MEASURING HETEROGENEOUS ADAPTATION: FROM NOMINAL CONFIGURATION TO INTERVENTION DEPENDENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern frozen-backbone adaptation systems combine heterogeneous controls, including available transformation banks, conditional allocation, and learned prompt-derived states. For such systems, nominal capacity scalars do not identify mechanism usage, intervention dependence, or downstream utility. We study how nominal configuration, structural availability, conditional allocation, realized actuation, same-checkpoint dependence, and downstream utility should be measured separately. We instantiate the study in VEFT+, which exposes these controls to observation and intervention. By varying residual-transformation availability (K), the number of subnetworks, and soft-prompt conditioning capacity (`sp_len`), the number of prompt states, we examine whether configurations with equal nominal products yield equivalent mechanism use or utility. Within the evaluated configurations, increasing either K or `sp_len` does not produce a metric-independent ordering, learned nonuniformity does not consistently outperform uniform composition, and evaluation-time prompt removal is not equivalent to training without the prompt branch. Ultimately, nominal allocation, realized composition, intervention dependence, and downstream utility operate as distinct properties in dual-axis adaptation.

1 INTRODUCTION

LLMs exhibit strong general-purpose capabilities, but efficient adaptation requires distinguishing what knowledge to adapt from how it is deployed. While conventional parameter-efficient fine-tuning (PEFT) methods reduce costs by training lightweight modules while freezing backbone parameters (He et al., 2022; Li et al., 2025a), and KADA addresses domain shift via latent knowledge alignment (Kawamae, 2025a), conditional computation via mixture-of-experts (MoE) introduces distinct routing and scaling challenges (Jordan & Jacobs, 1994; Fedus et al., 2022). As logical size and active compute can diverge, behavior depends heavily on weighting dynamics rather than raw capacity (Huang et al., 2024; Wang et al., 2024).

This paper studies a measurement problem in heterogeneous adaptation and proposes VEFT+ to address it. VEFT+ provides controllable transformation availability and prompt-side conditioning. We use configuration comparisons and same-checkpoint interventions to test whether nominal configuration is empirically sufficient to characterize realized mechanism use, checkpoint dependence, downstream utility, and measured cost. An adaptation system controls available transformations, their weighting, and conditional routing. These quantities cannot be adequately characterized by a single scalar. Just as Dynamic Hierarchical Knowledge Routing (DHKR) extends KADA toward dynamic capacity expansion (Anonymous, 2026a), VEFT+ extends VEFT within a dual-conditioned framework (Kawamae, 2025b). We propose VEFT+ as a controllable extension of VEFT and use it to examine whether a single scalar adequately characterizes heterogeneous adaptation. Cross-configuration comparisons alter both installed controls and learned checkpoints, and therefore include differences in optimization trajectory and co-adaptation.

As the principal study fixes K and L and disables capacity growth, this static mode separates capacity-composition effects from capacity-transition effects. Furthermore, to ensure internal validity,

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.

all interventions are evaluated on the same in-memory model, automatically restoring states post-evaluation and preventing simultaneous all-mode sweeps to avoid experimental confounds.

Contributions. Our contributions are as follows. (1) We formulate heterogeneous adaptation as a measurement problem separating nominal configuration, structural availability, conditional allocation, realized actuation, same-checkpoint dependence, and downstream utility. (2) We combine equal-nominal-product configuration comparisons with same-checkpoint interventions to evaluate distinct experimental dimensions. (3) We instantiate this protocol in VEFT+ to provide controllable transformation availability, prompt-side conditioning, and allocation. (4) Within the evaluated configurations, we find that the nominal product $B_{\mathrm{nom}} = K \times \texttt{sp_len}$ is not empirically sufficient to characterize the measured mechanism-use and metric profiles.

The results show that equal nominal products do not imply equivalent mechanism use or metric profiles, and that installed capacity, realized allocation, intervention dependence, and downstream utility require separate measurements. The contribution is a characterization methodology rather than a new optimizer or a universal allocation rule.

2 Previous Work

PEFT methods, including prompt tuning Lester et al. (2021), prefix tuning Li & Liang (2021), LoRA Hu et al. (2022), BitFit Zaken et al. (2022), and QLoRA Dettmers et al. (2023), adapt frozen backbones by inserting minimal task-specific parameters. These methods primarily differ in where adaptation parameters are placed and how much trainable state is introduced.

