PRINCIPLED TOP-k SELECTION FOR LANGUAGE MODELS WITH HYBRID GRADIENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Selecting the best k items out of m candidates is a critical component of modern large language model systems, such as document selection in Retrieval-Augmented Generation (RAG) and expert routing in Mixture-of-Experts (MoEs). However, training these selection modules remains challenging due to weak gradient signals and suboptimal exploration-exploitation tradeoffs. Furthermore, prior works often rely on heuristics, lacking principled objectives and approaches that explicitly model and solve the top-k selection problem. In this work, we propose a principled objective for training selection modules, whose gradient naturally provides richer training signals in a hybrid form—containing a supervised-gradient component and a policy-gradient component. We show that the selection problem becomes harder as m increases, and our algorithm converges at rate $O(1/\sqrt{T})$, with the optimal upper bound achieved by balancing between bias and variance. Practically, we apply our method to a set of tasks involving top-k selection, including synthetic regression problems, RAG, and MoE systems, showing that our method outperforms the baselines in next-token prediction perplexity and QA accuracies.

1 INTRODUCTION

In machine learning systems, selecting k items from a total of m candidates is a common and critical component. We consider the following abstraction. Given a training data $(x, y) \in \mathcal{D}$, suppose a model (or part of a model) is composed of a *selection module* θ, which outputs a score of each item $s_\theta(i|x)$, and a *prediction module* ϕ, which takes the selected item i and runs the forward pass to obtain predictions $h_\phi(x, i)$. Following the conditional computation graph, if we were able to consider the outputs of all the items, the loss can be formulated as

$$\min_{\theta,\phi} F(\theta, \phi) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathcal{L}\left(y, \sum_{i=1}^{m} p_\theta(i|x) h_\phi(x, i)\right), \quad p_\theta(i|x) = \frac{e^{s_\theta(i|x)}}{\sum_{j=1}^{m} e^{s_\theta(j|x)}}. \tag{1}$$

Here, $\mathcal{L}$ can be an arbitrary loss such as mean-squared-error in regression and cross-entropy in classification. In practice, however, this objective can be impractical and expensive to optimize. The candidate pool size m continuously increases, such as the increasing number of experts in MoE and the increasing size of candidate document pools in RAG systems. As it is infeasible to incorporate all the m candidates during training or inference, we typically select the most promising k items to generate predictions by training top-k selection modules, such as routers in MoE and rerankers in RAG. A common compromise is to simply apply the summation over (noisy) top-k items (Ram et al., 2023; Lewis et al., 2020; Guu et al., 2020a), i.e., selecting $I_x = \arg\text{top-}k_{i\in[m]}(s_\theta(i|x))$, and optimizing

$$\min_{\theta,\phi} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathcal{L}\left(y, \sum_{i\in I_x} p_\theta(i|x) h_\phi(x, i)\right), \quad p_\theta(i|x) = \frac{e^{s_\theta(i|x)}}{\sum_{j\in I_x} e^{s_\theta(j|x)}}. \tag{2}$$

However, several challenges in top-k selection still remain. There exist numerous updating rules that do not explicitly model the selection procedure into the optimization problem, but rather rely on generic end-to-end training losses to update all trainable variables (Shazeer et al., 2017; Dai et al., 2024; Fedus et al., 2022). Understanding the dynamics of the underlying selection mechanisms remain open. Even if the selection objective exists, the connection between existing formulations (such as Eq. (2)) and the true objective Eq. (1) is unclear. In addition, the top-k operation is generally

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

not differentiable, and the gradient signal for training the selection module can be suboptimal. For instance, in Eq. (2), the gradient of the selection module θ flows via the aggregation weights in the truncated summation over k items. Therefore, the gradient over θ teaches how to adjust weights of the *already selected* items in I_x, but does not directly supervise *what items to select among all m options*. This lack of training signals for selection has been a well-known bottleneck of training selection modules (Liu et al., 2023; Panda et al., 2025), and the issue is more pronounced as m increases, leading to a growing search space $O(m^k)$.

When the loss $\mathcal{L}$ is the negative log-likelihood, the per-sample objective can equivalently be expressed in terms of maximizing $p(y|x)$. Introducing either an item-level latent variable i or a subset-level latent variable I, we can decompose $p(y|x)$ as $p(y|x) = \sum_i p(y|x,i)p(i|x) = \sum_I p(y|x,I)p(I|x)$. This partially motivates our proposed set-level objective. Additionally, unbiased sampling following an item-level distribution may result in selecting the same item more than once, which does not match common practical implementations. Therefore, we propose the following surrogate objective where we perform top-k selection at the **set level**, as opposed to the item level:

$$J(\theta, \phi) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} J_x(\theta, \phi) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathbb{E}_{I\sim\pi_\theta(\cdot|x)} \left[\mathcal{L}(y, f_{\theta,\phi}(x, I))\right], \quad |I| = k, \tag{3}$$

where J_x denotes the per-sample loss over (x, y), $\pi_\theta(\cdot|x)$ is a distribution *over subsets with cardinality* k, and $f_{\theta,\phi}(x, I)$ denotes a general weighted aggregation function to combine selected items in I. In Section 3.1), we discuss some specific requirements on $\mathcal{L}$ and $f_{\theta,\phi}(x, I)$ such that Objective (3) can be viewed as an upper bound of Objective (1).

We develop an algorithm to optimize this objective. The unbiased gradient of Objective (3) naturally comprises two hybrid terms that learn both selection and weight adjustments over the selected subset. As we detail in Section 3.1, the first term corresponds to a policy gradient, and the second corresponds to a classic supervised gradient. As the policy gradient could have high variance, we additionally introduce a scalar λ in front of the policy gradient to adjust the tradeoff between the two terms. In Figure 1, our method achieves an improved bias/variance tradeoff and outperforms existing approaches with only the supervised gradient (Zibakhsh et al., 2025) or approximating the policy gradient (Liu et al., 2023; 2024) by more than 20% in top-1 selection accuracy (details in Section 5).

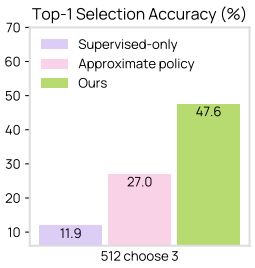


Figure 1: Our method outperforms prior ones that abandon or approximate the policy gradient.

Theoretically, we prove that our training method converges to stationary points of our objective and to neighborhoods of stationary points of the true objective (1) at a rate of $O(1/\sqrt{T})$. The bias/variance tradeoff and thus the convergence upper bound is adjusted by λ, whose optimal value is problem-dependent. We explore a synthetic regression problem to illustrate the difficulty of learning the selection module and the superiority of our method in exploring more subsets and improving the final performance. Furthermore, we apply our method to language models, including finetuning rerankers for in-context RAG and training MoE models from scratch. We observe that our method consistently outperforms the existing methods under all candidate pool sizes m. Our contributions are summarized as follows.

Contributions. We formalize the problem of sparse top-k selection in language models and propose a surrogate objective that directly optimizes over selected subsets of items, which also serves as an upper bound of the true objective for certain functions. This objective naturally yields hybrid gradients with two components (policy and supervised parts) that offer richer gradient signals for top-k selection than prior approaches. We establish convergence guarantees of our algorithm, and analyze the optimal bias/variance tradeoff. We evaluate our approach on different losses and tasks, including mean-squared error regression and cross-entropy loss on language tasks. Particularly, we fine-tune rerankers in RAG systems for language modeling and question answering (QA), and we train MoE models for different numbers of experts for language modeling from scratch. Our approach consistently outperforms strong baselines in terms of perplexity and task-specific metrics.

2 Related Work

Top-k selection. Selecting a sparse subset of k items from a large candidate pool is a challenging problem due to the non-differentiable top-k operator. A line of work proposes differentiable

ranking and relaxations of discrete selection, such as stochastic softmax tricks (Paulus et al., 2020), optimal-transport-based approximations (Xie et al., 2020), PiRank (Swezey et al., 2021), Laplace-based relaxations (Struski et al., 2025), and linear-time top-k operators for recommendation (Zhu et al., 2025). While these methods improve gradient-based optimization, they typically require dense forward passes over all candidates or violate strict sparsity constraints (Rosenbaum et al., 2017; Bengio et al., 2015), making them incompatible with applications considered in this work such as strict k-expert MoE routing.

Retrieval-augmented generative models. Classical RAG training optimizes a marginal likelihood over latent documents, typically approximated using deterministic top-k selection (Izacard et al., 2023; Lewis et al., 2020; Guu et al., 2020b; Borgeaud et al., 2022; Izacard and Grave, 2021). More recent approaches improve retrieval via reranking (Nogueira et al., 2019; Nogueira and Cho, 2019; Zhuang et al., 2023; Qin et al., 2024) or alignment with language model signals (Shi et al., 2024; Ram et al., 2023). Some work optimizes retrieval with reinforcement learning signals such as answer correctness or heuristic feedback (Li et al., 2025; Ma et al., 2023; Zhang and Gao, 2024; Basu et al., 2024). In contrast, our method yields an unbiased hybrid gradient in which the policy gradient component corresponds to a single-sample REINFORCE estimator (Williams, 1992; Sutton et al., 1999).

Mixture-of-experts. MoE architectures scale model capacity via sparse activation of experts (Jacobs et al., 1991; Jordan and Jacobs, 1994; Shazeer et al., 2017; Fedus et al., 2022; Lepikhin et al., 2020; Lewis et al., 2021; Zhou et al., 2022; Zoph et al., 2022; Dai et al., 2024; Jiang et al., 2024), but training the router remains challenging due to suboptimal gradient signals flowing through discrete top-k selection. Prior work relies on load-balancing losses (Riquelme et al., 2021; Lepikhin et al., 2020; Shazeer et al., 2017; Fedus et al., 2022; Dai et al., 2024; Lewis et al., 2021) or noisy gating (Shazeer et al., 2017; Zibakhsh et al., 2025; Fedus et al., 2022; Lepikhin et al., 2020; Zoph et al., 2022; Riquelme et al., 2021) for improved exploration/exploitation tradeoffs in expert selection. More recent work improves gradient flow through the router, including proxy gradients (Panda et al., 2025) and higher-order approximations to the policy gradient (Liu et al., 2023; 2024). However, many of these approaches remain heuristics, lack gradient signals for unselected items, or follow objectives that are inconsistent with practical deployment by sampling each expert independently. In contrast, our framework explicitly models subset selection as stochastic sampling over unordered k-sized sets and derives a principled objective and unbiased hybrid gradient composed of policy and supervised parts.

3 Objective and Hybrid Gradients

Let $\mathcal{D}$ be the training set containing a collection of samples $\{(x, y)\}$ and $\mathcal{L}(y, \hat{y})$ be a general per-sample loss, such as mean squared error in regression or negative log-likelihood in language modeling. In various language models involving selecting k items, the optimization variables consist of two groups: the selection parameters θ (e.g., rerankers in RAG or the router in MoE), and the remaining prediction parameters ϕ (e.g., the language model in RAG or the experts in MoE). The selection module θ produces a score $s_\theta(i|x), i \in [m]$ for each item, from which we obtain a *probability distribution over subsets* $\pi_\theta(I|x)$ where $|I| = k$. We then sample a subset I from $\pi_\theta(I|x)$ and feed it to the weighted aggregation function $f_{\theta,\phi}(x, I)$. In Figure 2, we illustrate the forward pass of one selection decision and prediction (left) and its instantiations in RAG (middle) and MoE (right).

In the following, we describe the general form of our proposed surrogate objective and its gradients in Section 3.1, discuss methods of sampling from π_θ in Section 3.2, and show how the prediction module performs the subsequent forward computation Section 3.3.

3.1 Surrogate Objective and Hybrid Gradients

In Eq. (3), we present the objective assuming the number of selection modules is one. In this section, we first examine this objective in detail, discuss its gradients, and then extend it to the case of having multiple selection modules in the model architecture.

Recall that $p(y|x) = \sum_{i=1}^{m} p_\theta(i|x)p(y|x, i) = \sum_I p(I|x)p(y|x, I)$. When the output $f_{\theta,\phi}(x, I)$ defines a consistent conditional probability given I such that $f_{\theta,\phi}(x, I) = p(y|x, I)$ and loss $\mathcal{L}$ is convex, such as negative log-likelihood in language modeling, our objective (Eq. (3)) can be viewed as an upper-bound of the true objective (Eq. (1)). Specifically, we apply $\sum_{i=1}^{m} p_\theta(i|x)h_\phi(x, i) =$

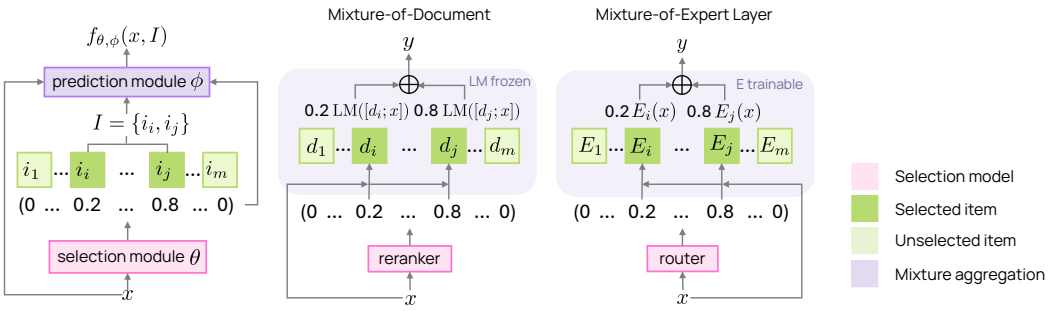


Figure 2: Our framework (left) and its applications to RAG and MoE. Middle: In RAG, given input query x, the reranker selects a subset of the most relevant documents to augment x to predict the target y. RAG has one selection and prediction module in total, with the prediction module (i.e., LM) frozen in our setting. Right: In each MoE layer, given the input hidden state x, the router selects a subset of expert models to compute the output hidden state y. The full MoE model contains multiple MoE layers of selection and aggregation, with both the selection and prediction modules trainable.

$\sum_I \pi_\theta(I|x) f_{\theta,\phi}(x, I)$ to Eq. (1) and by Jensen's inequality, for each sample x, we have

$$\mathcal{L}\left(y, \sum_I f_{\theta,\phi}(x, I) \cdot \pi_\theta(I|x)\right) \leq \sum_I \mathcal{L}(y, f_{\theta,\phi}(x, I)) \cdot \pi_\theta(I|x) = \mathbb{E}_{I\sim\pi_\theta}[\mathcal{L}(y, x, I; \theta, \phi)], \quad (4)$$

where we slightly switch notations in the last step, using $\mathcal{L}(y, x, I; \theta, \phi)$ and $\mathcal{L}(y, f_{\theta,\phi}(x, I))$ interchangeably. Since the per-sample loss $J_x(\theta, \phi) = \mathbb{E}_{I\sim\pi_\theta(\cdot|x)}[\mathcal{L}(y, x, I; \theta, \phi)]$ and the prediction module ϕ is independent of θ, we have

$$\nabla_\phi J_x(\theta, \phi) = \sum_I \pi_\theta(I|x) \nabla_\phi \mathcal{L}(y, x, I; \theta, \phi) = \mathbb{E}_{I\sim\pi_\theta}\left[\nabla_\phi \mathcal{L}(y, x, I; \theta, \phi)\right].$$

In contrast, the selection module θ affects the per-sample loss $J_x(\theta, \phi)$ in two ways: (i) via subset sampling $\pi_\theta(I|x)$ and (ii) via aggregation $f_{\theta,\phi}(x, I)$. We define leave-one-out batch loss b_x as

$$b_x := \frac{1}{|\mathcal{B}| - 1} \sum_{(x',y')\in\mathcal{B},\ (x',y')\neq(x,y)} \mathcal{L}(y', x', I; \theta, \phi) \quad \text{for each } x. \quad (5)$$

Then applying the product rule and the log-derivative trick yields

$$\nabla_\theta J_x(\theta, \phi) = \mathbb{E}_{I\sim\pi_\theta}\left[\underbrace{(\mathcal{L}(y, x, I; \theta, \phi) - b_x) \cdot \nabla_\theta \log \pi_\theta(I|x)}_{\text{policy gradient}} + \underbrace{\nabla_\theta \mathcal{L}(y, x, I; \theta, \phi)}_{\text{supervised gradient}}\right]. \quad (6)$$

In the above, we use the leave-one-out batch loss as the baseline b_x, and the full derivation and unbiasedness of Eq. (6) is proved in Appendix A. The gradient is composed of two components, the policy gradient and the supervised gradient. While such decomposition is standard for stochastic computation graphs (Kool et al., 2020) (named score functions and pathwise term in a different context), in our setting the two terms play complementary roles. The policy gradient provides training signals for *'what items to select'* and learns to explore unselected items. The supervised gradient, in contrast, provides signals for *'how to weigh each item given the selected set'* and improves exploitation.

