be the set of indices of the K largest coordinates of z with ties broken arbitrarily. For round n and token i, denote $\Gamma^{(n),i} := \left(\gamma_{i1}^{(n)}, \dots, \gamma_{iE}^{(n)}\right) \in \mathbb{R}^E$. Routing at round n sends token i to the experts in $\mathrm{TopKInd}\big(\Gamma^{(n),i} + p^{(n)}\big)$.

Fix an integer $K \in \{1, \dots, E\}$. We frame this as minimizing the per-round online loss, which corresponds to the dual online objective:

$$f^{(n)}(p) = \sum_{i=1}^{T} \sum_{k \in \mathrm{TopKInd}\left(\Gamma^{(n),i}+p\right)} \left(\gamma_{ik}^{(n)} + p_k\right) - L \sum_{k=1}^{E} p_k, \tag{10}$$

where $L := KT/E$ is the desired per-expert load for Top-K routing. The k-th component of the loss gradient $g^{(n)}(p) \in \mathbb{R}^E$ is given by

$$g_k^{(n)}(p) := \nabla_k f^{(n)}(p) = A_k^{(n)}(p) - L, \tag{11}$$

where $A_k^{(n)}(p) := \left|\left\{i : k \in \mathrm{TopKInd}\big(\Gamma^{(n),i} + p\big)\right\}\right|$ counts the number of tokens for which expert k lies in the per-token Top-K set under shifts p. The online dual update corresponding to (5) can then be rewritten as $p_k^{(n+1)} \leftarrow p_k^{(n)} - \epsilon_k^{(n)} g_k^{(n)}(p^{(n)})$.

5.2 Distributional Assumptions

Assume each affinity $\gamma_{ik}^{(n)}$, for a fixed expert k, is drawn independently[4] from a distribution that

- depends only on the expert k,
- has bounded support on $(0, 1)$, and
- has a probability density function (pdf) φ_k that is upper bounded by some expert-independent constant.

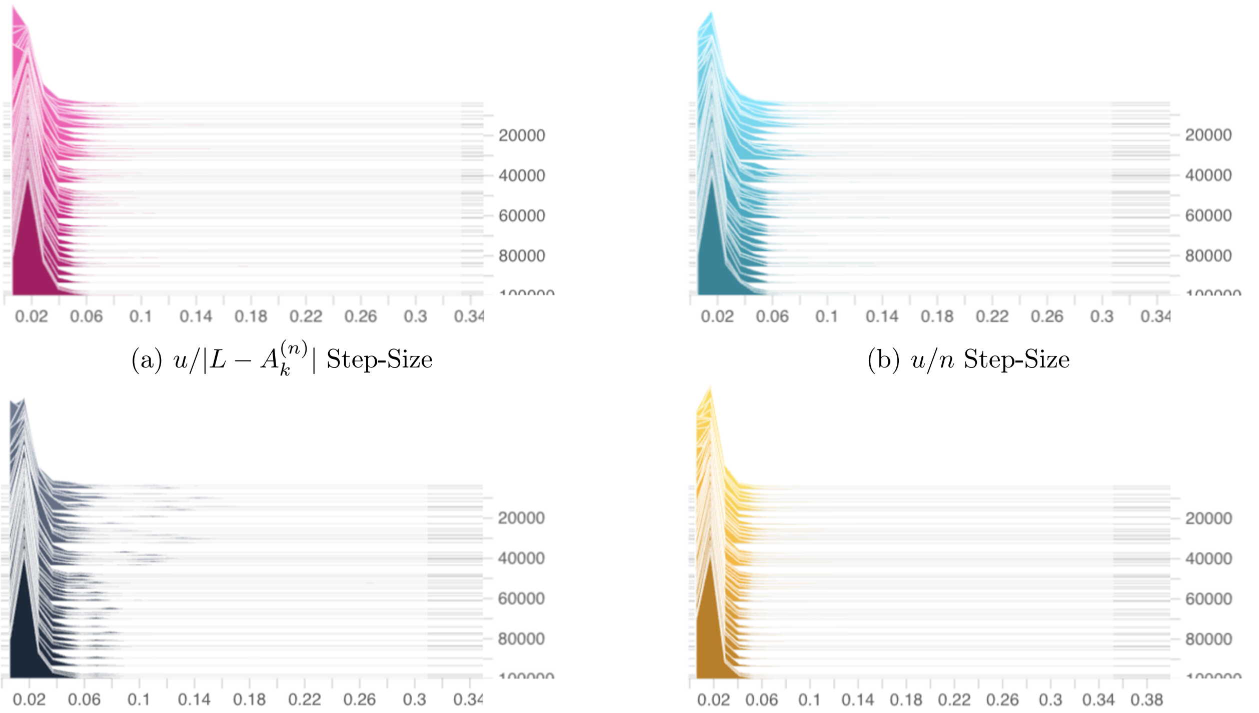
Thus, for fixed k, the vectors of random affinity scores $\Gamma^{(n),i} = \left(\gamma_{ik}^{(n)}\right)_{k=1}^{E}$ for the i-th token in the n-th iteration are i.i.d across i and n. While this assumption may seem strong, our experiments in Figure 3 from training 1B-parameter DeepSeekMoE models suggest that it is close to reality.