MoE architectures scale model capacity through conditional computation, activating only a subset of experts for each token. The classic hierarchical mixture-of-experts formulation dates back to Jordan & Jacobs (1994), and modern sparse MoE variants have demonstrated strong scaling behavior.

At the same time, sparse MoE models must manage expert load, communication cost, and routing stability (Guo et al., 2025; Wang et al., 2024; Huang et al., 2024), which has motivated work on dynamic expert activation, load balancing, and efficient deployment. Recent work studies adaptive expert capacity, allowing the model to adjust the number of experts and the number of active experts during training (Guo et al., 2025). Loss-Free Balancing shows that MoE expert load can be balanced through expert-wise routing-bias updates without adding auxiliary-loss gradients (Wang et al., 2024). However, its objective remains balanced sparse-MoE execution, which differs from VEFT, where concentration on a particular subnetwork is not treated as a failure. These studies provide a methodological comparison rather than a direct architectural equivalence. Sparse MoE models dispatch tokens to a selected subset of FFN experts, whereas VEFT forms a probability-weighted composition of knowledge-steering subnetworks. Accordingly, balanced expert load is not assumed to be an intrinsic objective of VEFT.

While MoE treats experts as computational specialization units, KADA follows the Lottery Hedge Fund Hypothesis (LHFH) (Kawamae, 2025a) and introduces subnetworks for knowledge alignment. DHKR extends this knowledge-routing perspective through dynamic hierarchical growth (Anonymous, 2026a). VEFT (Kawamae, 2025b) combines routed subnetworks with soft-prompt conditioning, while this paper focuses on its dual-axis capacity allocation. This perspective shifts the research question from "how many experts should be activated" to "what kind of knowledge should be aligned and how should it be routed."

Recent advances such as Progressive Prompts Razdaibiedina et al. (2023), Mixture of Prompts (MoPs) Dun et al. (2025), Sparse Mixture-of-Prompts (SMoP) Choi et al. (2023), and PT-MoE Li et al. (2025b) show that soft prompts can evolve, compose, or be routed. These methods primarily route or compose prompt-side components, leaving open how prompt capacity interacts with trainable knowledge-aligned subnetworks. KADA reframes domain adaptation as knowledge alignment using the Knowledge Steering Layer (KSL), while DHKR introduces dynamic weighting to grow subnetwork capacity K stably. Despite these advances, a fundamental gap remains: models like VEFT introduce a second adaptation axis through soft prompts `sp_len`, creating a dual-axis capacity problem. While DHKR addresses how to grow subnetworks, it does not determine when capacity should instead be allocated to prompt-side representations. Consequently, it remains unknown whether these two axes play redundant or distinct functional roles.

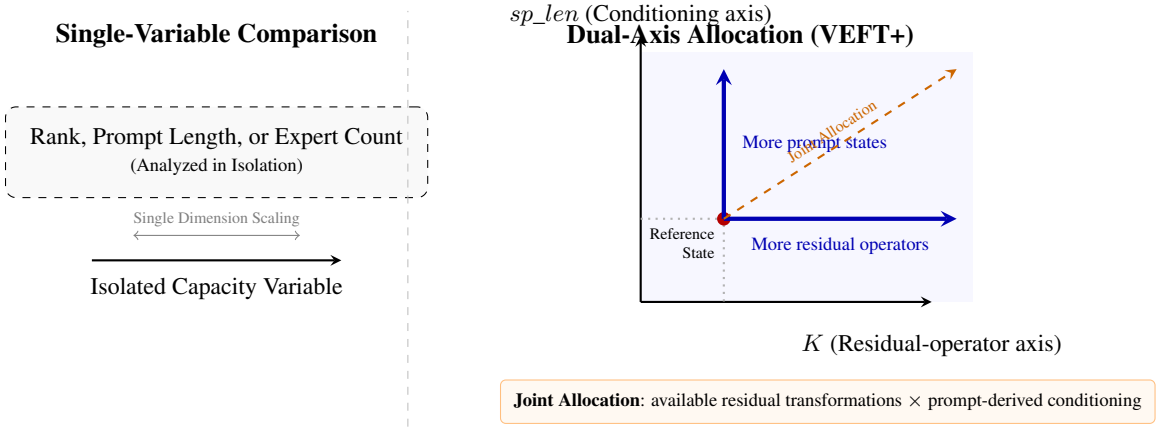


Figure 1: Conceptual comparison between a single-variable capacity analysis and the VEFT+ dual-axis allocation space. The horizontal axis changes the number of available residual transformations (K), while the vertical axis changes the number of learned soft-prompt states (sp_len). The two axes are separately parameterized and functionally non-equivalent, but not assumed to be mathematically orthogonal or causally independent.