Although Eq. (6) is the unbiased gradient, the policy gradient is usually missing (Hazimeh et al., 2021; Shazeer et al., 2017) or approximated (Liu et al., 2023; 2024) in existing literature due to the high variance of the raw policy gradient. In this work, we introduce a scalar λ in front of the policy gradient to adjust the bias/variance tradeoff: As $\lambda \to 0$, our gradient is biased with respect to the objective but of low variance; as $\lambda \to 1$, our gradient is unbiased but its variance increases. In our experiments, we conduct a small grid search over λ and show that the optimal test performance is achieved at $\lambda \in (0, 1)$.

Implementation. Our approach is easy to implement in practice. Denoting $\text{SG}(\cdot)$ as the stop-gradient operator, we can perform backpropagation on the following per-sample loss (given input (x, y)) to obtain the hybrid gradients with hyperparameter λ:

$$\texttt{loss} := \lambda\, \text{SG}(\ell_x - b_x) \cdot \log \pi_\theta(I|x) + \ell_x, \quad \text{where } \ell_x = \mathcal{L}(y, x, I; \theta, \phi),\ I \sim \pi_\theta(\cdot|x).$$

Extending to L layers. We denote L as the number of selection and prediction modules (or layers) in the model architecture. The general form of extending of our proposed Objective (3) to L selection modules is as follows. We propose to find the selection parameters $\theta^{(1:L)}$ and prediction parameters $\phi^{(1:L)}$ such that the average loss under stochastic choices of the subsets $I^{(1:L)}$ is minimized:

$$\min \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathbb{E}_{I^{(1:L)}\sim\pi_{\theta^{(1:L)}}(\cdot|x)} \left[\mathcal{L}(y, x, I^{(1:L)}; \theta^{(1:L)}, \phi^{(1:L)})\right]. \tag{7}$$

The corresponding gradients and implementation for $L > 1$ are similar to the case of $L = 1$, and we present the exact format in Section 7.

3.2 Sampling $I \sim \pi_\theta(\cdot|x)$

We now describe how we define and sample from the set-level distribution $\pi_\theta(\cdot|x)$ over subsets $I \subseteq [m]$ of size k. More generally, such subset distributions have been studied previously in the context of sequential sampling without replacement (Plackett, 1975; Kool et al., 2020). Given item scores $s_\theta(i|x)$, we first define a categorical distribution over individual items with temperature τ: $p_\theta(i|x) = \frac{\exp(s_\theta(i|x)/\tau)}{\sum_{j=1}^m \exp(s_\theta(j|x)/\tau)}$. To obtain k distinct items, we sample sequentially from $p_\theta(\cdot|x)$ without replacement, obtaining an ordered list $L = \{L_1, \ldots, L_k\}$. The probability of sampling this particular order is $p_\theta(L_1, \ldots, L_k|x) = \prod_{t=1}^k \left[\frac{p_\theta(L_t|x)}{1-\sum_{s<t} p_\theta(L_s|x)}\right]$. Since our applications use the selected items as an unordered set, we define the probability of obtaining an unordered set I by summing the probabilities of all possible orderings of its elements. Let $\mathcal{S}$ denote the set of all permutations of L. Then we have

$$\pi_\theta(I|x) = \sum_{L'\in\mathcal{S}} p_\theta(L'_1, \ldots, L'_k|x). \tag{8}$$

Lemma 3.1 (proof in Appendix A) below formalizes this construction.

Lemma 3.1 (Kool et al. (2020))**.** *Sampling I from π_θ in Eq. (8) can be implemented by* `L=multinomial(`$p_\theta(\cdot|x)$`, k, replacement=False)` *and then using L as a set.*

3.3 Application to Language Models

After the selection module chooses a subset of items I, the prediction module combines the outputs associated with the selected items. The form of this aggregation depends on the application. In our framework, RAG corresponds to a single selection stage with $L = 1$. A reranker samples a set of documents following Eq. (8), and a frozen language model computes a prediction conditioned on each selected document. These document-conditioned predictions are then aggregated based on the probabilities of each document (details in Section 6). In contrast, an MoE model contains multiple selection stages with $L > 1$. At each MoE layer, a router selects a subset of experts, and their outputs are combined to form the hidden state passed to the next layer. We discuss detailed MoE procedures in Section 7. Thus, RAG performs output-level aggregation once, whereas MoE performs hidden-state aggregation for L times throughout the network.

In the following, we discuss the proposed training procedure in Algorithm 1 for $L = 1$, and we present the natural extension for $L > 1$ in Section 7. In line 7, $\mathcal{L}$ represents the MSE loss for regression tasks (Section 5) or the cross-entropy loss for RAG and MoE applications (Sections 6−7).

4 Convergence Guarantees

In this section, we prove the convergence of our method to our surrogate objective J (Eq. (3)) and the true objective F (Eq. (1)) at rate $O(1/\sqrt{T})$, with the optimal upper-bound obtained by adjusting λ for bias/variance tradeoff. We consider optimization over all the parameters $w := (\theta, \phi)$ with $L = 1$. The selection module produces logits $z = s_\theta(x)$ and the prediction module outputs $h_\phi(x, i), i \in [m]$, which we abbreviate as $h_{i,\phi}(x)$. In the following, we present the assumptions and main theorems, with the proof detailed in Appendix B.

Assumption 1 (Compact domain)**.** *There exists a compact set Ω s.t. $w_t = (\theta_t, \phi_t) \in \Omega, \forall t \geq 0$, in which the selection module is discriminative, i.e., the minimal logit difference between the selected*

Algorithm 1 Training the Selection and Prediction Module with Hybrid Gradients

Input: Selection parameters θ, prediction parameters ϕ, η, λ, τ
for each batch $\mathcal{B}$ **do**
  for $(x, y) \in \mathcal{B}$ **do**
    $p_\theta(\cdot|x) \leftarrow \text{softmax}(s_\theta(\cdot|x)/\tau)$
    Sample k items $I \sim \text{Multinomial}(p_\theta(\cdot|x))$ without replacement ▷ Selection (§3.2)
    Compute the probability of the set $\pi_\theta(I|x)$ using Eq. (8)
    $\ell_x \leftarrow \mathcal{L}(y, x, I; \theta, \phi)$ ▷ Aggregation and compute loss (§5−7)
  end for
  Compute $b_x \leftarrow$ leave-one-out loss average for each $x \in \mathcal{B}$ by Eq. (5)
  $\ell_\mathcal{B} \leftarrow \frac{\lambda}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} \text{SG}\left(\ell_x - b_x\right) \cdot \log \pi_\theta(I|x) + \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} \ell_x$
  $[\theta, \phi] \leftarrow [\theta, \phi] - \eta \nabla_{[\theta,\phi]} \ell_\mathcal{B}$
end for

and unselected item along the optimization trajectory is positive:

$$s_{(k)}(x; \theta_t) - s_{(k+1)}(x; \theta_t) \geq \gamma_{\min} > 0, \forall x.$$

Assumption 2 (Bounded input)**.** *The input samples have compact support:* $\|x\| \leq C_x, \forall x \in \mathcal{D}$.

Assumption 3 (Loss regularity)**.** $\mathcal{L}(y, \hat{y})$ *is* $\bar{L}$*-Lipschitz and* $\bar{\beta}$*-smooth in* $\hat{y}$.

Assumption 4 (Model smoothness)**.** s_θ *and* $h_{i,\phi}, i \in [m]$ *are smooth in* θ, ϕ *and their inputs.*

We remark that standard transformer blocks with smooth activations (e.g., Hendrycks and Gimpel, 2016; Elfwing et al., 2018; Ramachandran et al., 2017) satisfy Assumption 4. In addition, under Assumptions 1−4, we prove in Appendix B that the following constants are bounded: σ_S^2 (stochastic supervised gradient variance bound), σ_P^2 (stochastic policy gradient variance bound), ϵ (upper-bound of the full-batch gradient bias $\|\nabla_w J - \nabla_w F\|$ from objective truncation), and $R_{\max}$ (the maximal full-batch policy gradient norm on the optimization trajectory).

Theorem 4.1 (Convergence)**.** *Recall that* J *denotes our surrogate objective (Eq.* (3)*) and* F *denotes the true objective (Eq.* (1)*). Under Assumptions 1-4, let* $\eta = 1/(\beta\sqrt{T})$ *and* $T \geq 4$*. For any fixed* $\lambda \in [0, 1]$*, Algorithm 1 satisfies*

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla J(w_t)\|^2 \leq \frac{2\beta(J(w_0) - J^*) + 2\sigma_S^2 + 2\lambda^2\sigma_P^2}{\sqrt{T}} + 3(1-\lambda)^2 R_{\max}^2, \tag{9}$$

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla F(w_t)\|^2 \leq \frac{2\beta(F(w_0) - F^*) + 2\sigma_S^2 + 2\lambda^2\sigma_P^2}{\sqrt{T}} + 3\big(\epsilon + (1-\lambda)R_{\max}\big)^2. \tag{10}$$

Full proofs are given in Appendix B. $\lambda = 1$ corresponds to the unbiased gradient (Eq. (6)), which gives convergence to the stationary point of J. The hybrid gradient provides a smooth bias-variance tradeoff controlled by λ: On the one hand, increasing λ reduces the bias and therefore lowers the asymptotic bias floor in the stationarity bound. On the other hand, increasing λ increases the variance of the policy gradient. In practice, the policy gradient variance can be very large, so we empirically search in a grid of three to four values in our experiments, observing that optimal λ's tend to be small.

5 Regression on Synthetic Data

To rigorously study the properties of the top-k selection problem, we design a stylized synthetic regression problem with ground-truth labels indicating which items should be selected, and instantiate $\mathcal{L}$ as a mean squared error (MSE) loss. This allows us to measure selection accuracy and diagnose the selection mechanism. Assume there are m candidate items in total, each represented by a fixed vector $v_i \in \mathbb{R}^h$ for $i \in [m]$. We generate the ground-truth selection parameters $\theta^\star \in \mathbb{R}^{m \times h}$, all the items $\{v_i\}_{i \in [m]}$, and each input feature $x \in \mathbb{R}^h$ from Gaussian distributions. For each x, we obtain the ground-truth subset indices $I_x = \arg\text{top-}k(\theta^\star x)$. We then generate optimal weights $p = \text{softmax}(\text{mask-topk}(\theta^\star x)) \in \mathbb{R}^k$ where $\text{mask-topk}(a) = a$ if $a \in$ top-k and $-\infty$

otherwise. Finally, the label y for input x is generated via a linear combination of these k items, i.e., $y = \sum_{i \in I_x} p_i v_i \in \mathbb{R}^h$. Given $\{v_i\}_{i \in [m]}$ and $\mathcal{D} = \{(x, y)\}$, we aim to learn the selection parameters θ so as to minimize the empirical MSE loss $\frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \|y - \sum_{i \in I_x} p_i v_i\|^2$, $|I_x| = k$.

Data, models, and baselines. We take $h = 128, k = 3$, and $m \in \{64, 128, 256, 512\}$. We generate 10K training samples, 1K validation samples, and 1K test samples. We use the linear selection model $s_\theta : x \to \theta x$ and aggregate the selected hidden states via $f_{\theta,\phi}(x, I) = \sum_{i \in I_x} \frac{e^{s_\theta(i|x)}}{\sum_{j \in I_x} e^{s_\theta(j|x)}} v_i$. We compare with five classic baselines in the top-k selection literature, which are also common methods in MoE work. Details of methods and hyperparameter tuning are in Appendix C.1. The baselines are: (1) Deterministic top-k which selects the k items based on the raw scores, (2) Gaussian top-k (Shazeer et al., 2017), which adds Gaussian noise to item scores with learnt noise variance, (3) DefaultMoE (Panda et al., 2025), which uses each expert's running average of outputs for aggregation when the experts are not selected, (4) SparseMixer (Liu et al., 2023), which uses high-order derivatives to approximate the policy gradient, and (5) Supervised-only, which is implemented by the classic Gumbel-Top-k trick (Kool et al., 2019) and corresponds to our method with $\lambda = 0$.

We report the MSE loss and the selection accuracies on this toy problem, as we have access to the ground-truth selected items. In Figure 3, we present the average performance of the five selection baselines under different m's. We observe that as m increases, the optimization becomes more challenging: The test top-1 selection accuracy drops drastically, and test MSE loss increases. Furthermore, in Figure 4, we show that our method outperforms all baselines in terms of top-1 selection accuracy and MSE loss under the optimal λ. The exact numerical values are reported in Table 1, and an ablation on λ is presented in Figure 9 in the appendix.

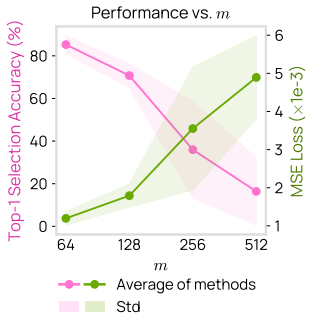


Figure 3: Selection module is harder to optimize as m increases.

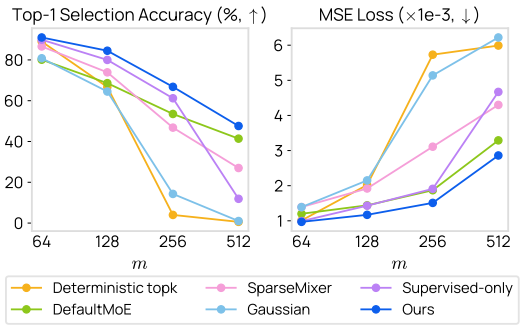


Figure 4: Our method outperforms baselines on the regression problem in test top-1 selection accuracy and MSE loss.

We further visualize the evolution of θ's selection decision (on two test samples) during optimization with different methods (Figure 8). Our method traverses more diverse selection decisions than the deterministic top-k and supervised-only baselines, which illustrates the benefits of the policy gradient.

6 Document Selection for Retrieval-Augmented Generation

In this section, we apply our method to in-context RAG, which generally involves three model components: a retriever that selects m documents from a datastore of size M, a reranker that selects k documents out of m, and a language model that performs next-token (or next-sequence) prediction based on the concatenation of the original query and the selected document(s). We consider the common setting of training the reranker θ only (Shi et al., 2024; Ram et al., 2023).

Denote x as the query sequence and y as the ground-truth continuation of x. Given (x, y), the reranker produces a score $s_\theta(i|x)$ for each document $i \in [m]$, indicating the relevance of each document to the query x. The reranker next chooses a set of documents $I = \{I_1, \ldots, I_k\}$ via sampling following Eq. (8). For each selected document I_i, we concatenate it with query x to form $[I_i; x]$ and compute $p_{\text{LM}}(y|x, I_i)$, the probability of the target continuation given input $[I_i; x]$. Then the output $f_{\theta,\phi}(x, I)$

is the probability of the continuation y computed by

$$p(y|x,I) = \sum_{i=1}^{k} a_i(x,I) \cdot p_{\text{LM}}(y|x,I_i), \quad a_i(x,I) = \frac{\sum_{L:\{L_1,\ldots,L_k\}=I, L_1=i} \pi_\theta(L|x)}{\pi_\theta(I|x)}. \tag{11}$$

The proof that our choice of $a_i(x,I)$ in Eq. (11) makes our objective upper-bound the true objective is in Appendix C.2. In our experiments, we follow prior work (Shi et al., 2024; Ram et al., 2023) to take both x and y as a sequence of s tokens for retrieval efficiency. In language modeling, we use $s = 128$, so the task is to predict the next-chunk based on the previous chunk, consistent with Shi et al. (2024). In question answering, x is the question and y is the answer.