5.3 Online Loss Gradient Analysis

5.3.1 Unbiasedness

It is easy to check that the loss $f^{(n)}$ is convex. Thus, the expected loss $\mathbf{f}(p) = \mathbb{E}\left[f^{(n)}(p) \middle| p\right]$ is also convex. We first show that the loss gradient $g^{(n)}(p)$ is an unbiased estimator of $\nabla \mathbf{f}(p)$ with an explicit form expression.

[4] This independence assumption is a stylization: various mechanisms (attention, layer norm, etc.) in earlier layers could create dependencies between token embeddings. However, it gives us a starting point for building a tractable, baseline theory. Furthermore, Figure 3 demonstrates that the distributions of $\gamma_{ik}^{(n)}$ remain stable and well-behaved throughout training on DeepSeekMoE-1B models, which indicates that independence is empirically well-founded as an approximating simplification at least at the marginal distribution level.



(a) $u/|L - A_k^{(n)}|$ Step-Size

(b) u/n Step-Size

(c) $u/\sqrt{n}$ Step-Size

(d) Auxiliary Loss

Figure 3: Time-lapse histograms of the marginal distributions of $\gamma_{ik}^{(n)}$ during the training of 1B-parameter DeepSeekMoE models using different choices of step-size (Section 2.2). Experimental details in Section 3.

Proposition 8 (Unbiased Stochastic Gradient)**.** *For any fixed $p \in \mathbb{R}^E$,*

$$\mathbb{E}\left[g^{(n)}(p)\middle| p\right] \;=\; \nabla \mathbf{f}(p) \;=\; T\,\pi(p) - L\,\mathbf{1},$$

where $\mathbf{1}$ is the ones-vector, and $\pi(p) \in \mathbb{R}^E$ is the selection probabilities vector with k-th coordinate

$$\pi_k(p) \;:=\; \Pr(k \in \mathrm{TopKInd}(\Gamma + p))\,,$$

for some generic affinities $\Gamma \overset{d}{=} \Gamma^{(n),i}$. Necessarily, $\sum_k \pi_k(p) = K$ almost surely.

Proof. For a given token i, let

$$X_{ik}(p) := \mathbb{1}\left\{k \in \mathrm{TopKInd}\left(\Gamma^{(n),i} + p\right)\right\}, \tag{12}$$

where $\mathbb{1}\{\cdot\}$ denotes the indicator function. Then, $\mathbb{E}\left[X_{ik}(p) \mid p\right] = \pi_k(p)$ by definition and the k-th expert loads are $A_k^{(n)}(p) = \sum_{i=1}^{T} X_{ik}(p)$. Hence, $\mathbb{E}\left[A_k^{(n)}(p) \mid p\right] = T\,\pi_k(p)$. Therefore,

$$\begin{aligned}\mathbb{E}\left[g^{(n)}(p) \mid p\right] &= \mathbb{E}\left[\nabla f^{(n)}(p) \mid p\right] \\ &= \left(\mathbb{E}\left[A_1^{(n)}(p) \mid p\right] - L, \ldots, \mathbb{E}\left[A_E^{(n)}(p) \mid p\right] - L\right) \quad \text{by Eq. 11}\end{aligned}$$

$$= T\,\pi(p) - L\,\mathbf{1}.$$

Since the distribution of $\Gamma^{(n),i}$ is independent of i, observe that

$$\mathbf{f}(p) = T\,\mathbb{E}\left[\sum_{k\in\mathrm{TopKInd}(\Gamma+p)} (\Gamma_k + p_k)\right] - L\sum_k p_k.$$

To calculate $\nabla\mathbf{f}$, we can move the differentiation into the expectation via the dominated convergence theorem: the Γ_k have continuous densities so ties occur with probability zero; thus, for almost every realization, the partial derivative $\partial_{p_k}\sum_{m\in\mathrm{TopKInd}(\Gamma+p)}(\Gamma_m + p_m)$ exists and equals $\mathbb{1}\{k \in \mathrm{TopKInd}(\Gamma+p)\} \le 1$, yielding $\partial_{p_k}\mathbb{E}\left[\sum_{m\in\mathrm{TopKInd}(\Gamma+p)}(\Gamma_m + p_m)\right] = \Pr\{k \in \mathrm{TopKInd}(\Gamma+p)\} = \pi_k(p)$. The desired result follows. □

Using Proposition 8, we can compute the variance and second moment of $g^{(n)}(p)$.

Proposition 9. *(Variance and 2nd Moment) The variance and second moment of $g^{(n)}(p)$ are given by*

$$\mathrm{Var}\left[g^{(n)}(p)\middle|\, p\right] = T\left(K - \sum_{k=1}^{E}\pi_k(p)^2\right),$$

$$\mathbb{E}\left[\|g^{(n)}(p)\|^2\middle|\, p\right] = T^2\left(\sum_{k=1}^{E}\pi_k(p)^2 - \tfrac{K^2}{E}\right) + T\left(K - \sum_{k=1}^{E}\pi_k(p)^2\right).$$