VEFT+ is adopted here as a static mechanism-characterization framework. The primary experiments evaluate separate fixed configurations over K and `sp_len` while disabling natural and forced growth. Consequently, dynamic capacity control lies outside the principal estimand, preserving an intervention-based formulation that separates nominal allocation, realized composition, evaluation-time dependence, and downstream utility.

3 Measurement Framework and VEFT+ Instantiation

3.1 Overview

This work defines VEFT+ as a mechanism-characterization framework for dual-conditioned adaptation rather than a new optimizer or weighting mechanism. The main study compares static allocations across subnetwork count (K) and soft-prompt length (`sp_len`) at fixed hierarchy depth $L = 1$. As illustrated in Figure 1, we reframe adaptation capacity from a single scalar axis into a two-dimensional allocation space. The configuration varies a residual-transformation component (K) and a learned soft-prompt component that conditions the adaptation path. Treating these distinct functional roles as a unified variable obscures their operation. We instead distinguish the following measurable quantities:

$$\mathcal{G} = (C_{\text{nom}}, C_{\text{avail}}, A_{\text{alloc}}, U_{\text{act}}, D_{\text{int}}, V_{\text{task}}), \tag{1}$$

where C_{nom} denotes configured capacity, C_{avail} the available transformation set, A_{alloc} the realized conditional allocation, U_{act} the resulting representation displacement, D_{int} the response to a controlled same-model intervention, and V_{task} downstream utility. Because differences at an upstream stage do not necessarily propagate to downstream utility, the framework treats each quantity as requiring its own observable or intervention.

3.2 Dual-Axis Capacity Decomposition

The dual-axis formulation separates the set of available residual transformations from the learned continuous states that condition their use. Following VEFT, knowledge alignment and viewpoint conditioning motivate the two architectural controls. The measured variables in this study are transformation availability and prompt-side conditioning capacity. The experiments do not identify semantic specialization of individual transformations or establish viewpoint control from prompt length alone.

Transformation-availability axis Under KADA and DHKR, subnetworks are interpreted as latent knowledge units whose weighting reflects the model's internal knowledge distribution. Increasing

K expands the set of available residual transformations and the simplex over which conditional allocation is defined. It does not by itself establish that all available transformations are used or behaviorally valuable.

Soft-prompt axis (conditioning-capacity axis) In VEFT, soft prompts provide learned continuous control states that condition the adaptation path. The present experiments evaluate their allocation and evaluation-time dependence, but do not determine whether they merely reweight existing knowledge or encode task information of their own.

Implication These two controls modify distinct architectural objects: K alters transformation availability, whereas `sp_len` alters the parameterization of the conditioning state. Whether these modifications enhance usable knowledge or control expressivity remains an empirical question rather than an architectural guarantee. This distinction motivates our central question: *how do transformation availability and prompt-side conditioning differ in realized allocation, intervention dependence, utility, and measured cost within equal nominal-product groups?*

3.3 VEFT Formulation

We instantiate VEFT on top of a frozen language model backbone and a trainable Knowledge Steering Layer (KSL), and define VEFT+ as the setting that varies both the subnetwork and soft-prompt axes, following the knowledge-aligned adaptation framework of VEFT.

Let $H \in \mathbb{R}^{T \times d}$ denote hidden states from the frozen backbone. VEFT introduces two transformations:

1. a weighting-based subnetwork transformation parameterized by K (and optional hierarchy depth L),
2. a soft-prompt transformation parameterized by embeddings of length `sp_len`.

The output representation is defined as:

$$H' = H + \alpha\, \Delta(H; K, \texttt{sp_len}), \tag{2}$$

where $\Delta(\cdot)$ represents the joint effect of routed subnetworks and soft prompts, and α controls the contribution of the adaptation branch. The backbone parameters remain frozen, and all adaptation is performed through the VEFT branch.