We fine-tune the reranker MS Marco[1], whose selected documents are concatenated with the original query and fed as input to generative models, including Qwen-2.5-1.5B (Team, 2024) and Llama-3.2-3B (Grattafiori et al., 2024). We evaluate our method and the baselines on Wikitext-103 (Merity et al., 2016) and open-domain QA datasets, including Natural Questions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017). Following prior work (Guu et al., 2020a; Shi et al., 2024), we use documents from Wikipedia (2018, December) as our retrieval datastore and use BM25 (Robertson and Zaragoza, 2009) as the retriever that pre-selects the 32 most relevant passages for each training query.

We compare our method with the following baselines, whose algorithms are detailed in Appendix C.2: (1) Frozen reranker that yields zero-shot performance; (2) The classic ICRALM objective (Eq. (17)) (Ram et al., 2023; Lewis et al., 2020; Guu et al., 2020a), which performs deterministic top-k; (3) REPLUG (Shi et al., 2024), which matches the reranker's score distribution with LM's perplexity distribution conditioned on each selected document (Eq. (19)); (4) Policy-only (Basu et al., 2024), which samples i.i.d. for k times from the policy defined on 'items' and updates with the policy gradient (Eq. (21)); (5) Supervised-only, which is a new method in RAG and a special case of our method with $\lambda = 0$. In Figure 5, we report the test performance on Wikitext-103 (in perplexity) and NQ and TriviaQA (in exact match accuracy). We observe more performance gains as m increases, and our method outperforms all the baselines under all $m \in \{8, 16, 32\}$ for both Qwen-2.5-1.5B and Llama-3.2-3B. Furthermore, we evaluate the document recall on NQ and TriviaQA (Figure 10), observing up to a 3.5% increase from our method compared to methods in existing literature.

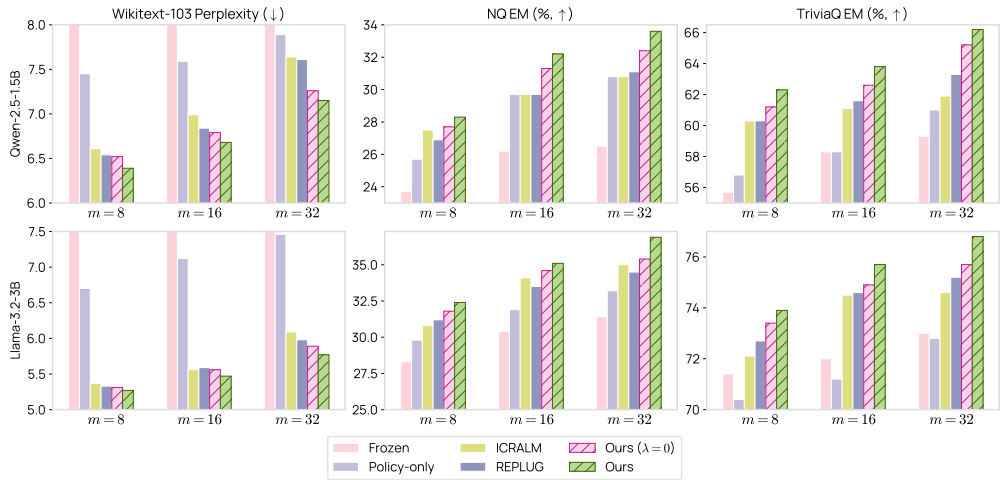


Figure 5: Performance of retrieval-augmented Qwen-2.5-1.5B and Llama-3.2-3B with the reranker trained via different methods. We observe improved performance as m increases, and our method outperforms all the baselines. The detailed metric values are reported in Tables 2-6.

7 Expert Selection for Mixture of Experts

In this section, we apply our method to MoE models, which contain multiple MoE layers and each layer has one router and m experts. In the l-th MoE layer, we have an input $x^{(l)} \in \mathbb{R}^{d^{(l)}}$ where $d^{(l)}$ is the hidden dimension, the experts $h_1^{(l)}, \ldots, h_m^{(l)}$ parameterized by weights $\phi^{(l)}$, and a router $\theta^{(l)}$.

[1] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L12-v2

Given input $x^{(l)}$, the router outputs score $s_{\theta^{(l)}}(i|x^{(l)})$ for all experts $i \in [m]$. Denote $I^{(l)}$ as the sampled set of the indices of the selected unique experts from policy $\pi_{\theta^{(l)}}$. Then this MoE layer outputs

$$y^{(l)} = \sum_{i \in I^{(l)}} \frac{\exp(s_{\theta^{(l)}}(i|x^{(l)}))}{\sum_{j \in I^{(l)}} \exp(s_{\theta^{(l)}}(j|x^{(l)}))} h_i^{(l)}(x^{(l)}).$$

Therefore, for any sample (x, y) where x is the input sequence and y is the target token, we optimize the router parameters $\theta := \{\theta^{(1)}, \ldots, \theta^{(L)}\}$ and expert parameters $\phi := \{\phi^{(1)}, \ldots, \phi^{(L)}\}$ for a total of L MoE layers via minimizing the loss Eq. (7). We have that for an arbitrary layer l,

$$\nabla_{\theta^{(l)}} J_x(\theta) = \mathbb{E}_{I^{(1:L)}} \left[(\mathcal{L}(y, x, I^{(1:L)}) - b_x) \cdot \sum_{n=l}^{L} \nabla_{\theta^{(n)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)}) + \nabla_{\theta^{(l)}} \mathcal{L}(y, x, I^{(1:L)}) \right].$$

That is, to compute an unbiased estimate of the gradient, we only need to sample $I^{(1)} \sim \pi_{\theta^{(1)}}, \ldots, I^{(L)} \sim \pi_{\theta^{(L)}}$ and compute one forward pass. We present the full proof and the multi-layer version of Algorithm 1 as Algorithm 2 in Appendix A.

Data, models, and baselines. We train MoE models of nine architectures on Wikitext-103 (Merity et al., 2016) from scratch, including the total expert numbers $m \in \{8, 16, 32, 64, 128\}$ with one or two MoE layers. To ensure that our observed improvements are systematic, we evaluate our method in a over-training setting (820M tokens, with around 230 tokens-per-activated-parameter) to mitigate spurious gains, providing a rigorous setup for algorithmic comparison.

We compare our method with the same baselines in Section 5, whose methods and hyperparameters are tuned based on Appendix C: (1) Deterministic top-k; (2) Gaussian MoE (Shazeer et al., 2017); (3) Default MoE (Panda et al., 2025); (4) SparseMixer (Liu et al., 2023); (5) Supervised-only (Zibakhsh et al., 2025) (our method with $\lambda = 0$). We observe that scaling up m improves model performance, and our method outperforms all the baselines under all m. In addition, our method is related to Supervised-only, which is our special case, and SparseMixer, which approximates the policy gradient with high-order derivatives to reduce variance. We show that our λ is an effective knob for bias/variance tradeoff: the best $\lambda \in (0, 1)$ outperforms Supervised-only ($\lambda = 0$) by 2.0 points and outperforms the unbiased update rule ($\lambda = 1$) by 7.0 points in perplexity in the case of '128 choose 3', and this observation is consistent in other m's, as presented in Figure 12.

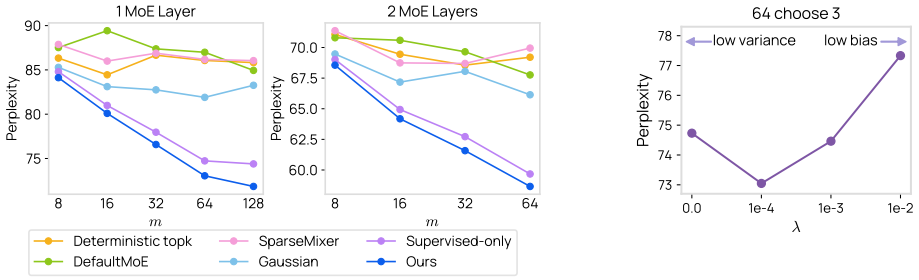


Figure 6: Test perplexity of different methods after training from scratch on Wikitext-103. Left: Our method outperforms baselines under different numbers of total experts. Right: The best bias/variance tradeoff is achieved by adjusting λ. The exact numerical values are reported in Table 7.

8 Conclusion and Limitations

We study the problem of top-k selection and propose a principled surrogate objective that models subset selection and upper-bounds the true objective under certain conditions. Our unbiased gradient naturally decomposes into two complementary components: the supervised gradient improves aggregation within the selected subset, and the policy gradient provides direct learning signals on which items to select. We provide convergence results, and demonstrate consistent improvements upon strong baselines across regression, RAG, and MoE training in language modeling and QA tasks. We identify two limitations of this work. First, while our analysis characterizes the bias/variance tradeoff, selecting λ remains problem-dependent in practice. Second, our experiments are conducted on specific model sizes and datasets, and we leave further evaluation at larger scales to future work.

AI USE STATEMENT

In this work, we used generative AI tools for polishing writing and verifying the correctness of our proofs. We have not used generative AI tools for providing critical ingredients for proving mathematical claims, designing research methodology, or implementing experiments. The rest of the required disclosure tasks, including generating synthetic data, cleaning and reformatting the dataset, and assisting with translation, are not applicable to this work. Additionally, we used generative AI tools to expand our knowledge of related work in case we missed relevant papers. We have reviewed all AI-assisted work—we checked LLM-suggested related work through a manual literature survey. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We describe the experimental setup, including dataset details, model configurations, hyperparameters, and evaluation protocols in the main text and appendices. Our implementation, data processing scripts, and training and evaluation code will be released upon acceptance.

REFERENCES

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020a.

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. *arXiv preprint arXiv:1701.06538*, 2017.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1280–1297, 2024.

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.

Liyuan Liu, Jianfeng Gao, and Weizhu Chen. Sparse backpropagation for moe training. *arXiv preprint arXiv:2310.00811*, 2023.

Ashwinee Panda, Vatsal Baherwani, Zain Sarwar, Benjamin Therien, Sambit Sahu, Tom Goldstein, and Supriyo Chakraborty. Dense backpropagation improves training for sparse mixture-of-experts. *arXiv preprint arXiv:2504.12463*, 2025.

Soheil Zibakhsh, Mohammad Samragh, Kumari Nishu, Lauren Hannah, Arnav Kundu, and Minsik Cho. Moes are stronger than you think: Hyper-parallel inference scaling with roe. *arXiv preprint arXiv:2509.17238*, 2025.

Liyuan Liu, Young Jin Kim, Shuohang Wang, Chen Liang, Yelong Shen, Hao Cheng, Xiaodong Liu, Masahiro Tanaka, Xiaoxia Wu, Wenxiang Hu, et al. Grin: Gradient-informed moe. *arXiv preprint arXiv:2409.12136*, 2024.

Max Paulus, Dami Choi, Daniel Tarlow, Andreas Krause, and Chris J Maddison. Gradient estimation with stochastic softmax tricks. *Advances in Neural Information Processing Systems*, 33:5691–5704, 2020.

Yujia Xie, Hanjun Dai, Minshuo Chen, Bo Dai, Tuo Zhao, Hongyuan Zha, Wei Wei, and Tomas Pfister. Differentiable top-k with optimal transport. *Advances in neural information processing systems*, 33:20520–20531, 2020.

Robin Swezey, Aditya Grover, Bruno Charron, and Stefano Ermon. Pirank: Scalable learning to rank via differentiable sorting. *Advances in Neural Information Processing Systems*, 34:21644–21654, 2021.

Łukasz Struski, Michał B Bednarczyk, Igor T Podolak, and Jacek Tabor. Lapsum–one method to differentiate them all: Ranking, sorting and top-k selection. *arXiv preprint arXiv:2503.06242*, 2025.

Yanjie Zhu, Zhen Zhang, Yunli Wang, Zhiqiang Wang, Yu Li, Rufan Zhou, Shiyang Wen, Peng Jiang, Chenhao Lin, and Jian Yang. Differentiable fast top-k selection for large-scale recommendation. *arXiv preprint arXiv:2510.11472*, 2025.

Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. *arXiv preprint arXiv:1711.01239*, 2017.

Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020b.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: main volume*, pages 874–880, 2021.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424*, 2019.

Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. Rankt5: Fine-tuning t5 for text ranking with ranking losses. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pages 2308–2313, 2023.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, et al. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518, 2024.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8371–8384, 2024.

Yuan Li, Qi Luo, Xiaonan Li, Bufan Li, Qinyuan Cheng, Bo Wang, Yining Zheng, Yuxin Wang, Zhangyue Yin, and Xipeng Qiu. R3-rag: Learning step-by-step reasoning and retrieval for llms via reinforcement learning. *arXiv preprint arXiv:2505.23794*, 2025.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, 2023.

Xuan Zhang and Wei Gao. Reinforcement retrieval leveraging fine-grained feedback for fact checking news claims with black-box llm. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 13861–13873, 2024.

Soumya Basu, Ankit Singh Rawat, and Manzil Zaheer. A statistical framework for data-dependent retrieval-augmented models. *arXiv preprint arXiv:2408.15399*, 2024.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

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.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2020.

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.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models, 2022. *URL https://arxiv. org/abs/2202.08906*, 2022.

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.

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.

Wouter Kool, Herke van Hoof, and Max Welling. Estimating gradients for discrete random variables by sampling without replacement. *arXiv preprint arXiv:2002.06043*, 2020.

Hussein Hazimeh, Zhe Zhao, Aakanksha Chowdhery, Maheswaran Sathiamoorthy, Yihua Chen, Rahul Mazumder, Lichan Hong, and Ed Chi. Dselect-k: Differentiable selection in the mixture of experts with applications to multi-task learning. *Advances in Neural Information Processing Systems*, 34:29335–29347, 2021.

Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11, 2018.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International conference on machine learning*, pages 3499–3508. PMLR, 2019.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL `https://qwenlm.github.io/blog/qwen2.5/`.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, 2017.

Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc, 2009.

Malcolm R Leadbetter, Georg Lindgren, and Holger Rootzén. *Extremes and related properties of random sequences and processes*. Springer Science & Business Media, 2012.

Laurens De Haan and Ana Ferreira. *Extreme value theory: an introduction*. Springer, 2006.