Proof. Using Equation (11) and Proposition 8,

$$\begin{aligned}\mathrm{Var}\left(g^{(n)}(p)\middle|\, p\right) &= \mathbb{E}\left[\|g^{(n)}(p) - \nabla\mathbf{f}(p)\|^2 \mid p\right]\\ &= \mathbb{E}\left[\|T\pi(p) - A^{(n)}(p)\|^2 \mid p\right]\end{aligned}$$

where $A^{(n)}(p) = \left(A_1^{(n)}(p), \ldots, A_E^{(n)}(p)\right)$ is the vector of expert loads. Let $X_i(p) \in \{0,1\}^E$ denote the assignment vector for token i with components as in (12). Then, $A^{(n)}(p) = \sum_{i=1}^{T} X_i(p)$ and $\mathbb{E}[X_i(p)] = \pi(p)$. So,

$$\begin{aligned}\mathrm{Var}\left(g^{(n)}(p)\middle|\, p\right) &= \mathbb{E}\left[\left\|\sum_{i=1}^{T} T\pi(p) - X_i(p)\right\|^2 \middle|\, p\right]\\ &= \sum_{i=1}^{T}\mathbb{E}\left[\|\pi(p) - X_i(p)\|^2 \mid p\right] \quad \text{by independence across } i\\ &= T\sum_{k=1}^{E}\mathrm{Var}(X_{1k} \mid p) \quad \text{by identical distribution across } i\\ &= T\left(K - \sum_{k=1}^{E}\pi_k(p)^2\right),\end{aligned}$$

since $\sum_k X_{ik} = K$ a.s. and $\mathrm{Var}(X_{1k} \mid p) = \pi_k(p)\,(1 - \pi_k(p))$. Finally, decomposing the second moment and applying Proposition 8 gives

$$\begin{aligned}
\mathbb{E}\Big[\|g^{(n)}(p)\|^2 \mid p\Big] &= \|\nabla \mathbf{f}(p)\|^2 + \mathbb{E}\Big[\|g^{(n)}(p) - \nabla \mathbf{f}(p)\|^2 \mid p\Big] \\
&= T^2 \left(\sum_{k=1}^{E} \pi_k(p)^2 - \tfrac{K^2}{E}\right) + T\left(K - \sum_{k=1}^{E} \pi_k(p)^2\right).
\end{aligned}$$

This completes the proof. □

Proposition 9 will be useful later to prove Theorem 13.

5.4 Second-Order Analysis of Expected Loss

In the following Sections 5.4-5.6, we show that the expectation of the Top-K objective is *strongly convex* with respect to p updates under certain (realistic) assumptions. The strong convexity then allows us to show a logarithmic regret bound in Theorem 13. Without strong convexity, it is routine to verify that the regret bound is at best $\mathrm{O}(\sqrt{N})$ without additional assumptions. The next lemma characterizes the second directional derivative of the expected objective.

Proposition 10 (Second Directional Derivative)**.** *Let $\Gamma = (\Gamma_1, \ldots, \Gamma_E)$ be a random affinity vector in $\mathbb{R}^E$ with the properties in Section 5.2. For biases $p \in \mathbb{R}^E$ define*

$$\mathbf{F}_K(p) \;=\; \mathbb{E}\left[\sum_{k \in \mathrm{TopKInd}(\Gamma + p)} (\Gamma_k + p_k)\right],$$

and let φ_k and Φ_k denote the density and CDF of Γ_k, respectively. Then, for any direction $\delta \in \mathbb{R}^E$, its second directional derivative at p is given by the formula

$$D^2\mathbf{F}_K(p)[\delta, \delta] \;=\; \sum_{k<\ell} w_{k\ell}^{(K)}(p)\,(\delta_k - \delta_\ell)^2, \tag{13}$$

where the symmetric edge weights are

$$w_{k\ell}^{(K)}(p) \;=\; \int_{-\infty}^{\infty} \varphi_k(v - p_k)\,\varphi_\ell(v - p_\ell)\,B_{k,\ell}^{(K-1)}(v;p)\,dv, \qquad w_{k\ell}^{(K)}(p) \geq 0,$$

with

$$B_{k,\ell}^{(K-1)}(v;p) \;=\; \sum_{\substack{S \subseteq [E]\setminus\{k,\ell\} \\ |S| = K-1}} \prod_{j \in S} \Phi_j^c(v - p_j) \prod_{m \in [E]\setminus(\{k,\ell\}\cup S)} \Phi_m(v - p_m).$$

Proof. For some fixed argument $\gamma \in [0,1]^E$, define the function

$$\bar{f}_{p,K}(\gamma) = \sum_{k \in \mathrm{TopKInd}(\gamma + p)} (\gamma_k + p_k),$$

so that $\mathbf{F}_K(p) = \int \bar{f}_{p,K}(\gamma)\, \varphi(\gamma)\, d\gamma$, where $\varphi(\gamma) = \prod_{k=1}^{E} \varphi_k(\gamma_k)$ is the joint density of Γ. For $t \in \mathbb{R}$, define $p(t) := p + t\,\delta$ and $\widetilde{\mathbf{F}}_K(t) := \mathbf{F}_K\left(p(t)\right)$. Since ties occur with probability zero, $\mathbf{F}_K$ is a.s. differentiable with gradient $\nabla \mathbf{F}_K(p) = \pi(p)$. Hence, the chain rule gives

$$\widetilde{\mathbf{F}}'_K(0) \;=\; \sum_{k=1}^{E} \delta_k\, \pi_k(p).$$

We will next compute $\widetilde{\mathbf{F}}''_K(0)$. For each k, using independence and conditioning on $\Gamma_k = v$,