3.4 Training Objective

The principal capacity study uses

$$\mathcal{L}_{\text{capacity}} = \mathcal{L}_{\text{task}} + \lambda_{\text{KL}}(s)\text{KL}(\bar{p}_s \| U_K), \tag{3}$$

where $\bar{p}_s$ is the mean first-level weighting distribution at step s and U_K is the uniform distribution over the active subnetworks. The implementation also supports the following extended regularized objective inherited from DHKR (Anonymous, 2026a):

$$\begin{aligned}\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{task}} + \lambda_{\text{KL}}(s)\,\text{KL}(\bar{p}_s \| U_K) + \lambda_{\text{bal}}(s)\,\frac{1}{K}\sum_{k=1}^{K}\left(\bar{p}_{s,k} - \frac{1}{K}\right)^2 \\ + \lambda_{\text{ent}}(s)\max\left(0,\, 0.2 - H(z_s)\right) + \lambda_{\text{TR}}(s)\text{KL}(r_{s-1} \| \bar{p}_s) + \lambda_{\text{KA}}\text{KL}(\tilde{q}_s \| \bar{p}_s) + \beta_{\text{MDL}}\frac{K_{\text{act}} + L_{\text{act}}}{\sqrt{1+s}}.\end{aligned} \tag{4}$$

The operational components are defined as follows:

- $\mathcal{L}_{\text{task}}$: Cross-entropy task supervision loss on response tokens (`ignore_index=-100`).
- $\text{KL}(\bar{p}_s \| U_K)$: Weighting distribution regularization toward a uniform baseline.
- Balance term: Mean squared deviation enforcing uniform subnetwork load.
- $\mathcal{L}_{\text{ent}}$: Hinge penalty enforcing weighting diversity via token entropy $H(z_s)$.

- $\mathcal{L}_{\mathrm{TR}}$: Trust-region KL divergence between EMA reference r_{s-1} and $\bar{p}_s$.
- $\mathcal{L}_{\mathrm{KA}}$: Alignment KL divergence between anchor distribution $\tilde{q}_s$ and $\bar{p}_s$.
- $\mathcal{L}_{\mathrm{MDL}}$: Step-dependent capacity complexity penalty.

This design bridges stability-aware knowledge weighting (KADA/DHKR) with MoE weighting balance and collapse avoidance Guo et al. (2025); Wang et al. (2024); Huang et al. (2024). The extended objective is used only in separately identified operating conditions. Raw training-loss values are not compared across the two objective families.

3.5 Design Principles

The contribution of this work is a formalization of the adaptation space in VEFT rather than a new weighting algorithm or optimizer. We make three claims:

1. **Reframing VEFT as KEFT.** We position VEFT within knowledge-effective fine-tuning systems rather than treating it as a hybrid of PEFT and MoE.
2. **Dual-axis capacity formulation.** We explicitly separate adaptation capacity into subnetwork and soft-prompt axes, corresponding to knowledge alignment and viewpoint control.
3. **Capacity allocation problem.** We replace conventional hyperparameter ablation with a structured comparison between increasing K and increasing `sp_len` while reporting stability and efficiency separately.

This formulation shows that VEFT adaptation is fundamentally a problem of allocating capacity between two interacting components, rather than a single-dimensional capacity-allocation problem.

4 Evaluation and Experimental Setup

Because VEFT+ is a capacity-fixed conditional adaptation system, task quality alone is insufficient to assess the benefits of additional capacity. A meaningful evaluation must answer three questions simultaneously: (i) does the model improve task performance, (ii) does the additional capacity maintain weighting stability, and (iii) does the added capacity preserve active-path efficiency? We use instruction-generation and factual-QA tasks to evaluate utility, calibration, weighting behavior, and runtime under the reported conditions. Accordingly, we evaluate VEFT+ along three dimensions: **task quality**, **weighting stability**, and **efficiency**.

4.1 Research questions

We use $B_{\mathrm{nom}} = K \times \texttt{sp_len}$ as a grouping variable for testing scalar sufficiency. It is not a parameter, compute, memory, or information budget. RQ1: Is the nominal product $B_{\mathrm{nom}} = K \times \texttt{sp_len}$ a sufficient statistic for the evaluated behavioral and efficiency profile? RQ2: How do transformation availability and prompt-side conditioning capacity relate to realized conditional allocation? RQ3: What properties are identified exclusively through same-model routing and prompt interventions rather than cross-configuration comparisons? RQ4: Do behavioral dependence and downstream utility coincide under the evaluated interventions?