A METHOD

A.1 GRADIENT DERIVATION

Case of $L = 1$.

$$\begin{aligned}
\nabla_\theta J_x(\theta,\phi) &= \nabla_\theta \sum_I \pi_\theta(I|x)\mathcal{L}(y,x,I;\theta,\phi) \\
&= \sum_I \nabla_\theta \pi_\theta(I|x)\mathcal{L}(y,x,I;\theta,\phi) + \pi_\theta(I|x)\nabla_\theta \mathcal{L}(y,x,I;\theta,\phi) \\
&\overset{(a)}{=} \sum_I \pi_\theta(I|x)\nabla_\theta \log \pi_\theta(I|x)\mathcal{L}(y,x,I;\theta,\phi) + \pi_\theta(I|x)\nabla_\theta \mathcal{L}(y,x,I;\theta,\phi) \\
&= \mathbb{E}_{I\sim\pi_\theta}\left[\mathcal{L}(y,x,I;\theta,\phi)\cdot\nabla_\theta \log \pi_\theta(I|x) + \nabla_\theta \mathcal{L}(y,x,I;\theta,\phi)\right] \\
&\overset{(b)}{=} \mathbb{E}_{I\sim\pi_\theta}\left[\underbrace{(\mathcal{L}(y,x,I;\theta,\phi) - b_x)\cdot\nabla_\theta \log \pi_\theta(I|x)}_{\text{policy gradient}} + \underbrace{\nabla_\theta \mathcal{L}(y,x,I;\theta,\phi)}_{\text{supervised gradient}}\right].
\end{aligned}$$

where step (a) applies the log-derivative trick $\nabla_\theta \pi_\theta = \pi_\theta \nabla_\theta \log \pi_\theta$ and step (b) uses leave-one-out average batch loss as the baseline b_x for any x. Abbreviating $\ell_x := \mathcal{L}(y,x,I;\theta,\phi)$, step (b) holds due to

$$\begin{aligned}
&\mathbb{E}_{I\sim\pi_\theta}\left[(\ell_x - b_x)\cdot\nabla_\theta \log \pi_\theta(I|x)\right] \\
&= \mathbb{E}_{I\sim\pi_\theta}\left[\ell_x \cdot \nabla_\theta \log \pi_\theta(I|x)\right] - \mathbb{E}_{I\sim\pi_\theta}\left[\left(\frac{1}{|\mathcal{B}|-1}\sum_{x'\in\mathcal{B},x'\neq x}\ell_{x'}\right)\cdot\nabla_\theta \log \pi_\theta(I|x)\right] \\
&= \mathbb{E}_{I\sim\pi_\theta}\left[\ell_x \cdot \nabla_\theta \log \pi_\theta(I|x)\right] - \left(\frac{1}{|\mathcal{B}|-1}\sum_{x'\in\mathcal{B},x'\neq x}\ell_{x'}\right)\underbrace{\mathbb{E}_{I\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(I|x)\right]}_{=0} \\
&= \mathbb{E}_{I\sim\pi_\theta}\left[\ell_x \cdot \nabla_\theta \log \pi_\theta(I|x)\right].
\end{aligned}$$

Case of $L > 1$. For any sample (x, y) where x is the input sequence and y is the target token, we optimize the router parameters $\theta := \{\theta^{(1)},\ldots,\theta^{(L)}\}$ and expert parameters $\phi := \{\phi^{(1)},\ldots,\phi^{(L)}\}$ for a total of L MoE layers via minimizing the loss

$$J_x(\theta,\phi) = \mathbb{E}_{I^{(1)},\ldots,I^{(L)}}\left[\mathcal{L}(y,x,I^{(1)},\ldots,I^{(L)}|\theta,\phi)\right].$$

We denote $I^{(1)}\ldots I^{(L)}$ as $I^{(1:L)}$ and $\mathcal{L}(\ldots|\theta,\phi)$ as $\mathcal{L}(\ldots)$ for brevity. By the law of iterated expectation, $\mathbb{E}_{I^{(1:L)}}[\cdot] = \mathbb{E}_{I^{(1)}}\left[\mathbb{E}_{I^{(2)}|I^{(1)}}\left[\cdots\mathbb{E}_{I^{(L)}|I^{(1:L-1)}}[\cdot]\right]\right]$. Thus, for an arbitrary layer l,

$$\begin{aligned}
&\nabla_{\theta^{(l)}} J_x(\theta) \\
&= \nabla_{\theta^{(l)}}\mathbb{E}_{I^{(1:l-1)}}\left[\mathbb{E}_{I^{(l)}|I^{(1:l-1)}}\left[\underbrace{\mathbb{E}_{I^{(l+1:L)}|I^{(1:l)}}[\mathcal{L}(y,x,I^{(1:L)})]}_{g(I^{(1:l)})}\right]\right] \\
&= \mathbb{E}_{I^{(1:l-1)}}\left[\nabla_{\theta^{(l)}}\ \mathbb{E}_{I^{(l)}|I^{(1:l-1)}}\left[g(I^{(1:l)})\right]\right] \\
&\overset{(a)}{=} \mathbb{E}_{I^{(1:l-1)}}\left[\sum_{I^{(l)}}\nabla_{\theta^{(l)}}\left(\pi_{\theta^{(l)}}(I^{(l)}|x^{(l)})g(I^{(1:l)})\right)\right] \\
&= \mathbb{E}_{I^{(1:l-1)}}\left[\sum_{I^{(l)}}\nabla_{\theta^{(l)}}\pi_{\theta^{(l)}}(I^{(l)}|x^{(l)})g(I^{(1:l)}) + \pi_{\theta^{(l)}}(I^{(l)}|x^{(l)})\nabla_{\theta^{(l)}}g(I^{(1:l)})\right] \\
&= \mathbb{E}_{I^{(1:l)}}\left[g(I^{(1:l)})\cdot\nabla_{\theta^{(l)}}\log\pi_{\theta^{(l)}}(I^{(l)}|x^{(l)}) + \nabla_{\theta^{(l)}}g(I^{(1:l)})\right].
\end{aligned}$$

where (a) is due to $p(I^{(l)}|I^{(1:l-1)};\theta^{(1:l-1)}) = \pi_{\theta^{(l)}}(I^{(l)}|x^{(l)})$. That is, under the given $\theta^{(1:l-1)}$, the mapping from the past choices $I^{(1:l-1)}$ to the current state $x^{(l)}$ is deterministic.

In the above derivation, note that

$$\begin{aligned} g(I^{(1:l)}) &= \mathbb{E}_{I^{(l+1:L)}}\left[\mathcal{L}(y,x,I^{(1:L)})|I^{(1:l)} \text{ is sampled}\right] \\ &= \sum_{I^{(l+1:L)}} p(I^{(l+1:L)}|I^{(1:l)})\mathcal{L}(y,x,I^{(1:L)}) \end{aligned}$$

is the expected loss $\mathcal{L}$ from the perspective of layer l, after the router has chosen the set $I^{(l)}$, but before the future layers have made their stochastic choices. Denoting $\ell_x := \mathcal{L}(y,x,I^{(1:L)})$, we have

$$p(I^{(l+1:L)}|I^{(1:l)}) = \prod_{n=l+1}^{L} \pi_{\theta^{(n)}}\left(I^{(n)}|x^{(n)}\right);$$

$$\nabla_{\theta^{(l)}} g(I^{(1:l)}) = \mathbb{E}_{I^{(l+1:L)}|I^{(1:l)}}\left[\nabla_{\theta^{(l)}}\ell_x + \ell_x \sum_{n=l+1}^{L} \nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right].$$

Therefore, the full derivative is

$$\begin{aligned} &\nabla_{\theta^{(l)}} J_x(\theta) \\ &= \mathbb{E}_{I^{(1:L)}}\left[\ell_x \cdot \nabla_{\theta^{(l)}} \log \pi_{\theta^{(l)}}(I^{(l)}|x^{(l)})\right] + \mathbb{E}_{I^{(1:L)}}\left[\nabla_{\theta^{(l)}}\ell_x + \ell_x \sum_{n=l+1}^{L} \nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right] \\ &= \mathbb{E}_{I^{(1:L)}}\left[\underbrace{\mathcal{L}(y,x,I^{(1:L)}) \cdot \sum_{n=l}^{L} \nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})}_{\text{policy gradient}} + \underbrace{\nabla_{\theta^{(l)}}\mathcal{L}(y,x,I^{(1:L)})}_{\text{supervised gradient}}\right]. \end{aligned} \tag{12}$$

That is, an unbiased estimate of the gradient is composed of the policy part and the supervised part, and it can be obtained by sampling $I^{(1)} \sim \pi_{\theta^{(1)}}, \ldots, I^{(L)} \sim \pi_{\theta^{(L)}}$ and computing one forward pass. We can again add leave-one-out average batch loss as baseline b_x for any x while keeping the the gradient unbiased: Abbreviating $\ell_{x,I^{(1:L)}} := \mathcal{L}(y,x,I^{(1:L)})$, for $\forall\, n : l \le n \le L$, we have

$$\begin{aligned} &\mathbb{E}_{I^{(1:L)}}\left[b_x \cdot \nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right] \\ &= \mathbb{E}_{I^{(1:L)}}\left[\left(\frac{1}{|\mathcal{B}|-1}\sum_{x'\in\mathcal{B},x'\neq x} \ell_{x',I^{(1:L)}}\right) \cdot \nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right] \\ &= \left(\frac{1}{|\mathcal{B}|-1}\sum_{x'\in\mathcal{B},x'\neq x} \ell_{x',I^{(1:L)}}\right) \mathbb{E}_{I^{(1:L)}}\left[\nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right] \end{aligned}$$

where

$$\begin{aligned} \mathbb{E}_{I^{(1:L)}}\left[\nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right] &= \mathbb{E}_{I^{(1:n-1)}}\left[\mathbb{E}_{I^{(n)}|I^{(1:n-1)}}\left[\nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right]\right] \\ &= \mathbb{E}_{I^{(1:n-1)}}\left[\sum_{I^{(n)}} \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)}) \nabla_{\theta^{(l)}} \log \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right] \\ &= \mathbb{E}_{I^{(1:n-1)}}\left[\sum_{I^{(n)}} \nabla_{\theta^{(l)}} \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right] \\ &= \mathbb{E}_{I^{(1:n-1)}}\left[\nabla_{\theta^{(l)}}\left(\sum_{I^{(n)}} \pi_{\theta^{(n)}}(I^{(n)}|x^{(n)})\right)\right] \\ &= 0. \end{aligned}$$

We thus have Algorithm 2 for the case of $L > 1$. We note that by summing the policy gradient from all layers $l \in [L]$, we only need to run backpropagation to the loss in Line 10 once.

Algorithm 2 Training the Selection and Prediction Module with Hybrid Gradients ($L > 1$)

Input: Selection parameters θ, prediction parameters ϕ, η, λ, τ
for each batch $\mathcal{B}$ **do**
  for $(x, y) \in \mathcal{B}$ **do**
    $p_{\theta^{(l)}}(\cdot|x) \leftarrow \text{softmax}(s_{\theta^{(l)}}(\cdot|x)/\tau)$ for all $l \in [L]$
    Sample $I^{(l)} \sim \text{Multinomial}(p_{\theta^{(l)}}(\cdot|x))$ without replacement for all $l \in [L]$
    Compute $\log \pi_{\theta^{(l)}}(I^{(l)}|x^{(l)})$ by Eq. (8) for all $l \in [L]$
    $\ell_x \leftarrow \mathcal{L}(y, x, I^{(1:L)}|\theta^{(1:L)}, \phi^{(1:L)})$
  end for
  Compute $b_x \leftarrow$ leave-one-out loss average for each $x \in \mathcal{B}$ by Eq. (5)
  $\ell_{\mathcal{B}} \leftarrow \frac{\lambda}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} \sum_{l \in [L]} \text{SG}\left(\ell_x - b_x\right) \cdot \log \pi_{\theta^{(l)}}(I^{(l)}|x^{(l)}) + \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} \ell_x$
  $[\theta, \phi] \leftarrow [\theta, \phi] - \eta \nabla_{[\theta,\phi]} \ell_{\mathcal{B}}$
end for

A.2 Selection Module Policy

Lemma A.1. *Sampling I from π_θ in Eq. (8) can be implemented by* `L=multinomial(`$p_\theta(\cdot|x)$`, k, replacement=False)` *and then using L as a set. In other words, if $\pi(I)$ is the probability of I where I is an unordered version of an ordered sequence, then sampling an ordered sequence and then forgetting the order follows exactly $\pi(I)$.*

Proof. Let the sampled ordered sequence be $l = \{l_1, \ldots, l_k\}$ and denote its unordered set version as $i = \text{set}(l)$. Denote $\text{perm}(i)$ as a set containing all permutations (ordered sequences) of the elements in i. Then by the definition of marginalization,

$$P(I = i|x) = \sum_{l' \in \text{perm}(i)} p(l'_1, \ldots, l'_k|x) \overset{(8)}{=} \pi_\theta(I = i|x).$$

□

A.3 Recipe for Selecting λ

Theoretically, the optimal $\lambda^\star$ value would be small if the policy gradient variance bound σ_P^2 is large (Theorem 4.1). Empirically, however, selecting λ remains problem-dependent. To reduce the tuning workload, we study how to quickly determine whether a λ value is too big or too small.

We compare the gradient norm and the loss standard deviation of the supervised component and the policy component, respectively. We observe that both metrics are informative indicators of appropriate λ values: As shown in Figure 7, large λ leads to an overwhelming policy gradient norm and policy loss standard deviation compared to the supervised gradient. In contrast, small λ yields a small contribution from the policy component. Under appropriate λ, the gradient norm and loss standard deviation from the two components are comparable.

Based on this observation, we provide a recipe for quickly searching for the largest appropriate λ: First, pick a λ and examine the first few steps of the optimization; then reduce λ by $10\times$ each time if the policy gradient norm or loss standard deviation overwhelms that of the supervised component.

B Convergence Analysis

This section proves the convergence of our method to our objective and the true objective. In the following, we first present the problem formulation and prove the useful lemmas.

B.1 Problem Setup

Let the training set be $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$. We train prediction module parameters ϕ jointly with the selection module parameters θ and let $w := (\theta, \phi)$. The selection module $s_\theta : \mathbb{R}^h \to \mathbb{R}^m$ produces

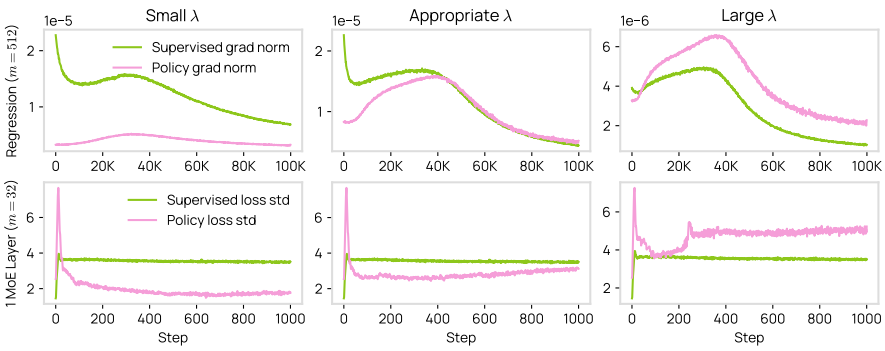


Figure 7: Gradient norm and loss standard deviation of the supervised component and the policy component. Under appropriate λ, the two metrics for the two components are comparable.

scores $z_i := s_\theta(x)_i, i \in [m]$. The empirical true objective is

$$F(\theta, \phi) := \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}\left(y_n, \sum_{i=1}^{m} \frac{e^{s_\theta(x_n)_i}}{\sum_{j=1}^{m} e^{s_\theta(x_n)_j}} h_{i,\phi}(x_n)\right).$$

Our surrogate. For each training sample x_n, let $I \sim \pi_\theta(\cdot|x_n)$ be the sampled subset. Define

$$\ell_n(\theta, \phi, I) := \mathcal{L}\left(y_n, \sum_{i \in I} \frac{e^{s_\theta(x_n)_i}}{\sum_{j \in I} e^{s_\theta(x_n)_j}} h_{i,\phi}(x_n)\right)$$

and our surrogate objective is

$$J(\theta, \phi) := \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{I \sim \pi_\theta(\cdot|x_n)}[\ell_n(\theta, \phi, I)].$$

Recall that the gradient of J with respect to θ decomposes into the policy gradient $P_\theta(\theta, \phi)$ and the supervised gradient $S_\theta(\theta, \phi)$, i.e., $\nabla_\theta J(\theta, \phi) = P_\theta(\theta, \phi) + S_\theta(\theta, \phi)$ where

$$P_\theta(\theta, \phi) := \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{I \sim \pi_\theta(\cdot|x_n)}[\ell_n(\theta, \phi, I) \nabla_\theta \log \pi_\theta(I|x_n)],$$

$$S_\theta(\theta, \phi) := \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{I \sim \pi_\theta(\cdot|x_n)}[\nabla_\theta \ell_n(\theta, \phi, I)].$$

Since π_θ does not depend on ϕ,

$$\nabla_\phi J(\theta, \phi) = S_\phi(\theta, \phi) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{I \sim \pi_\theta(\cdot|x_n)}[\nabla_\phi \ell_n(\theta, \phi, I)].$$

Minibatch update. At iteration t, we sample a minibatch $\mathcal{B}_t$ i.i.d. uniformly from $\mathcal{D}$ with replacement. For each $b \in \mathcal{B}_t$, we sample $I_b \sim \pi_{\theta_t}(\cdot|x_b)$. Using a leave-one-out batch baseline,

$$b_{-b} := \frac{1}{B-1} \sum_{a \neq b} \ell_a(\theta_t, \phi_t, I_a),$$

we use the selection-model update

$$g^{\lambda}_{\theta,t} = \frac{1}{B} \sum_{b=1}^{B} \left[\lambda\big(\ell_b(\theta_t, \phi_t, I_b) - b_{-b}\big) \nabla_\theta \log \pi_{\theta_t}(I_b|x_b) + \nabla_\theta \ell_b(\theta_t, \phi_t, I_b)\right].$$

and the prediction-model update

$$g_{\phi,t} = \frac{1}{B}\sum_{b=1}^{B} \nabla_\phi \ell_b.$$

The full-model gradient is denoted as $g_t^\lambda := (g_{\theta,t}^\lambda, g_{\phi,t})$, with which we conduct the joint update $w_{t+1} = w_t - \eta g_t^\lambda$. Note that the joint gradient estimator is related to the gradient of our objective via

$$\mathbb{E}[g_t^\lambda | w_t] = \nabla_w J(w_t) + \tilde{b}_t^\lambda, \quad \tilde{b}_t^\lambda = \big(-(1-\lambda)P_\theta(w_t), 0\big).$$

Relative to the true objective F, our update satisfies $g_t^\lambda = \nabla_w F(w_t) + b_t^\lambda + \xi_t$, where ξ_t is the gradient variance with $\mathbb{E}[\xi_t | w_t] = 0$ and

$$b_t^\lambda = \underbrace{\nabla_w J(w_t) - \nabla_w F(w_t)}_{\text{truncation bias}} + \underbrace{\big(-(1-\lambda)P_\theta(w_t), 0\big)}_{\text{policy gradient suppression bias}}.$$

In the following, we first derive the lemmas showing that the constants σ_S^2 (supervised gradient variance bound), σ_P^2 (policy gradient variance bound), ϵ (upper-bound of the gradient bias $\|\nabla_w J - \nabla_w F\|$ from objective truncation), and $R_{\max}$ (the maximal full-batch policy gradient norm on the optimization trajectory) are finite under Assumptions 1−4.