$$\begin{aligned}
\pi_k(p) &= \Pr(k \in \mathrm{TopKInd}(\Gamma + p)) \\
&= \int_0^1 \varphi_k(v)\; \Pr(k \in \mathrm{TopKInd}(\Gamma + p) \,|\, \Gamma_k = v)\, dv \\
&= \int_0^1 \varphi_k(v)\; \Pr(|\{j \neq k : \Gamma_j + p_j > v + p_k\}| \leq K{-}1)\, dv \\
&= \int_0^1 \varphi_k(v) \sum_{r=0}^{K-1} \Pr(|\{j \neq k : \Gamma_j + p_j > v + p_k\}| = r)\, dv \\
&= \int_0^1 \varphi_k(v) \sum_{r=0}^{K-1} \sum_{\substack{S \subseteq [E]\setminus\{k\} \\ |S|=r}} \Pr(\forall j{\in}S : \Gamma_j{+}p_j > v{+}p_k \text{ and } \forall m \notin S{\cup}\{k\} : \Gamma_m{+}p_m \leq v{+}p_k)\, dv \\
&= \int_0^1 \varphi_k(v) \sum_{r=0}^{K-1} \sum_{\substack{S \subseteq [E]\setminus\{k\} \\ |S|=r}} \prod_{j \in S} \Phi_j^c(v - p_j + p_k) \prod_{m \notin S \cup \{k\}} \Phi_m(v - p_m + p_k)\, dv.
\end{aligned}$$

where $\Phi_j^c(\cdot) := 1 - \Phi_j(\cdot)$ and S represents possible index sets within the top-K components of $\Gamma + p$ that are also larger than $v + p_k$. For notational convenience, set

$$\theta_{jk}(v,t) := v - (p_j - p_k) - t\,(\delta_j - \delta_k).$$

Then,

$$\pi_k\left(p(t)\right) = \int_0^1 \varphi_k(v) \sum_{r=0}^{K-1} \sum_{\substack{S \subseteq [E]\setminus\{k\} \\ |S|=r}} \left[\prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,t)\right) \prod_{m \notin S \cup \{k\}} \Phi_m\left(\theta_{mk}(v,t)\right) \right] dv.$$

Consider the integrand

$$\varphi_k(v) \sum_{r=0}^{K-1} \sum_{\substack{S \subseteq [E]\setminus\{k\} \\ |S|=r}} \prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,t)\right) \prod_{m \notin S \cup \{k\}} \Phi_m\left(\theta_{mk}(v,t)\right). \tag{14}$$

For each j, because Φ_j is differentiable everywhere on $\mathbb{R}$ except (possibly) at 0 or 1, the derivative of the integrand (14) with respect to t at $t = 0$ exists for all but finitely many $v \in (0,1)$. Thus, the integrand (14) is differentiable at $t = 0$ for almost all $v \in (0, 1)$.

Next, observe that for each fixed $S \subseteq [E] \setminus \{k\}$, the product in (14) has the a.e. derivative

$$\begin{aligned}
&\frac{d}{dt}\left[\prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,t)\right) \prod_{m \notin S \cup \{k\}} \Phi_m\left(\theta_{mk}(v,t)\right)\right] \\
&\qquad = \sum_{\ell \notin S \cup \{k\}} \underbrace{\left(\delta_k - \delta_\ell\right) \varphi_\ell\left(\theta_{\ell k}(v,t)\right) \prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,t)\right) \prod_{m \notin S \cup \{k,\ell\}} \Phi_m\left(\theta_{mk}(v,t)\right)}_{:= \Xi^+_{S,\ell}} \\
&\qquad\qquad - \sum_{\ell \in S} \underbrace{\left(\delta_k - \delta_\ell\right) \varphi_\ell\left(\theta_{\ell k}(v,t)\right) \prod_{j \in S \setminus \{\ell\}} \Phi_j^c\left(\theta_{jk}(v,t)\right) \prod_{m \notin S \cup \{k\}} \Phi_m\left(\theta_{mk}(v,t)\right)}_{:= \Xi^-_{S,\ell}}.
\end{aligned}$$

This leads to a telescoping cancellation across r in the integrand (14). Specifically, for each fixed $r < K-1$ and ℓ, every index set S_r such that $|S_r| = r$ corresponds to another index set S_{r+1}^ℓ such that $|S_{r+1}^\ell| = r + 1$ and $S_{r+1}^\ell = S_r \cup \{\ell\}$. It is easy to check that $\Xi^+_{S_r,\ell} = \Xi^-_{S_{r+1}^\ell,\ell}$.

So, the sum in (14) telescopes over r except at the $r = K-1$ boundary where $\Xi^+_{S_{K-1},\ell}$ has no corresponding "$\Xi^-_{S_K^\ell,\ell}$" term to cancel with. Hence, for almost all fixed $v \in (0, 1)$,

$$\begin{aligned}
&\frac{d}{dt}\left[\varphi_k(v) \sum_{r=0}^{K-1} \sum_{\substack{S \subseteq [E]\setminus\{k\} \\ |S| = r}} \prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,t)\right) \prod_{m \notin S \cup \{k\}} \Phi_m\left(\theta_{mk}(v,t)\right)\right]\Bigg|_{t=0} \\
&= \varphi_k(v) \sum_{\ell \neq k} \left(\delta_k - \delta_\ell\right) \varphi_\ell\left(\theta_{\ell k}(v,0)\right) \sum_{\substack{S \subseteq [E]\setminus\{k,\ell\} \\ |S| = K-1}} \prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,0)\right) \prod_{m \notin S \cup \{k,\ell\}} \Phi_m\left(\theta_{mk}(v,0)\right).
\end{aligned}$$