B.2 Lemmas

Lemma B.1 (Derived constants)**.** *Under Assumptions 1–4, the following quantities are finite:*

(a) ***Prediction module output.*** $C_h := \max_{i\in[m]} \sup_{w\in\Omega, \|x\|\le C_x} \|h_{i,\phi}(x)\| < \infty$.

(b) ***Prediction module Jacobian.*** $C_{h'} := \max_{i\in[m]} \sup_{w\in\Omega, \|x\|\le C_x} \|\nabla_\phi h_{i,\phi}(x)\| < \infty$.

(c) ***Selection module Jacobian.*** $C_r := \max_{i\in[m]} \sup_{w\in\Omega, \|x\|\le C_x} \left\|\frac{\partial z_i}{\partial \theta}(x;\theta)\right\| < \infty$.

(d) ***Score function.*** $C_\pi := \sup_{w\in\Omega, \|x\|\le C_x, I:|I|=k} \|\nabla_\theta \log \pi_\theta(I|x)\| < \infty$.

(e) ***Loss value.*** $\ell_{\max} := \sup_{w\in\Omega, \|x\|\le C_x, I:|I|=k} \ell_n(\theta,\phi,I) < \infty$.

(f) ***Loss variance.*** $\sigma_l^2 := \sup_{w\in\Omega} \mathrm{Var}_{n,I}(\ell_n(\theta,\phi,I)) \le \sigma_l^2 < \infty$.

(g) ***Policy gradient.*** $R_{\max} := \sup_{w\in\Omega} \|P_\theta(w)\| < \infty$.

Proof. Parts (a)–(c) follow from the same argument: each quantity is the supremum of a continuous function over the compact set $\Omega \times \{x : \|x\| \le C_x\}$. By Assumption 4, $h_{i,\phi}(x)$, $\nabla_\phi h_{i,\phi}(x)$, and $\partial z_i/\partial\theta$ are all continuous in (w, x). A continuous function on a compact set attains its supremum, so each quantity is finite.

To prove Part (d), recall that

$$\pi_\theta(I|x) = \sum_{L\in\mathrm{perm}(I)} \prod_{t=1}^{k} \frac{p_\theta(L_t|x)}{1-\sum_{s<t} p_\theta(L_s|x)}, \qquad p_\theta(L_t|x) = \mathrm{softmax}(z/\tau).$$

By the chain rule,

$$\nabla_\theta \log \pi_\theta(I|x) = \sum_{i=1}^{m} \frac{\partial \log \pi_\theta(I|x)}{\partial z_i}\frac{\partial z_i}{\partial \theta}.$$

Hence,

$$\|\nabla_\theta \log \pi_\theta(I|x)\| \le \sum_{i=1}^{m} \left|\frac{\partial \log \pi_\theta(I|x)}{\partial z_i}\right| \left\|\frac{\partial z_i}{\partial \theta}\right\| \le C_r \sum_{i=1}^{m} \left|\frac{\partial \log \pi_\theta(I|x)}{\partial z_i}\right|$$

by Part (c). To bound $\sum_i |\partial \log \pi_\theta/\partial z_i|$, we note that

$$\frac{\partial p_\theta(i|x)}{\partial z_j} = \frac{1}{\tau} p_\theta(i|x)(\delta_{ij} - p_\theta(j|x)), \qquad \left|\frac{\partial p_\theta(i|x)}{\partial z_j}\right| \le \frac{p_\theta(i|x)}{\tau}.$$

Write

$$\pi_\theta(I|x) = \sum_{L \in \mathrm{perm}(I)} \prod_{t=1}^{k} q_t(L), \qquad q_t(L) = \frac{p_\theta(L_t|x)}{1 - \sum_{s<t} p_\theta(L_s|x)}.$$

Then

$$\frac{\partial \log \pi_\theta}{\partial z_j} = \frac{1}{\pi_\theta} \sum_{L \in \mathrm{perm}(I)} \prod_{t=1}^{k} q_t(L) \sum_{t=1}^{k} \frac{1}{q_t(L)} \frac{\partial q_t(L)}{\partial z_j}.$$

For each factor q_t, by the quotient rule,

$$\frac{\partial q_t}{\partial z_j} = \frac{1}{1 - \sum_{s<t} p_\theta(L_s|x)} \frac{\partial p_\theta(L_t|x)}{\partial z_j} + \frac{p_\theta(L_t|x)}{\left(1 - \sum_{s<t} p_\theta(L_s|x)\right)^2} \sum_{s<t} \frac{\partial p_\theta(L_s|x)}{\partial z_j}.$$

Since L_t is still available at stage t, $1 - \sum_{s<t} p_\theta(L_s|x) \geq p_\theta(L_t|x)$. Using $|\partial p_\theta(i|x)/\partial z_j| \leq p_\theta(i|x)/\tau$, we obtain

$$\left| \frac{1}{q_t} \frac{\partial q_t}{\partial z_j} \right| \leq \frac{1}{\tau} + \frac{t-1}{\tau} \leq \frac{k}{\tau}.$$

Therefore,

$$\left| \frac{\partial \log \pi_\theta}{\partial z_j} \right| \leq \sum_{t=1}^{k} \frac{k}{\tau} = \frac{k^2}{\tau}.$$

Summing over $j = 1, \ldots, m$ gives

$$\sum_{j=1}^{m} \left| \frac{\partial \log \pi_\theta}{\partial z_j} \right| \leq \frac{mk^2}{\tau}.$$

Hence,

$$\|\nabla_\theta \log \pi_\theta(I|x)\| \leq C_\pi := \frac{mk^2 C_r}{\tau}.$$

For Part (e)–(f), we leverage the bounded domain, prediction modules, and contexts assumptions, and the Lipschitz loss $\mathcal{L}$ assumption, which gives that the per-sample loss is uniformly upper-bounded: $\ell_n(\theta, \phi, I) \leq \ell_{\max}, \forall w \in \Omega, n \in [|\mathcal{D}|], I$, where $\ell_{\max}$ depends on $\bar{L}$, C_h, and Ω. Therefore, there exists a finite constant σ_l^2 such that $\mathrm{Var}_{n,I}(\ell_n(\theta, \phi, I)) \leq \sigma_l^2$.

For Part (g), note that for any $w \in \Omega$,

$$\begin{aligned}
\|P_\theta(w)\| &= \left\| \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{I \sim \pi_\theta(\cdot|x_n)} \left[\ell_n(\theta, \phi, I) \nabla_\theta \log \pi_\theta(I|x_n) \right] \right\| \\
&\leq \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_I \left[|\ell_n(\theta, \phi, I)| \|\nabla_\theta \log \pi_\theta(I|x_n)\| \right] \\
&\leq \frac{1}{N} \sum_{n=1}^{N} \ell_{\max} C_\pi \\
&= \ell_{\max} C_\pi.
\end{aligned}$$

Therefore, we take $R_{\max} := \ell_{\max} C_\pi < \infty$. □

Lemma B.2 (Smoothness and boundedness)**.** *Both J and F are bounded below and β-smooth for some finite β.*

Proof. $J(w) = \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \sum_I \pi_\theta(I|x) \mathcal{L}(y, x, I; w)$, which contains a finite sum over $\binom{m}{k}$ subsets. Each term is smooth in w: (i) $\pi_\theta(I|x)$ is smooth in θ since $\mathrm{softmax}(z/\tau)$ is composed with the smooth selection module $z = s_\theta(x)$; (ii) $\mathcal{L}(y, x, I; w)$ is smooth in w since $\mathcal{L}$ is smooth, before which the weighted aggregations (both $h_{i,\phi}, i \in [m]$ and s_θ) are smooth. Therefore, $\nabla^2 J$ exists and is continuous on Ω, hence bounded: $\|\nabla^2 J\| \leq \beta$ for some $\beta < \infty$. The same argument applies to $F(w) = \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \mathcal{L}(y, \sum_{i \in [m]} p_\theta(i|x) h_{i,\phi}(x))$. □

Lemma B.3 (Bounded supervised gradient variance)**.** *There exists $\sigma_S^2 < \infty$ such that the minibatch variance of the joint supervised gradient estimator satisfies*

$$\mathbb{E}\left[\left\|\frac{1}{B}\sum_{b=1}^{B}\begin{pmatrix}\nabla_\theta \ell_b \\ \nabla_\phi \ell_b\end{pmatrix} - \begin{pmatrix}S_\theta(w) \\ S_\phi(w)\end{pmatrix}\right\|^2 \middle| w\right] \le \sigma_S^2.$$

Proof. The supervised gradient for a single sample is $(\nabla_\theta \ell, \nabla_\phi \ell)$. The minibatch variance is $\frac{1}{B}\mathrm{Var}_{n,I}((\nabla_\theta \ell, \nabla_\phi \ell))$, which is bounded if each per-sample gradient is bounded.

Term $\nabla_\theta \ell$. By the chain rule, $\nabla_\theta \ell = \nabla_{\hat{y}} \mathcal{L} \cdot \nabla_\theta \hat{y}$. Since $\mathcal{L}(y, \hat{y})$ is $\bar{L}$-Lipschitz, $\|\nabla_{\hat{y}} \mathcal{L}\| \le \bar{L}$. For $\nabla_\theta \hat{y}$, the output $\hat{y} = \sum_{i \in I} p_i h_i$ where $p_i = e^{z_i} / \sum_{j \in I} e^{z_j}$. For $l \in I$, the derivative with respect to z_l is the standard softmax Jacobian that

$$\frac{\partial p_i}{\partial z_l} = p_i(\delta_{il} - p_l).$$

By chain rule,

$$\nabla_\theta p_i = \sum_{l \in I} \frac{\partial p_i}{\partial z_l} \cdot \frac{\partial z_l}{\partial \theta} = \sum_{l \in I} p_i(\delta_{il} - p_l) \cdot \frac{\partial z_l}{\partial \theta},$$

we take norms and obtain

$$\|\nabla_\theta p_i\| \le \sum_{l \in I} p_i |\delta_{il} - p_l| \cdot C_r = p_i C_r \sum_{l \in I} |\delta_{il} - p_l|.$$

To compute $\sum_{l \in I} |\delta_{il} - p_l|$, we split into $l = i$ and $l \ne i$ cases, so

$$\sum_{l \in I} |\delta_{il} - p_l| = |1 - p_i| + \sum_{l \in I, l \ne i} p_l = (1 - p_i) + (1 - p_i) = 2(1 - p_i).$$

Therefore, $\|\nabla_\theta p_i\| \le 2C_r p_i(1 - p_i) \le 2C_r p_i$ and

$$\|\nabla_\theta \hat{y}\| = \|\sum_{i \in I} \nabla_\theta p_i h_i\| \le \sum_{i \in I} \|\nabla_\theta p_i\| \|h_i\| \le 2C_r C_h.$$

This yields $\|\nabla_\theta \ell\| \le 2\bar{L} C_r C_h$.

Term $\nabla_\phi \ell$. By the chain rule, $\nabla_\phi \ell = \nabla_{\hat{y}} \mathcal{L} \cdot \nabla_\phi \hat{y} = \nabla_{\hat{y}} \mathcal{L} \cdot \sum_{i \in I} p_i \nabla_\phi h_i(x)$. By Lemma B.1 Part (b), $\|\nabla_\phi h_i(x)\|$ is upper-bounded by finite $C_{h'}$, so $\|\nabla_\phi \ell\| \le \bar{L} C_{h'}$. Combining terms yields

$$\|\nabla_w \ell\| \le \|\nabla_\theta \ell\| + \|\nabla_\phi \ell\| \le \bar{L} C_{h'} + 2\bar{L} C_r C_h < \infty,$$

so the minibatch variance is bounded:

$$\frac{1}{B}\mathrm{Var}_{n,I}(\nabla_w \ell) \le \frac{1}{B}\mathbb{E}[\|\nabla_w \ell\|^2] \le \sigma_S^2 := \frac{(\bar{L} C_{h'} + 2\bar{L} C_h C_r)^2}{B}.$$

□

Lemma B.4 (Bounded policy gradient variance)**.** *Denote $\widehat{P}_B$ as the batch policy gradient estimator. Then there exists $\sigma_P^2 < \infty$ such that the minibatch variance of the policy gradient estimator satisfies*

$$\mathbb{E}\|\widehat{P}_B - P_\theta\|^2 \le \sigma_P^2.$$

Proof. Denote

$$Y_b := \ell_{n_b}(\theta, \phi, I_b), \qquad s_b := \nabla_\theta \log \pi_\theta(I_b | x_{n_b}), \qquad b_{-b} := \frac{1}{B-1}\sum_{a \ne b} Y_a$$

and then our policy gradient estimator is $\widehat{P}_B = \frac{1}{B}\sum_{b=1}^{B}(Y_b - b_{-b})s_b$. Let $\mu := \mathbb{E}_{n,I}[Y]$ and $X_b := Y_b - \mu$. Since $\mathbb{E}[s_b] = 0$, subtracting the constant μ does not change the estimator, i.e.,

$$(Y_b - b_{-b})s_b = \left(X_b - \frac{1}{B-1}\sum_{a\neq b} X_a\right) s_b.$$

Therefore,

$$\widehat{P}_B = U - V, \quad \text{where} \quad U := \frac{1}{B}\sum_{b=1}^{B} X_b s_b, \quad V := \frac{1}{B(B-1)}\sum_{b=1}^{B}\sum_{a\neq b} X_a s_b.$$

In addition, $P_\theta(w) = \mathbb{E}[Xs]$ since $\mathbb{E}[\mu s] = 0$. Hence, $\widehat{P}_B - P_\theta = (U - P_\theta) - V$ and

$$\mathbb{E}\|\widehat{P}_B - P_\theta\|^2 \leq 2\mathbb{E}\|U - P_\theta\|^2 + 2\mathbb{E}\|V\|^2.$$

For the first component,

$$\mathbb{E}\|U - P_\theta\|^2 = \frac{1}{B}\operatorname{Var}(Xs) \leq \frac{1}{B}\mathbb{E}[X^2\|s\|^2] \leq \frac{C_\pi^2}{B}\mathbb{E}[X^2].$$

By Lemma B.1 Part (f), $\mathbb{E}[X^2] = \operatorname{Var}_{n,I}(\ell_n(\theta,\phi,I)) \leq \sigma_l^2$, so we have $\mathbb{E}\|U - P_\theta\|^2 \leq \frac{C_\pi^2}{B}\sigma_l^2$.

For the second component, we expand $\mathbb{E}\|V\|^2$ and the only nonzero cross terms are identical-pair and swapped-pair terms. Using $\mathbb{E}[X] = 0$, $\mathbb{E}[s] = 0$, independence across minibatch samples, and $\|s\| \leq C_\pi$, we obtain

$$\mathbb{E}\|V\|^2 \leq \frac{2C_\pi^2}{B(B-1)}\operatorname{Var}_{n,I}(\ell_n(\theta,\phi,I)) \leq \frac{2C_\pi^2}{B(B-1)}\sigma_l^2.$$

Combining the two components, we have

$$\mathbb{E}\|\widehat{P}_B - P_\theta\|^2 \leq \frac{2C_\pi^2}{B}\sigma_l^2 + \frac{4C_\pi^2}{B(B-1)}\sigma_l^2 \leq \sigma_P^2 := \frac{4C_\pi^2}{B}\left(1 + \frac{1}{B-1}\right)\sigma_l^2.$$

□

Lemma B.5 (Master descent lemma)**.** *Let ϕ be a β-smooth function and bounded below by $\phi^\star$. Consider $w_{t+1} = w_t - \eta g_t$ with $g_t = \nabla\phi(w_t) + b_t + \xi_t$ where $\mathbb{E}[\xi_t|w_t] = 0$, $\mathbb{E}[\|\xi_t\|^2|w_t] \leq \sigma^2$, $\eta \leq \frac{1}{2\beta}$. Then*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla\phi(w_t)\|^2 \leq \frac{2(\phi(w_0) - \phi^\star)}{\eta T} + \beta\eta\sigma^2 + \frac{3}{T}\sum_{t=0}^{T-1}\|b_t\|^2.$$

Proof. By β-smoothness,

$$\phi(w_{t+1}) \leq \phi(w_t) + \langle\nabla\phi(w_t), w_{t+1} - w_t\rangle + \frac{\beta}{2}\|w_{t+1} - w_t\|^2.$$

Substituting $w_{t+1} - w_t = -\eta(\nabla\phi(w_t) + b_t + \xi_t)$, taking conditional expectation, and using $\mathbb{E}[\xi_t|w_t] = 0$ gives

$$\begin{aligned}\mathbb{E}[\phi(w_{t+1})|w_t] &\leq \phi(w_t) - \eta\|\nabla\phi\|^2 - \eta\langle\nabla\phi, b_t\rangle + \frac{\beta\eta^2}{2}\left(\|\nabla\phi + b_t\|^2 + \sigma^2\right) \\ &\leq \phi(w_t) - \eta\left(1 - \frac{\beta\eta}{2}\right)\|\nabla\phi\|^2 - \eta(1 - \beta\eta)\langle\nabla\phi, b_t\rangle + \frac{\beta\eta^2}{2}(\|b_t\|^2 + \sigma^2)\end{aligned}$$

by expanding the square. Since $\eta \leq 1/(2\beta)$, we have $1 - \frac{\beta\eta}{2} \geq \frac{3}{4}$ and $|1 - \beta\eta| \leq 1$. Then using Young's inequality $|\langle\nabla\phi, b_t\rangle| \leq \frac{1}{4}\|\nabla\phi\|^2 + \|b_t\|^2$ yields

$$\mathbb{E}[\phi(w_{t+1})|w_t] \leq \phi(w_t) - \frac{\eta}{2}\|\nabla\phi(w_t)\|^2 + \frac{3\eta}{2}\|b_t\|^2 + \frac{\beta\eta^2}{2}\sigma^2.$$

Rearranging, summing over $t = 0, \ldots, T-1$, telescoping, and dividing by $\eta T/2$ gives the claim. □

B.3 Convergence to Our Objective

Let Assumptions 1-4 hold. Then J is β-smooth for some finite β and bounded below by $J^\star$. For any fixed λ, running Algorithm 1 with $\eta = \frac{1}{\beta\sqrt{T}}$ and $T \geq 4$ yields

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla J(w_t)\|^2 \leq \frac{2\beta\Delta_0 + 2\sigma_S^2 + 2\lambda^2\sigma_P^2}{\sqrt{T}} + 3(1-\lambda)^2 R_{\max}^2,$$

where $\Delta_0 := J(w_0) - J^\star$. For any finite T, the upper bound is minimized at

$$\lambda^\star(T) = \frac{3R_{\max}^2}{2\sigma_P^2/\sqrt{T} + 3R_{\max}^2},$$

which gives

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla J(w_t)\|^2 \leq \frac{2\beta\Delta_0 + 2\sigma_S^2}{\sqrt{T}} + \frac{6\sigma_P^2 R_{\max}^2}{2\sigma_P^2 + 3R_{\max}^2\sqrt{T}}.$$

Proof. Recall that the joint estimator satisfies $\mathbb{E}[g_t^\lambda | w_t] = \nabla_w J(w_t) + b_t^\lambda$ where $b_t^\lambda = \big(-(1-\lambda)P_\theta(w_t),\, 0\big)$. With $R_{\max} := \sup_{w\in w}\|P_\theta(w)\|$, we have $\|b_t^\lambda\| \leq (1-\lambda)R_{\max}$.

Let $\xi_t := g_t^\lambda - \mathbb{E}[g_t^\lambda | w_t]$. Since the estimator decomposes into the joint supervised gradient minibatch noise and the scaled policy gradient noise, by $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$, we have $\mathbb{E}[\|\xi_t\|^2 | w_t] \leq 2\sigma_S^2 + 2\lambda^2\sigma_P^2$.

Applying the master descent lemma with $\phi = J$, $\sigma^2 = 2\sigma_S^2 + 2\lambda^2\sigma_P^2$ and $\|b_t^\lambda\|^2 \leq (1-\lambda)^2 R_{\max}^2$ yields

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla J(w_t)\|^2 \leq \frac{2\beta\Delta_0 + 2\sigma_S^2 + 2\lambda^2\sigma_P^2}{\sqrt{T}} + 3(1-\lambda)^2 R_{\max}^2.$$

The λ-dependent part is

$$g(\lambda) = \frac{2\lambda^2\sigma_P^2}{\sqrt{T}} + 3(1-\lambda)^2 R_{\max}^2.$$

We set $g'(\lambda) = 0$ and obtain

$$\lambda^\star(T) = \frac{3R_{\max}^2}{2\sigma_P^2/\sqrt{T} + 3R_{\max}^2}.$$

Since $g''(\lambda) = \frac{4\sigma_P^2}{\sqrt{T}} + 6R_{\max}^2 > 0$, $\lambda^\star$ is the unique minimizer. Applying $\lambda^\star$ to $g(\lambda)$ gives

$$g(\lambda^\star) = \frac{6\sigma_P^2 R_{\max}^2}{2\sigma_P^2 + 3R_{\max}^2\sqrt{T}}.$$

Adding the λ-independent term $\frac{2\beta\Delta_0 + 2\sigma_S^2}{\sqrt{T}}$ gives the claim. □

B.4 Convergence to the True Objective

Lemma B.6 (Jacobian mismatch)**.** *Denote $p = \mathit{softmax}(z/\tau)$. For any subset I with $|I| = k$ and $\rho_I := \sum_{i\notin I} p_i \leq \frac{1}{2}$, we have*

$$\|\nabla_\theta \hat{f}(I, x; \theta) - \nabla_\theta f(x;\theta)\| \leq C_B \rho_I, \quad \textit{where}$$

$$C_B := 12 C_r C_h, \qquad f = \sum_i p_i h_i, \qquad \hat{f} = \sum_{i\in I}\frac{p_i}{1-\rho_I} h_i.$$

Proof. Note that $\nabla_\theta p_i = \sum_{l=1}^m p_i(\delta_{il} - p_l)\nabla_\theta z_l$ and thus $\|\nabla_\theta p_i\| \leq p_i \sum_l |\delta_{il} - p_l| C_r$. Since

$$\sum_l |\delta_{il} - p_l| = (1-p_i) + \sum_{l\neq i} p_l = (1-p_i) + (1-p_i) = 2(1-p_i) \leq 2,$$

we obtain

$$\|\nabla_\theta p_i\| \le 2C_r p_i \quad \text{for all } i. \tag{13}$$

Note that

$$\nabla_\theta \hat{f} = \frac{1}{1-\rho_I}\sum_{i\in I}\nabla_\theta p_i \cdot h_i + \frac{\nabla_\theta \rho_I}{(1-\rho_I)^2}\sum_{i\in I} p_i h_i, \quad \text{and}$$
$$\nabla_\theta f = \sum_{i\in I}\nabla_\theta p_i \cdot h_i + \sum_{i\notin I}\nabla_\theta p_i \cdot h_i.$$

Subtracting them yields

$$\nabla_\theta \hat{f} - \nabla_\theta f = \underbrace{\frac{\rho_I}{1-\rho_I}\sum_{i\in I}\nabla_\theta p_i \cdot h_i}_{\text{(i)}} + \underbrace{\frac{\nabla_\theta \rho_I}{(1-\rho_I)^2}\sum_{i\in I} p_i h_i}_{\text{(ii)}} - \underbrace{\sum_{i\notin I}\nabla_\theta p_i \cdot h_i}_{\text{(iii)}}.$$

For term (i), using Eq. (13) gives

$$\sum_{i\in I}\|\nabla_\theta p_i\| C_h \le 2C_r C_h \sum_{i\in I} p_i = 2C_r C_h(1-\rho_I),$$

so

$$\|\text{(i)}\| \le \frac{\rho_I}{1-\rho_I}\cdot 2C_r C_h(1-\rho_I) = 2C_r C_h \rho_I.$$

For term (ii), by Eq. (13),

$$\|\nabla_\theta \rho_I\| = \left\|\sum_{i\notin I}\nabla_\theta p_i\right\| \le 2C_r \sum_{i\notin I} p_i = 2C_r\rho_I.$$

Also,

$$\left\|\sum_{i\in I} p_i h_i\right\| \le (1-\rho_I)C_h \le C_h.$$

Therefore,

$$\|\text{(ii)}\| \le \frac{2C_r\rho_I}{(1-\rho_I)^2}\cdot C_h = \frac{2C_r C_h \rho_I}{(1-\rho_I)^2}.$$

For $\rho_I \le 1/2$, we have $(1-\rho_I)^2 \ge 1/4$, hence $\|\text{(ii)}\| \le 8C_r C_h \rho_I$.

For term (iii),

$$\|\text{(iii)}\| \le \sum_{i\notin I} 2C_r p_i \cdot C_h = 2C_r C_h \rho_I.$$

Combining the three bounds,

$$\|\nabla_\theta \hat{f} - \nabla_\theta f\| \le (2+8+2)C_r C_h \rho_I = 12 C_r C_h \rho_I = C_B \rho_I.$$

□

Lemma B.7 (θ-gradient truncation bias)**.**

$$\|b_t^{\text{trunc}}\| = \|\nabla_\theta J(\theta) - \nabla_\theta F(\theta)\| \le c_\theta (m-k)\left(e^{-\gamma_{\min}} + e^{-\gamma_{\min}/\tau}\right) =: \epsilon_\theta,$$

where c_θ depends on $\bar{L}, \bar{\beta}, C_h, C_r, m, k, \tau$.

Proof. Denote $\Delta L_I := \mathcal{L}(y,\hat{f}) - \mathcal{L}(y,f)$. By the score-function identity,

$$b_t^{\text{trunc}} = \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}} \mathbb{E}_{I\sim\pi_\theta}\left[\Delta L_I \cdot \nabla_\theta \log \pi_\theta(I|x) + \nabla_\theta \Delta L_I\right].$$

Term of $\mathbb{E}_I[\Delta L_I \cdot \nabla_\theta \log \pi_\theta]$. By Lipschitzness of $\mathcal{L}$ and Lemma B.6, $|\Delta L_I| \le \bar{L}\|\hat{f} - f\| \le 2\bar{L}C_h\rho_I$. Since $\|\nabla_\theta \log \pi_\theta\| \le C_\pi$, we have

$$\|\mathbb{E}_I[\Delta L_I \cdot \nabla_\theta \log \pi_\theta]\| \le 2\bar{L}C_hC_\pi \cdot \mathbb{E}_I[\rho_I]. \tag{14}$$

Term of $\mathbb{E}_I[\nabla_\theta \Delta L_I]$. By adding and subtracting,

$$\nabla_\theta \Delta L_I = \nabla_\theta \hat{f} \cdot \nabla_{\hat{y}} \mathcal{L}(\hat{f}) - \nabla_\theta f \cdot \nabla_{\hat{y}} \mathcal{L}(f) = \underbrace{[\nabla_{\hat{y}} \mathcal{L}(\hat{f}) - \nabla_{\hat{y}} \mathcal{L}(f)]}_{A} \cdot \nabla_\theta \hat{f} + \nabla_{\hat{y}} \mathcal{L}(f) \cdot \underbrace{[\nabla_\theta \hat{f} - \nabla_\theta f]}_{B}.$$

For term A, by the smoothness of $\mathcal{L}$, $\|A\| \le \bar{\beta}\|\hat{f} - f\| \le 2\bar{\beta}C_h\rho_I$. Also, by the same bounds as in Lemma B.6, $\|\nabla_\theta \hat{f}\| \le 12C_rC_h =: C'_B$. Therefore, $\|A \cdot \nabla_\theta \hat{f}\| \le 2\bar{\beta}C_hC'_B\rho_I$ and hence

$$\|\mathbb{E}_I[A \cdot \nabla_\theta \hat{f}]\| \le 2\bar{\beta}C_hC'_B\mathbb{E}_I[\rho_I]. \tag{15}$$

For term B, since $\|\nabla_{\hat{y}} \mathcal{L}(f)\| \le \bar{L}$, we split the expectation

$$\mathbb{E}_I[\|B\|] = \mathbb{E}_I[\|B\|\mathbf{1}\{I = S\}] + \mathbb{E}_I[\|B\|\mathbf{1}\{I \ne S\}].$$

By Assumption 1, we denote $\gamma_{\min}$ as the minimal logit difference between the selected and unselected item along the optimization trajectory that

$$\operatorname{essinf}_x \big(s_{(k)}(x;\theta) - s_{(k+1)}(x;\theta)\big) \ge \gamma_{\min} > 0.$$

Therefore, when $I = S$, for sufficiently large $\gamma_{\min}$, we have

$$\rho_S \le (m-k)e^{-\gamma_{\min}} < \frac{1}{2},$$

so Lemma B.6 applies and yields $\|B\| = \|\nabla_\theta \hat{f} - \nabla_\theta f\| \le C_B\rho_S$. Hence,

$$\mathbb{E}_I[\|B\|\mathbf{1}\{I = S\}] \le C_B\rho_S\mathbb{P}(I = S).$$

When $I \ne S$, we use $\|B\| \le C_{\bar{B}}$ where $C_{\bar{B}} := 14\bar{L}C_rC_h$ due to $\|\nabla_\theta f\| \le 2C_rC_h$ by Lemma B.3 and $\|\nabla_\theta \hat{f}\| \le C'_B = 12C_rC_h$ by Lemma B.6, which yields $\|\nabla_\theta \hat{f} - \nabla_\theta f\| \le \|\nabla_\theta \hat{f}\| + \|\nabla_\theta f\| \le 14C_rC_h$. Therefore,

$$\mathbb{E}_I[\|B\|\mathbf{1}\{I \ne S\}] \le C_{\bar{B}}\mathbb{P}(I \ne S).$$

Combining the two cases, $\mathbb{E}_I[\|B\|] \le C_B\rho_S\mathbb{P}(I = S) + C_{\bar{B}}\mathbb{P}(I \ne S)$. Thus

$$\mathbb{E}_I\left[\|\nabla_{\hat{y}} \mathcal{L}(f) \cdot B\|\right] \le \bar{L}C_B\rho_S\mathbb{P}(I = S) + L_{\mathcal{L}}C_{\bar{B}}\mathbb{P}(I \ne S).$$

Combining this with Eq. (15) yields

$$\|\mathbb{E}_I[\nabla_\theta \Delta L_I]\| \le 2\beta_{\mathcal{L}}C_hC'_B\mathbb{E}_I[\rho_I] + L_{\mathcal{L}}C_B\rho_S\mathbb{P}(I = S) + L_{\mathcal{L}}C_{\bar{B}}\mathbb{P}(I \ne S).$$

Using $\rho_S \le (m-k)e^{-\gamma_{\min}}$ and $\mathbb{P}(I \ne S) \le (m-k)e^{-\gamma_{\min}/\tau}$, we obtain

$$\|\mathbb{E}_I[\nabla_\theta \Delta L_I]\| \le c(m-k)\left(e^{-\gamma_{\min}} + e^{-\gamma_{\min}/\tau}\right) \tag{16}$$

for a constant $c > 0$ depending on $L_{\mathcal{L}}, \beta_{\mathcal{L}}, C_h, C_r, m, k, \tau$.