Next, for any non-zero $h \in \mathbb{R}$, it is routine to check that the assumptions in Section 5.2 imply that the integrand of the following is uniformly bounded:

$$\begin{aligned}
&\frac{1}{h}\left[\pi_k\left(p(h)\right) - \pi_k\left(p(0)\right)\right] \\
&\quad = \int_0^1 \frac{1}{h}\left[\varphi_k(v) \sum_{r=0}^{K-1} \sum_{\substack{S \subseteq [E]\setminus\{k\} \\ |S| = r}} \prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,h)\right) \prod_{m \notin S \cup \{k\}} \Phi_m\left(\theta_{mk}(v,h)\right)\right. \\
&\qquad\qquad \left. - \varphi_k(v) \sum_{r=0}^{K-1} \sum_{\substack{S \subseteq [E]\setminus\{k\} \\ |S| = r}} \prod_{j \in S} \Phi_j^c\left(\theta_{jk}(v,0)\right) \prod_{m \notin S \cup \{k\}} \Phi_m\left(\theta_{mk}(v,0)\right)\right] dv
\end{aligned}$$

Therefore, by the dominated convergence theorem, we have that $\pi_k\left(p(t)\right)$ is differentiable at $t = 0$, with the derivative being given (after a change of variables) by

$$
\begin{aligned}
&D\pi_k(p)[\delta] \\
&\quad = \sum_{\ell \neq k}(\delta_k - \delta_\ell) \int_{-\infty}^{\infty} \varphi_k(v - p_k)\, \varphi_\ell\,(v - p_\ell) \sum_{\substack{S \subseteq [E]\setminus\{k,\ell\} \\ |S| = K-1}} \prod_{j \in S} \Phi_j^c(v - p_j) \prod_{m \notin S \cup \{k,\ell\}} \Phi_m(v - p_m)\, dv
\end{aligned}
$$

Finally, by symmetry,

$$
\begin{aligned}
D^2\mathbf{F}_K(p)[\delta,\delta] &= \frac{d}{dt}\sum_{k=1}^{E} \delta_k\, \pi_k(p + t\delta)\bigg|_{t=0} = \sum_{k=1}^{E} \delta_k\, D\pi_k(p)[\delta] \\
&= \sum_{k=1}^{E}\sum_{\ell \neq k} \delta_k(\delta_k - \delta_\ell)\, w_{k\ell}^{(K)}(p) = \sum_{1 \leq k < \ell \leq E} w_{k\ell}^{(K)}(p)\,(\delta_k - \delta_\ell)^2,
\end{aligned}
$$

which is exactly (13). □

5.5 Experimentally-Realistic Assumptions on p

Observe that the TopKInd decision of the MoE router is invariant to adding the same constant to all coordinates of p. Motivated by this, we define the zero-sum subspace

$$
\mathcal{Z} := \left\{ z \in \mathbb{R}^E : \sum_{k=1}^{E} z_k = 0 \right\},
$$

where $\mathcal{Z}$ is the linear subspace orthogonal to the all-ones vector. Thus, we assume the following about ALF-LB for some update direction δ':

$$
p^{(n+1)} \leftarrow \mathrm{Proj}_{\mathcal{Z}}\left(p^{(n)} - \delta'\right). \tag{15}
$$

Remark 11 (**Practicality of $\mathcal{Z}$ Assumptions**). The assumption (15) is not artificial; it arises naturally from the problem definition:

- **Zero-sum gradients.** Since $\sum_k A_k^{(n)}(p) = TK$, the components of the gradient (11) sum to zero:

  $$
  \sum_k \nabla_k f^{(n)}(p) = \sum_k (A_k^{(n)}(p) - L) = TK - EL = 0.
  $$

  Thus, any update of the form

  $$
  p^{(n+1)} \leftarrow p^{(n)} - \epsilon^{(n)} \nabla f^{(n)}(p^{(n)})
  $$

  automatically preserves $p^{(n+1)} \in \mathcal{Z}$ as long as $p^{(n)} \in \mathcal{Z}$. In practice, we initialize with $p^{(0)} = 0$, so the projection in (15) is just the identity mapping.
- **Explicit $\mathcal{Z}$-projection with per-coordinate step-sizes.** In the more general case where