Combining Term 1 and Term 2. From (14) and (16), we obtain

$$\|b_t^{\text{trunc}}\| \le 2\bar{L}C_hC_\pi\mathbb{E}_x\mathbb{E}_I[\rho_I] + c(m-k)\left(e^{-\gamma_{\min}} + e^{-\gamma_{\min}/\tau}\right).$$

Using

$$\mathbb{E}_I[\rho_I] \le (m-k)e^{-\gamma_{\min}} + (m-k)e^{-\gamma_{\min}/\tau},$$

we conclude

$$\|b_t^{\text{trunc}}\| \le c_\theta(m-k)\left(e^{-\gamma_{\min}} + e^{-\gamma_{\min}/\tau}\right), \quad c_\theta := 2\bar{L}C_hC_\pi + c.$$

□

Lemma B.8 (ϕ-gradient truncation bias)**.**

$$\|\nabla_\phi J(\theta,\phi) - \nabla_\phi F(\theta,\phi)\| \le c_\phi (m-k)\left(e^{-\gamma_{\min}} + e^{-\gamma_{\min}/\tau}\right),$$

where c_ϕ depends on $\bar{L}, \bar{\beta}, C_h, C_{h'}, m, k, \tau$.

Proof. Since $\pi_\theta(I|x)$ does not depend on ϕ,

$$\nabla_\phi J(\theta,\phi) - \nabla_\phi F(\theta,\phi) = \mathbb{E}_x \mathbb{E}_{I\sim\pi_\theta}[\nabla_\phi \mathcal{L}(y,\hat{f}) - \nabla_\phi \mathcal{L}(y,f)].$$

By the chain rule, $\nabla_\phi \mathcal{L}(y,\hat{f}) - \nabla_\phi \mathcal{L}(y,f) = A_I + B_I$, where

$$A_I := \left[\nabla_{\hat{y}} \mathcal{L}(y,\hat{f}) - \nabla_{\hat{y}} \mathcal{L}(y,f)\right] \nabla_\phi \hat{f}, \quad B_I := \nabla_{\hat{y}} \mathcal{L}(y,f)\left[\nabla_\phi \hat{f} - \nabla_\phi f\right].$$

By the smoothness of $\mathcal{L}$, $\|\nabla_{\hat{y}} \mathcal{L}(y,\hat{f}) - \nabla_{\hat{y}} \mathcal{L}(y,f)\| \le \bar{\beta}\|\hat{f} - f\|$. Also, $\|\hat{f} - f\| \le 2C_h \rho_I$. and $\nabla_\phi \hat{f} = \sum_{i\in I} \frac{p_i}{1-\rho_I} \nabla_\phi h_{i,\phi}(x)$, so $\|\nabla_\phi \hat{f}\| \le C_{h'}$. Therefore, $\|A_I\| \le 2\bar{\beta} C_h C_{h'} \rho_I$. Since

$$\nabla_\phi f = \sum_{i=1}^{m} p_i \nabla_\phi h_{i,\phi}(x), \quad \nabla_\phi \hat{f} = \sum_{i\in I} \frac{p_i}{1-\rho_I} \nabla_\phi h_{i,\phi}(x),$$

we have

$$\nabla_\phi \hat{f} - \nabla_\phi f = \frac{\rho_I}{1-\rho_I} \sum_{i\in I} p_i \nabla_\phi h_{i,\phi}(x) - \sum_{i\notin I} p_i \nabla_\phi h_{i,\phi}(x).$$

Using $\|\nabla_\phi h_{i,\phi}(x)\| \le C_{h'}$, we get

$$\left\| \frac{\rho_I}{1-\rho_I} \sum_{i\in I} p_i \nabla_\phi h_{i,\phi}(x) \right\| \le \rho_I C_{h'}, \quad \left\| \sum_{i\notin I} p_i \nabla_\phi h_{i,\phi}(x) \right\| \le \rho_I C_{h'}.$$

Therefore, $\|\nabla_\phi \hat{f} - \nabla_\phi f\| \le 2C_{h'}\rho_I$. Since $\|\nabla_{\hat{y}} \mathcal{L}(y,f)\| \le \bar{L}$, we have $\|B_I\| \le 2\bar{L} C_{h'} \rho_I$. Combining the two bounds,

$$\|\nabla_\phi \mathcal{L}(y,\hat{f}) - \nabla_\phi \mathcal{L}(y,f)\| \le 2C_{h'}(\bar{\beta} C_h + \bar{L})\rho_I.$$

Taking expectation over I and x,

$$\|\nabla_\phi J(\theta,\phi) - \nabla_\phi F(\theta,\phi)\| \le 2C_{h'}(\bar{\beta} C_h + \bar{L}) \mathbb{E}_x \mathbb{E}_I[\rho_I].$$

Using $\mathbb{E}_I[\rho_I] \le (m-k)e^{-\gamma_{\min}} + (m-k)e^{-\gamma_{\min}/\tau}$, we obtain

$$\|\nabla_\phi J(\theta,\phi) - \nabla_\phi F(\theta,\phi)\| \le c_\phi (m-k)\left(e^{-\gamma_{\min}} + e^{-\gamma_{\min}/\tau}\right), \quad c_\phi := 2C_{h'}(\bar{\beta} C_h + \bar{L}).$$

□

Lemma B.9 (Truncation bias)**.** *We combine the θ-gradient and ϕ-gradient truncation bias and have*

$$\|\nabla_w J - \nabla_w F\|^2 = \|\nabla_\theta J - \nabla_\theta F\|^2 + \|\nabla_\phi J - \nabla_\phi F\|^2$$

and thus

$$\epsilon := \sqrt{\epsilon_\theta^2 + \epsilon_\phi^2} \le \epsilon_\theta + \epsilon_\phi$$

With the above lemmas derived, we prove Theorem 4.1 as follows.

Proof. Recall that relative to the true objective F, $g_t^\lambda = \nabla_w F(w_t) + b_t^\lambda + \xi_t$, where

$$b_t^\lambda = \underbrace{\nabla_w J(w_t) - \nabla_w F(w_t)}_{\text{truncation bias}} + \underbrace{\left(-(1-\lambda)P_\theta(w_t), 0\right)}_{\text{policy gradient suppression bias}}.$$

Define $\epsilon := \sup_t \|\nabla_w J(w_t) - \nabla_w F(w_t)\|$ and $R_{\max} := \sup_t \|P_\theta(w_t)\|$, then $\|b_t^\lambda\| \le \epsilon + (1-\lambda)R_{\max}$. With ϵ derived in Lemma B.9 and $R_{\max}$ derived in Lemma B.1, we apply the master descent

lemma with with $\phi = F$, $\sigma^2 = 2\sigma_S^2 + 2\lambda^2\sigma_P^2$, and $\|b_t^\lambda\|^2 \le (\epsilon + (1-\lambda)R_{\max})^2$. With step size $\eta = \frac{1}{\beta\sqrt{T}}$ and $\eta \le \frac{1}{2\beta}$ (i.e. $T \ge 4$), we have

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla F(w_t)\|^2 \le \frac{2\beta(F(w_0)-F^\star) + 2\sigma_S^2 + 2\lambda^2\sigma_P^2}{\sqrt{T}} + 3\left(\epsilon + (1-\lambda)R_{\max}\right)^2.$$

For any fixed T, the bound is minimized at

$$\begin{aligned}\lambda^\star(T) &= \arg\min_{\lambda\in[0,1]}\left[\frac{2\lambda^2\sigma_P^2}{\sqrt{T}} + 3(\epsilon + (1-\lambda)R_{\max})^2\right] \\ &= \min\left\{1,\ \frac{3R_{\max}(\epsilon + R_{\max})}{2\sigma_P^2/\sqrt{T} + 3R_{\max}^2}\right\}.\end{aligned}$$

□

The gradient of our objective provides a smooth bias-variance tradeoff controlled by λ: Increasing λ reduces the bias $b_t^\lambda = \nabla_w J(w_t) - \nabla_w F(w_t) - \big((1-\lambda)P_\theta(w_t), 0\big)$ and therefore lowers the asymptotic bias floor in the stationarity bound; At the same time, increasing λ also increases the variance of the estimator through the policy gradient term, which enters the bound as $2\lambda^2\sigma_P^2/\sqrt{T}$.

C Experiments

In the following, we detail the data, baselines, hyperparameters, and ablation results for each application. In Figure 13, we visualize the convergence of all methods in different applications. Our experiments are conducted on a cluster containing A40, L40S, and A100 GPUs.

C.1 Regression Task

Data generation. Recall that $\{v_1, \dots, v_m\}$ are m candidate vectors of dimension d, where we take $d = 128, k = 3$, and $m \in \{64, 128, 256, 512\}$. We sample each coordinate of $x \in \mathbb{R}^d$ following $\mathcal{N}(0,1)$, each coordinate of v following $\mathcal{N}(0, 1/d)$, and each element of $\theta^\star$ following $\mathcal{N}(0, 1/d)$. Denote S as the set that contains the top-3 elements. Then we have

$$y = \sum_{i\in S} p_i v_i, \quad v_i \sim \mathcal{N}(0, I_h/d).$$

Conditioned on the top-3 weights p, we use $\mathbb{E}[\|v_i\|^2] = 1$ and obtain

$$\mathbb{E}[\|y\|^2|p] = \sum_{i\in S} p_i^2 \mathbb{E}[\|v_i\|^2] = \sum_{i\in S} p_i^2.$$

Therefore,

$$\mathbb{E}[\|y\|^2] = \mathbb{E}\left[\sum_{i\in S} p_i^2\right] = \frac{1}{3} + \frac{1}{9\log m} + o\left(\frac{1}{\log m}\right)$$

where the second equality is due to Lemma C.1. That is, we show that as m increases, the expected squared norm of y only slightly decreases. Therefore, our observation that MSE loss increases as m increases indicates the increasing optimization difficulty.

Lemma C.1 (Top-3 softmax weight concentration)**.** *Let $z_1, \dots, z_m$ be i.i.d. sampled from $\mathcal{N}(0,1)$ and denote the top-3 as $z_{(1)} \ge z_{(2)} \ge z_{(3)}$. Define the top-3 softmax weights*

$$p_i = \frac{e^{z_{(i)}}}{\sum_{j=1}^{3} e^{z_{(j)}}}, \quad i = 1, 2, 3.$$

Then as $m \to \infty$,

$$\mathbb{E}\left[\sum_{i=1}^{3} p_i^2\right] = \frac{1}{3} + \frac{1}{9\log m} + o\left(\frac{1}{\log m}\right).$$

Proof. Define gaps $\Delta_2 = z_{(1)} - z_{(2)}$ and $\Delta_3 = z_{(1)} - z_{(3)}$. By shift invariance of softmax,

$$p_1 = \frac{1}{1 + e^{-\Delta_2} + e^{-\Delta_3}}, \quad p_2 = \frac{e^{-\Delta_2}}{Z}, \quad p_3 = \frac{e^{-\Delta_3}}{Z},$$

where $Z = 1 + e^{-\Delta_2} + e^{-\Delta_3}$. Denote Φ as the standard normal CDF, and let

$$b_m = \Phi^{-1}\left(1 - \frac{1}{m}\right) \sim \sqrt{2 \log m},$$

then the classical extreme value theory (Leadbetter et al., 2012; De Haan and Ferreira, 2006) gives $z_{(1)} = b_m + o_p(1)$, and the spacings satisfy

$$\Delta_2 = \frac{H_2}{b_m} + o_p\left(\frac{1}{b_m}\right), \quad \Delta_3 = \frac{H_3}{b_m} + o_p\left(\frac{1}{b_m}\right),$$

where $(H_2, H_3) = O_p(1)$. Therefore, $\Delta_i = O_p(1/b_m)$. Let $\delta_1 = 0$, $\delta_2 = \Delta_2$, $\delta_3 = \Delta_3$. We have

$$p_i = \frac{e^{-\delta_i}}{\sum_{j=1}^{3} e^{-\delta_j}}.$$

For small δ_i, expand $e^{-\delta_i} = 1 - \delta_i + \frac{\delta_i^2}{2} + O(\delta_i^3)$. Let $\bar{\delta} = \frac{1}{3}\sum_{i=1}^{3} \delta_i$, then a standard expansion gives

$$p_i = \frac{1}{3} - \frac{1}{3}(\delta_i - \bar{\delta}) + O(\|\delta\|^2).$$

Using the expansion,

$$p_i^2 = \frac{1}{9} - \frac{2}{9}(\delta_i - \bar{\delta}) + \frac{1}{9}(\delta_i - \bar{\delta})^2 + O(\|\delta\|^3).$$

Summing over i, we have

$$\sum_{i=1}^{3} p_i^2 = \frac{1}{3} + \frac{1}{9}\sum_{i=1}^{3}(\delta_i - \bar{\delta})^2 + O(\|\delta\|^3).$$

By $\delta_i = O_p(1/b_m)$, we have

$$\sum_{i=1}^{3}(\delta_i - \bar{\delta})^2 = \frac{1}{b_m^2}\sum_{i=1}^{3}(H_i - \bar{H})^2 + o_p\left(\frac{1}{b_m^2}\right).$$

Therefore, taking expectation yields

$$\mathbb{E}\left[\sum_{i=1}^{3} p_i^2\right] = \frac{1}{3} + \frac{1}{9b_m^2}\mathbb{E}\left[\sum_{i=1}^{3}(H_i - \bar{H})^2\right] + o\left(\frac{1}{b_m^2}\right).$$

For Gaussian extremes, we have

$$\mathbb{E}\left[\sum_{i=1}^{3}(H_i - \bar{H})^2\right] = 2.$$

Therefore,

$$\mathbb{E}\left[\sum_{i=1}^{3} p_i^2\right] = \frac{1}{3} + \frac{2}{9b_m^2} + o\left(\frac{1}{b_m^2}\right) = \frac{1}{3} + \frac{1}{9\log m} + o\left(\frac{1}{\log m}\right)$$

due to $b_m^2 \sim 2\log m$. □

Hyperparameters. For all the methods, we tune the learning rate in the grid {1e-5, 5e-5, 1e-4, 5e-4, 1e-3} and use full batch training. For all the methods that involve sampling from the multinomial distribution defined by the softmax of logits, we tune the temperature $\tau \in \{0.02, 0.05, 0.1, 0.5, 1.0, 2.0\}$. For our method, we tune $\lambda \in$ {0.01, 0.05, 0.1, 0.5}. For the DefaultMoE baseline, we tune $\beta \in$ {0.6, 0.7, 0.8, 0.9, 0.99}.

Table 1: Performance on the regression task.

Method	MSE Loss ($\times$1e-3, $\downarrow$)				Top-1 Selection Accuracy (%, $\uparrow$)			
	$m = 64$	$m = 128$	$m = 256$	$m = 512$	$m = 64$	$m = 128$	$m = 256$	$m = 512$
Deterministic	1.01	2.02	5.73	5.99	89.0	66.7	4.0	0.6
Gaussian	1.39	2.15	5.14	6.22	80.8	64.5	14.3	1.0
DefaultMoE	1.20	1.44	1.87	3.29	80.1	68.6	53.5	41.4
SparseMixer	1.39	1.92	3.11	4.30	86.6	73.9	46.8	27.0
Supervised-only	0.99	1.43	1.91	4.67	89.9	80.0	61.2	11.9
Ours	**0.96**	**1.17**	**1.51**	**2.86**	**91.0**	**84.5**	**66.8**	**47.6**

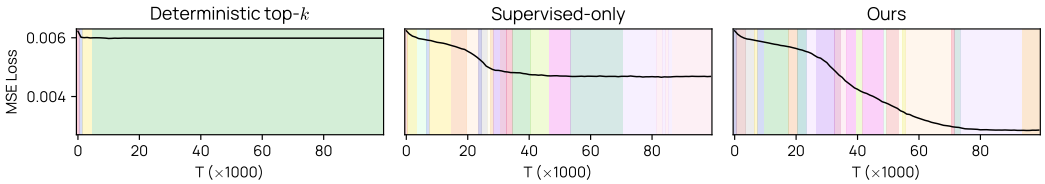


Figure 8: Decrease of test loss and the evolution of selection decisions (on two test samples) during optimization from θ_0 to θ_T under different methods. The line chart represents loss decrease, and each background color represents one selection decision on two test samples. We observe that the selection module explores more selection decisions with our method.

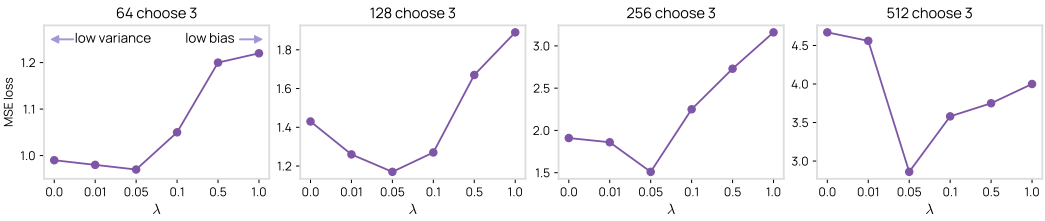


Figure 9: Bias/variance tradeoff adjusted by λ, showing that the optimal tradeoff is obtained at $\lambda \in (0, 1)$ under different m's.

C.2 Retrieval-Augmented Generative Models

Prediction module. Note that $p(y|x)$ can be decomposed in terms of latent variables, such as an item i or a subset I: $p(y|x) = \sum_i p(y|x, i) \cdot p(i|x) = \sum_I p(y|x, I) \cdot p(I|x)$. We denote $w_i(x) = e^{s(i|x)}$ and $p(i|x) = \frac{w_i(x)}{\sum_{j=1}^m w_j(x)}$. Since we generate an ordered sequence $L = (L_1, \ldots, L_k)$ without replacement and then discard the order and obtain $I = \{L_1, \ldots, L_k\}$, with

$$p(L|x) = \prod_{t=1}^{k} \frac{w_{L_t}(x)}{\sum_{l \notin \{L_1, \ldots, L_{t-1}\}} w_l(x)}, \quad p(I|x) = \sum_{L:\{L_1,\ldots,L_k\}=I} p(L|x).$$

In the following, we show that taking

$$p(y|x, I) = \sum_{i \in I} a_i(x, I) \cdot p(y|x, i), \quad a_i(x, I) = \frac{\sum_{L:\{L_1,\ldots,L_k\}=I, L_1=i} p(L|x)}{p(I|x)}$$

makes our objective an upper bound of the true objective. First, the weights $a_i(x, I)$ form a probability distribution within every set since $\sum_{i \in I} a_i(x, I) = 1$. Additionally,

$$\begin{aligned}\sum_{I: i \in I} p(I|x) a_i(x, I) &= \sum_{I: i \in I} \sum_{\substack{L: \{L_1, \ldots, L_k\} = I \\ L_1 = i}} p(L|x) \\ &= \sum_{I: i \in I} \Pr(I, L_1 = i|x) \\ &= \Pr(L_1 = i|x) \\ &= p(i|x),\end{aligned}$$

because the first draw follows our original softmax distribution. Therefore,

$$\begin{aligned}\sum_I p(I|x) p(y|x, I) &= \sum_I p(I|x) \sum_{i \in I} a_i(x, I) p(y|x, i) \\ &= \sum_I \sum_{i \in I} p(I|x) a_i(x, I) p(y|x, i) \\ &= \sum_i p(i|x) p(y|x, i) \\ &= p(y|x).\end{aligned}$$

That is, $a_i(x, I), i \in I$ are the correct aggregation weights for $p(y|x, i)$ so that $p(y|x) = \sum_i p(y|x, i) \cdot p(i|x) = \sum_I p(y|x, I) \cdot p(I|x)$ is valid.

Next, with the model parameters θ and ϕ, denote $\pi_\theta(I|x)$ as the probability of sampling the subset I given x, and denote $f_{\theta,\phi}(I, x)$ as the probability of outputting y given x and selecting subset I. By Jensen's inequality, since $\mathcal{L} = -\log(\cdot)$ which is convex, then for each sample x, we have

$$\begin{aligned}\mathcal{L}\left(y, \sum_{i=1}^m p_\theta(i|x) \cdot p(y|i, x)\right) &= \mathcal{L}\left(y, \sum_I f_{\theta,\phi}(I, x) \cdot \pi_\theta(I|x)\right) \\ &\overset{\text{Jensen}}{\leq} \sum_I \mathcal{L}(y, f_{\theta,\phi}(I, x)) \cdot \pi_\theta(I|x) \\ &= \mathbb{E}_{I \sim \pi_\theta}[\mathcal{L}(y, x, I; \theta, \phi)].\end{aligned}$$

The left-most expression is the true objective, and the right-most is our proposed objective. Therefore, our objective upper-bounds the true objective when (1) $\mathcal{L}$ is convex and (2) the model output is probability $p(y|I, x)$.

Baselines. In our RAG experiments, we compare with the following classic baselines. One classic objective to train a reranker (Ram et al., 2023; Lewis et al., 2020; Guu et al., 2020a) uses

$$-\log p(y|x) \approx -\log\left(\sum_{d \in \text{top-}k} p_\theta(d|x) \cdot p_{\text{LM}}(y|d, x)\right) \tag{17}$$

$$\text{where} \quad p_\theta(d|x) = \frac{\exp(s_\theta(d|x)/\tau)}{\sum_{d' \in \text{top-}k} \exp(s_\theta(d'|x)/\tau)}. \tag{18}$$

Furthermore, REPLUG (Shi et al., 2024) proposes minimizing

$$\text{KL}(p_\theta(d|x) \| q(d|x, y)) \quad \text{where} \tag{19}$$

$$p_\theta(d|x) = \frac{\exp(s_\theta(d|x)/\tau_1)}{\sum_{d' \in \text{top-}k} \exp(s_\theta(d'|x)/\tau_1)} \quad \text{and} \quad q(d|x, y) = \frac{\exp(p_{\text{LM}}(y|d, x)/\tau_2)}{\sum_{d' \in \text{top-}k} \exp(p_{\text{LM}}(y|d', x)/\tau_2)}. \tag{20}$$

In addition, Basu et al. (2024) proposes an objective that also takes expectation over the selected documents. However, our objective models a set's distribution, and their objective focuses on an

item's distribution. They propose two gradient estimators of their objective, and we compare with the superior one, which minimizes the per-sample loss

$$\sum_{d \in S} [-\log p_{\text{LM}}(y|x,d) - b] \cdot \log p_\theta(d|x) \tag{21}$$

where S contains k i.i.d. sampled docs following $p_\theta(\cdot|x)$.

Hyperparameters. For all the methods, we tune the learning rate in the grid {5e-6, 1e-5, 5e-5} and fix the batch size to 64. For all the methods that involve sampling from the multinomial distribution defined by the softmax of logits, we tune the temperature $\tau \in \{0.02, 0.05, 0.1, 0.5, 1.0, 2.0\}$. For REPLUG, we tune τ_1 in the previously mentioned search range and let $\tau_2 = 1$. For our method, we tune $\lambda \in$ {1e-3, 1e-4} in all the experiments.

Setup. We train on Wikitext-103, NQ, and TriviaQ all for maximally 3000 steps for all the methods. We experiment with $m \in \{8, 16, 32\}$ with $k = 3$.

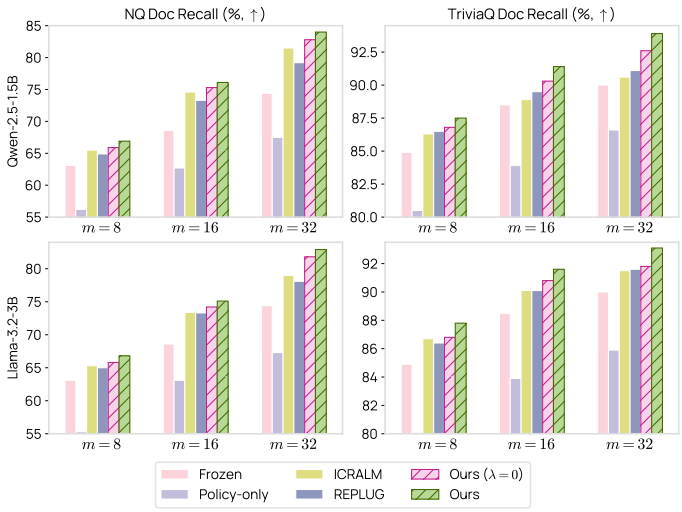


Figure 10: Document recall of rerankers trained via different methods.

Table 2: Perplexity on Wikitext-103 (↓).

Method	Qwen-2.5-1.5B			Llama-3.2-3B		
	$m = 8$	$m = 16$	$m = 32$	$m = 8$	$m = 16$	$m = 32$
Frozen	9.41	10.52	11.14	7.70	8.60	9.11
Policy-only	7.45	7.59	7.89	6.70	7.12	7.46
ICRALM	6.61	6.99	7.64	5.37	5.56	6.09
REPLUG	6.54	6.84	7.61	5.33	5.59	5.98
Supervised-only	6.52	6.79	7.26	5.31	5.56	5.89
Ours	**6.39**	**6.68**	**7.15**	**5.27**	**5.47**	**5.78**

C.3 Mixture-of-Experts

Hyperparameters. For all the methods, we tune the learning rate in the grid {1e-3, 2e-3, 5e-3, 1e-2} and fix the batch size to 128. For all the methods that involve sampling from the multinomial distribution defined by the softmax of logits, we tune the temperature $\tau \in \{0.02, 0.05, 0.1, 0.5, 1.0, 2.0\}$. For our method, we tune $\lambda \in$ {1e-1, 1e-2, 1e-3, 1e-4} in all the experiments. For the DefaultMoE baseline, we tune $\beta \in$ {0.6, 0.7, 0.8, 0.9, 0.99}.

Setup. For all the methods, we train for 25000 steps. Our model architecture uses n_embed 64, context window 256, n_head 8, and $m \in \{8, 16, 32, 64, 128\}$ with $k = 3$.

Table 3: Exact match accuracy on NQ (↑).

Method	Qwen-2.5-1.5B			Llama-3.2-3B		
	$m = 8$	$m = 16$	$m = 32$	$m = 8$	$m = 16$	$m = 32$
Frozen	23.7	26.2	26.5	28.3	30.4	31.4
Policy-only	25.7	29.7	30.8	29.8	31.9	33.2
ICRALM	27.5	29.7	30.8	30.8	34.1	35.0
REPLUG	26.9	29.7	31.1	31.2	33.5	34.5
Supervised-only	27.7	31.3	32.4	31.8	34.6	35.4
Ours	**28.3**	**32.2**	**33.6**	**32.4**	**35.1**	**36.9**

Table 4: Document recall on NQ (↑).

Method	Qwen-2.5-1.5B			Llama-3.2-3B		
	$m = 8$	$m = 16$	$m = 32$	$m = 8$	$m = 16$	$m = 32$
Frozen	63.1	68.6	74.4	63.1	68.6	74.4
Policy-only	56.2	62.7	67.5	55.3	63.1	67.3
ICRALM	65.5	74.6	81.5	65.3	73.4	79.0
REPLUG	64.9	73.3	79.2	65.0	73.3	78.1
Supervised-only	65.9	75.3	82.8	65.8	74.2	81.8
Ours	**66.9**	**76.1**	**84.0**	**66.8**	**75.1**	**82.9**

Table 5: Exact match accuracy on TriviaQ (↑).

Method	Qwen-2.5-1.5B			Llama-3.2-3B		
	$m = 8$	$m = 16$	$m = 32$	$m = 8$	$m = 16$	$m = 32$
Frozen	55.7	58.3	59.3	71.4	72.0	73.0
Policy-only	56.8	58.3	61.0	70.4	71.2	72.8
ICRALM	60.3	61.1	61.9	72.1	74.5	74.6
REPLUG	60.3	61.6	63.3	72.7	74.6	75.2
Supervised-only	61.2	62.6	65.2	73.4	74.9	75.7
Ours	**62.3**	**63.8**	**66.2**	**73.9**	**75.7**	**76.8**

Table 6: Document recall on TriviaQ (↑).

Method	Qwen-2.5-1.5B			Llama-3.2-3B		
	$m = 8$	$m = 16$	$m = 32$	$m = 8$	$m = 16$	$m = 32$
Frozen	84.9	88.5	90.0	84.9	88.5	90.0
Policy-only	80.5	83.9	86.6	80.1	83.9	85.9
ICRALM	86.3	88.9	90.6	86.7	90.1	91.5
REPLUG	86.5	89.5	91.1	86.4	90.1	91.6
Supervised-only	86.8	90.3	92.6	86.8	90.8	91.8
Ours	**87.5**	**91.4**	**93.9**	**87.8**	**91.6**	**93.1**

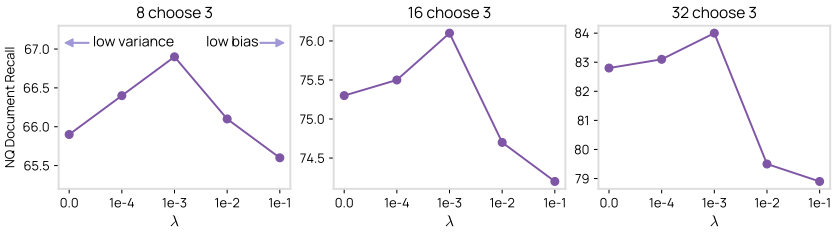


Figure 11: Bias/variance tradeoff of Qwen-2.5-1.5B adjusted by λ, whose optimal tradeoff is obtained at $\lambda \in (0, 1)$.

Table 7: Perplexity on Wikitext-103 by MoE (↓).

Method	One Layer					Two Layers			
	$m = 8$	$m = 16$	$m = 32$	$m = 64$	$m = 128$	$m = 8$	$m = 16$	$m = 32$	$m = 64$
Deterministic	86.33	84.44	86.66	86.06	85.82	71.07	69.45	68.56	69.21
Gaussian	85.29	83.11	82.75	81.90	83.26	69.47	67.17	68.06	66.14
DefaultMoE	87.52	89.42	87.37	86.99	84.94	70.80	70.59	69.65	67.76
SparseMixer	87.87	86.00	86.89	86.19	86.05	71.37	68.74	68.70	69.95
Supervised-only	84.80	80.99	77.97	74.73	74.39	69.02	64.94	62.72	59.67
Ours	**84.12**	**80.09**	**76.58**	**73.05**	**71.84**	**68.56**	**64.18**	**61.58**	**58.65**

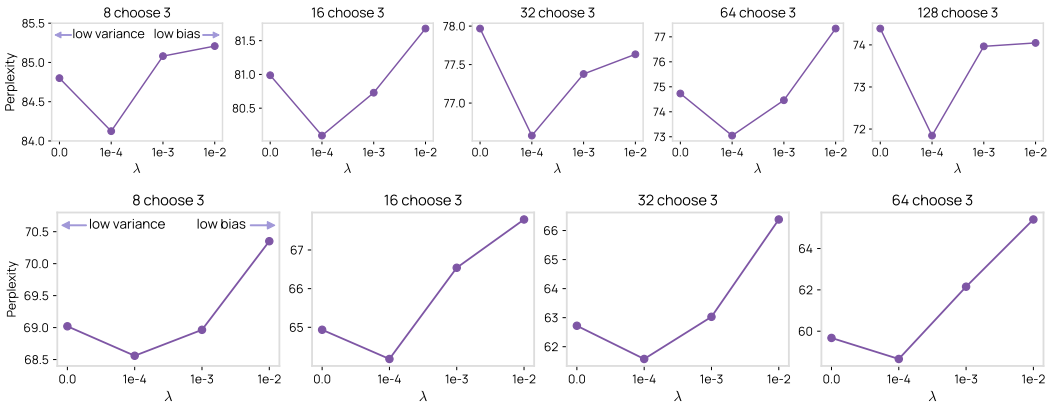


Figure 12: Bias/variance tradeoff adjusted by λ, showing that the optimal tradeoff is obtained at $\lambda \in (0, 1)$ under different m's. The upper row shows the results for 1 MoE layer, and the lower row shows the results for 2 MoE layers.

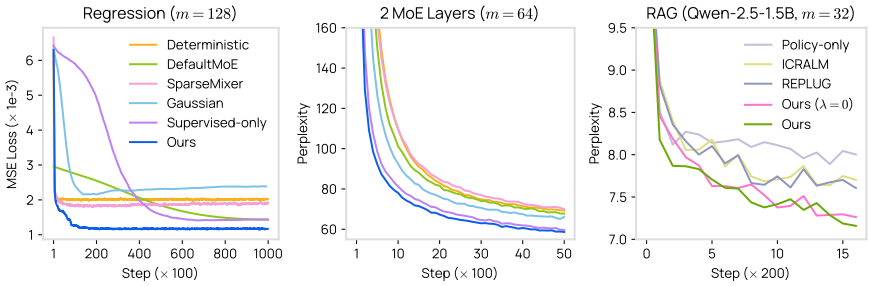


Figure 13: Convergence of difference methods for regression, MoE, and RAG applications.