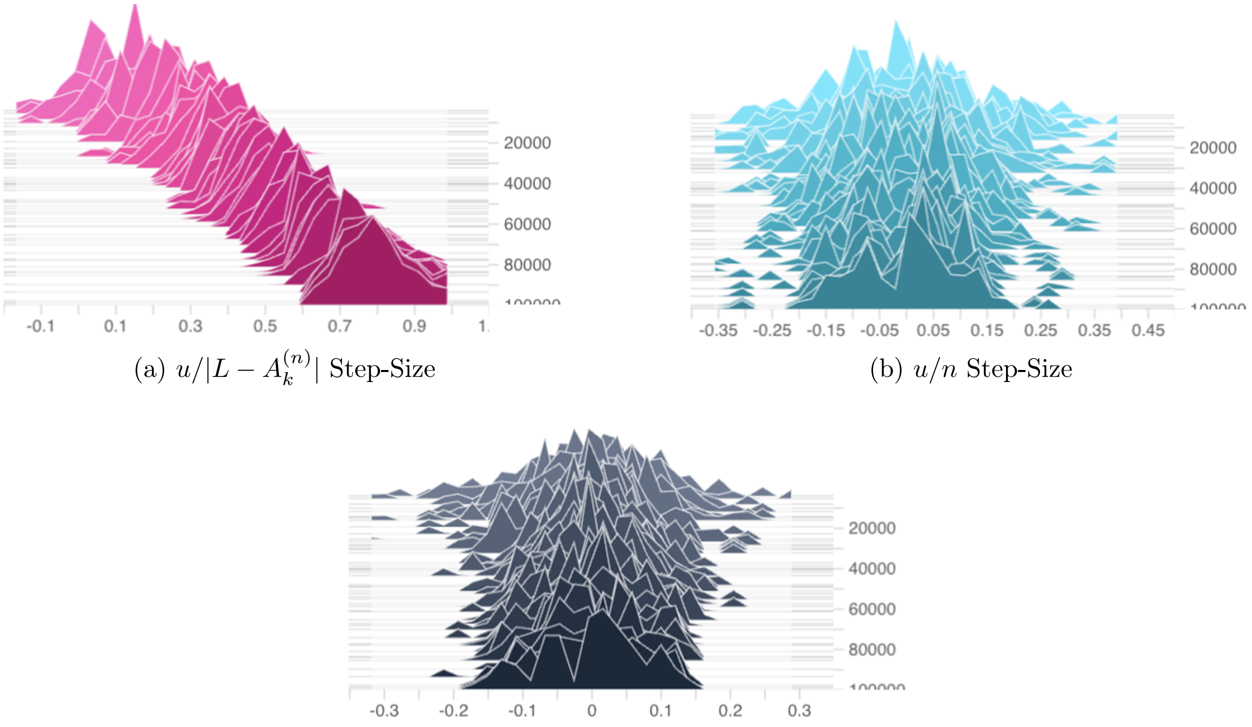
heterogeneous step-sizes $\epsilon_k^{(n)}$ are used across coordinates,

$$p^{(n+1)} \leftarrow p^{(n)} - \left(\epsilon_1^{(n)} g_1^{(n)}, \ldots, \epsilon_E^{(n)} g_E^{(n)}\right),$$

the updated $p^{(n+1)}$ may not reside in $\mathcal{Z}$. (In fact, the difference between per-coordinate step-sizes and homogeneous step-sizes can be seen in Figure 4 where the per-coordinate $\epsilon_k^{(n)} = u/|L - A_k^{(n)}|$ step-size results in a bias distribution that shifts rightward over time while the homogeneous $\epsilon^{(n)} = u/n$ and $\epsilon^{(n)} = u/\sqrt{n}$ step-sizes result in bias distributions that stay centered around zero.) However, in this per-coordinate step-size case, it is well-known that the projection onto $\mathcal{Z}$ is equivalent to subtracting the componentwise mean:

$$\mathrm{Proj}_{\mathcal{Z}}(p) = p - \left(\frac{1}{E}\sum_{k=1}^{E} p_k\right)\mathbf{1},$$

which is computationally negligible.



(a) $u/|L - A_k^{(n)}|$ Step-Size

(b) u/n Step-Size

(c) $u/\sqrt{n}$ Step-Size

Figure 4: Time-lapse histograms of the marginal distributions of the ALF-LB biases p during the training of 1B-parameter DeepSeekMoE models using different choices of step-size (Section 2.2). No explicit constraints were enforced on p. Section 3 provides experimental details.

Additionally, we will make the technical assumption that $\mathrm{diam}(p) := \max_j p_j - \min_j p_j \leq 1 - \kappa$ for some constant $\kappa > 0$, which we found holds without explicit enforcement in our experiments on

1B-parameter DeepSeekMoE models (Figure 4). Thus, we can realistically assume

$$p \in \mathrm{dom}_{\mathcal{Z}}^{\kappa} := \{p \in \mathcal{Z} : \mathrm{diam}(p) \leq 1 - \kappa\}.$$

5.6 Strong Convexity

Next, observe that the component densities of the affinity scores Γ are continuous and strictly positive on $(0, 1)$. Thus, by the continuity of $w_{k\ell}^{(K)}(p)$ in p and compactness of $\mathrm{dom}_{\mathcal{Z}}^{\kappa}$,

$$c_K(d) \;:=\; \inf_{p \in \mathrm{dom}_{\mathcal{Z}}^{\kappa}} \min_{k<\ell} w_{k\ell}^{(K)}(p) \;>\; 0.$$

Hence, by Proposition 10,

$$\delta^\top \nabla^2 \mathbf{F}_K(p)\, \delta \;\geq\; c_K(d) \sum_{k<\ell} (\delta_k - \delta_\ell)^2. \tag{16}$$

The assumption (15) ensures that $p^{(n)} \in \mathcal{Z}$ for all n. Thus, since $\mathcal{Z}$ is a linear subspace,

$$\begin{aligned} p^{(n+1)} &= \mathrm{Proj}_{\mathcal{Z}}\left(p^{(n)} - \delta'\right) \\ &= \mathrm{Proj}_{\mathcal{Z}}\left(p^{(n)}\right) - \mathrm{Proj}_{\mathcal{Z}}\left(\delta'\right) \\ &= p^{(n)} - \underbrace{\mathrm{Proj}_{\mathcal{Z}}\left(\delta'\right)}_{\delta}. \end{aligned}$$

Since the update direction δ lies in $\mathcal{Z}$,

$$\sum_{k<\ell} (\delta_k - \delta_\ell)^2 \;=\; E\,\|\delta\|^2 - \left(\sum_{k=1}^{E} \delta_k\right)^2 \;=\; E\,\|\delta\|^2.$$

Combining with property (16), this yields

$$\delta^\top \nabla^2 \mathbf{F}_K(p)\, \delta \;\geq\; c_K(d) E\,\|\delta\|^2.$$

Recall the expected loss is $\mathbf{f}(p) = T\mathbf{F}_K(p) - L\sum_k p_k$ and observe the linear term does not affect curvature; thus, for all $p, \delta \in \mathcal{Z}$ with p having diameter at most d, $\mathbf{f}$ is μ_K-strongly convex with

$$\mu_K := T c_K(d) E. \tag{17}$$

5.7 Logarithmic Regret Bound for ALF-LB

Consider the minimizer of the expected loss

$$p^* = \arg\min_{p \in \mathcal{Z}} \mathbf{f}(p).$$

Since $\mathbf{f}$ is μ_K-strongly convex in $\mathcal{Z}$, p^* is necessarily unique.

Define the regret $R_N := \sum_{n=1}^{N} \left(f^{(n)}(p^{(n)}) - f^{(n)}(p^*)\right)$. We now give a logarithmic bound on the expected regret $\mathbb{E}[R_N]$ with the ALF-LB update

$$p^{(n+1)} \leftarrow \mathrm{Proj}_{\mathcal{Z}}\left(p^{(n)} - \epsilon^{(n)}\,\nabla f^{(n)}\left(p^{(n)}\right)\right). \tag{18}$$

While the details are adapted to the specific problem at hand, the proof technique is standard in online convex optimization (see, for example, Hazan (2016, Section 3.3.1)). For clarity, define the following short-hand notations:

$$\Delta_n := \mathbb{E}\Big[\|p^{(n)}{-}p^*\|^2\Big]\,,\; s_n := \sum_{k=1}^{E} \pi_k\left(p^{(n)}\right)^2,\; a_n := \mathbb{E}\left[\mathbf{f}\left(p^{(n)}\right) - \mathbf{f}\left(p^*\right)\right],\; \sigma^2_{T,E,K} := T^2\left(K - \tfrac{K^2}{E}\right).$$

Lemma 12 (One-step accounting)**.** *Under the assumptions and notations of Section 5.1-5.6, for any* $\epsilon^{(n)}{>}0$*, the iteration* (18) *satisfies*

$$2\,a_n \;\le\; \frac{\Delta_n - \Delta_{n+1}}{\epsilon^{(n)}} - \mu_K\,\Delta_n \;+\epsilon^{(n)}\,\sigma^2_{T,E,K}. \tag{19}$$

Proof. Since $\mathcal{Z}$ is a linear subspace, the projection operator is nonexpansive. Thus,

$$\begin{aligned}
\|p^{(n+1)}{-}p^*\|^2 &= \left\|\mathrm{Proj}_{\mathcal{Z}}\left(p^{(n)} - \epsilon^{(n)}\nabla f^{(n)}\left(p^{(n)}\right)\right) - p^*\right\|^2 \\
&\le \left\|p^{(n)} - \epsilon^{(n)}\nabla f^{(n)}\left(p^{(n)}\right) - p^*\right\|^2 \\
&\le\; \|p^{(n)}{-}p^*\|^2 - 2\epsilon^{(n)}\left\langle \nabla f^{(n)}\left(p^{(n)}\right), p^{(n)}{-}p^*\right\rangle + \left(\epsilon^{(n)}\right)^2\left\|\nabla f^{(n)}\left(p^{(n)}\right)\right\|^2.
\end{aligned}$$

Taking conditional expectation and using Proposition 8 gives

$$\begin{aligned}
\mathbb{E}\left[\|p^{(n+1)}{-}p^*\|^2 \mid p^{(n)}\right] \le \|p^{(n)}{-}p^*\|^2 - 2\epsilon^{(n)}\left\langle \nabla\mathbf{f}(p^{(n)}), p^{(n)}{-}p^*\right\rangle \\
+ (\epsilon^{(n)})^2\,\mathbb{E}\left[\|\nabla f^{(n)}(p^{(n)})\|^2 \mid p^{(n)}\right].
\end{aligned}$$

Since the TopKInd decision is invariant to adding the same constant to all coordinates of p, we can assume without loss of generality that $p^* \in \mathcal{Z}$. Thus, since $\mathcal{Z}$ is a linear subspace, $p^{(n)} - p^* \in \mathcal{Z}$. Then, the μ_K-strong convexity of $\mathbf{f}$ in $\mathcal{Z}$ (Section 5.6) gives

$$2\left(\mathbf{f}\left(p^{(n)}\right) - \mathbf{f}\left(p^*\right)\right) + \mu_K\,\|p^{(n)}{-}p^*\|^2 \le 2\left\langle \nabla\mathbf{f}(p^{(n)}), p^{(n)}{-}p^*\right\rangle.$$

Combining the last two expressions, taking total expectation, and rearranging gives

$$2\,a_n \;\le\; \frac{\Delta_n - \Delta_{n+1}}{\epsilon^{(n)}} - \mu_K\,\Delta_n \;+\epsilon^{(n)}\,\mathbb{E}\Big[\|\nabla f^{(n)}(p^{(n)})\|^2\Big]. \tag{20}$$

Recall from Section 5.1 that the gradient is $\nabla f^{(n)}(p) = A^{(n)}(p) - L\,\mathbf{1}$ where $\sum_k A_k^{(n)}(p) = TK$ and each $A_k^{(n)}(p) \in [0, T]$. It is then easy to check that, for any p,

$$\|\nabla f^{(n)}(p)\|^2 \leq \sigma_{T,E,K}^2.$$

Substituting this bound into (20) yields the desired result. □

Theorem 13. ***(Logarithmic Regret)*** *Consider the update (18) run for N iterations with $\epsilon^{(n)} = 1/(\mu_K n)$. Then,*

$$\mathbb{E}[R_N] \;\leq\; \frac{\sigma_{T,E,K}^2}{2\mu_K}\,(1 + \ln N).$$

Proof. Observe that $\mathbb{E}[R_N] = \sum_{n=1}^N a_n$. Summing (19) over $n = 1, \ldots, N$ gives

$$2\sum_{n=1}^N a_n \leq \sum_{n=1}^N \left(\frac{\Delta_n - \Delta_{n+1}}{\epsilon^{(n)}} - \mu_K \Delta_n \right) + \sum_{n=1}^N \epsilon^{(n)} \sigma_{T,E,K}^2.$$

The first term on the right-hand side is a telescoping sum which evaluates to

$$\begin{aligned}\sum_{n=1}^N \left(\frac{\Delta_n - \Delta_{n+1}}{\epsilon^{(n)}} - \mu_K \Delta_n \right) &= \sum_{n=1}^N \left(\mu_K n (\Delta_n - \Delta_{n+1}) - \mu_K \Delta_n \right) \\ &= \mu_K \sum_{n=1}^N \left((n{-}1)\Delta_n - n\Delta_{n+1} \right) \\ &= -\,\mu_K \, N \, \Delta_{N+1}.\end{aligned}$$

Dropping this non-positive term, we are left with

$$2\sum_{n=1}^N a_n \;\leq\; \sum_{n=1}^N \epsilon^{(n)}\, \sigma_{T,E,K}^2 \;=\; \frac{\sigma_{T,E,K}^2}{\mu_K} \sum_{n=1}^N \frac{1}{n}.$$

Invoking the classic $\sum_{n=1}^N \frac{1}{n} \leq 1 + \ln N$ inequality and dividing by 2 yields the desired result. □

Acknowledgements. The authors thank Alexandre Belloni, John Birge, Rene Caldentey, and Chamsi Hssaine for helpful discussions and feedback during the preparation of this paper.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, and Shyamal Anadkat. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Shipra Agrawal and Nikhil R Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1405–1424. SIAM, 2014.

Santiago Balseiro, Haihao Lu, and Vahab Mirrokni. Dual mirror descent for online allocation problems. In *International Conference on Machine Learning*, pages 613–628. PMLR, 2020.

Santiago Balseiro, Haihao Lu, and Vahab Mirrokni. Regularized online allocation problems: Fairness and beyond. In *International Conference on Machine Learning*, pages 630–639. PMLR, 2021.

Dimitri Bertsekas. *Network optimization: continuous and discrete models*, volume 8. Athena Scientific, 1998.

Dimitri P Bertsekas. Auction algorithms for network flow problems: A tutorial introduction. *Computational optimization and applications*, 1(1):7–66, 1992.

Dimitri P Bertsekas. Auction algorithms. In *Encyclopedia of optimization*, pages 128–132. Springer, 2008.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, and Yu Wu. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

DeepSeek-AI. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Epoch AI. Key trends and figures in machine learning, 2023. URL https://epoch.ai/trends. Accessed: 2025-09-27.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, and Shibo Wang. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Hendricks, Johannes Welbl, and Aidan Clark. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Rodolphe Jenatton, Jim Huang, Dominik Csiba, and Cedric Archambeau. Online optimization and regret guarantees for non-additive long-term constraints. *arXiv preprint arXiv:1602.05394*, 2016.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, Zhifeng Chen, and Yonghui Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2021.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. Base layers: Simplifying training of large, sparse models. In *International Conference on Machine Learning*, pages 6265–6274. PMLR, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

Jaime Sevilla, Lasse Heim, Amanda Askell Ho, Noah Buchan, Alex Snell, Maruan Alhussein, Natasha Jaques McAleese, William Biles, Kevin McKee, and Joey Leung. Compute trends across three eras of machine learning. *arXiv preprint arXiv:2202.05924*, 2022.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

E Strubell, A Ganesh, and A McCallum. Energy and policy considerations for deep learning in nlp. proceedings of the 57th annual meeting of the association for computational linguistics (acl). *Stroudsburg, PA, USA. Association for Computational Linguistics*, 2019.

Neil Thompson, Kristjan Greenewald, Keeheon Lee, and Gustavo F. Manso. The computational limits of deep learning. *arXiv preprint arXiv:2007.05558*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*, 2024.

Wenxin Zhang, Santiago R Balseiro, Robert Kleinberg, Vahab Mirrokni, Balasubramanian Sivan, and Bartek Wydrowski. Optimal and stable distributed bipartite load balancing. *arXiv preprint arXiv:2411.17103*, 2024.