The experiments are organized accordingly. Phase I: Primary study on AlpaGAsus: We first use AlpaGAsus Chen et al. (2024) as the primary study domain, and Alpaca Taori et al. (2023) as an auxiliary instruction-tuning reference. This choice is motivated by two reasons. First, it provides an instruction-following SFT setting with reference-based lexical and semantic generation metrics. Second, the existing VEFT logs already include matched runs with multiple soft-prompt lengths and runtime statistics, allowing a controlled first-stage analysis.

In this phase, we fix hierarchy depth to $L = 1$ to avoid confounding the comparison with hierarchical weighting instability. We vary K on the subnetwork axis and `sp_len` on the soft-prompt axis. The goal is not to establish a final optimum, but to isolate the first-order effect of each axis.

Phase II: Factual QA and calibration: We then evaluate factual QA on OpenBookQA (Mihaylov et al., 2018) and SciQ (Welbl et al., 2017). This phase tests whether the reported utility profile

extends beyond instruction generation. We therefore report both correctness and calibration. These outcomes do not directly identify semantic knowledge alignment. We then compare DHKR against representative PEFT baselines, analyze runtime, and validate the stabilizing components through ablations.

Phase III: Secondary External Operating-Point References We implement the PEFT citepeft baselines with the Hugging Face PEFT library [1] and compute automatic metrics with the Evaluate library.[2], for the fair comparisons and reproducibility. To establish contextual baselines without asserting state-of-the-art dominance, we use LoRA, AdaLoRA Zhang et al. (2023), and optionally BitFit Zaken et al. (2022) or QLoRA (Dettmers et al., 2023) depending on memory constraints. Representative PEFT baselines provide external operating points under the available unified implementation; they are not used to claim state-of-the-art performance or exhaustive superiority. Trainable parameter counts, wall-clock runtime, and downstream performance are reported as separate operating-point dimensions. Because VEFT introduces conditional weighting capacity, matching solely on parameter counts can obscure active-path computational overhead, whereas matching exclusively on runtime may conceal underlying representational divergences. Consequently, these comparisons serve strictly as external operating-point references to contextualize the dual-axis allocation behaviors rather than to establish exhaustive empirical superiority.

We define the ablation order as follows, while the reported main results focus on the fixed-depth setting $L = 1$: (A) soft-prompt-only growth: fix K and vary `sp_len`; (B) subnetwork-only growth: fix `sp_len` and vary K; (C) joint growth: vary both while reporting stability and compute separately; (D) hierarchy reintroduction: after identifying stable $L = 1$ configurations, evaluate $L > 1$ as future or supplementary analysis. This order is necessary because previous observations suggest that hierarchy can destabilize weighting and confound the interpretation of first-order axis effects.

4.2 Model and Training Protocol

We fine-tuned `Meta-Llama-3-8B-Instruct` AI@Meta (2024) using 4-bit NF4 quantization via `bitsandbytes` Dettmers et al. (2022). We deliberately hold the backbone fixed to isolate differences caused by dual-axis allocation and interventions, thereby preventing performance dependencies, tokenizer variations, hidden-width idiosyncrasies, or instruction-tuning recipes from confounding the evaluation. While this formulation employs standard hidden-state and residual interfaces, validating cross-family invariance remains an open empirical question. The base model weights were kept frozen, and only VEFT-specific adaptation modules, including routed subnetworks and soft prompts, were trained. The implementation follows the Hugging Face `transformers` and `peft` interfaces, including `GenerationMixin` and `output_hidden_states=True`. All prompts are converted into LLaMA-3-style chat format using `tokenizer.apply_chat_template`. Training is performed in a causal language modeling setup with masking of non-response tokens and a maximum sequence length of 1024. The principal equal-nominal-product study and same-model interventions report seeds 42, 43, and 44 where indicated; remaining single-run comparisons are explicitly labeled as descriptive.

5 Results

5.1 Primary Results on AlpaGAsus

We evaluate VEFT+ on the AlpaGAsus dataset as our primary benchmark. The objective is to determine whether prompt-side capacity and subnetwork-side capacity play functionally distinct roles, rather than merely assessing if increased total capacity improves performance. To ensure a controlled comparison, we group configurations by the nominal product $B_{\text{nom}} = K \cdot \texttt{sp_len}$. This grouping matches control-slot allocation and does not assume parameter, FLOP, or information-capacity equivalence.

Table 1 presents model performance across capacity allocations between subnetwork count K and prompt length `sp_len`. For the three-seed $B_{\text{nom}} = 4$ comparison, no single allocation has the highest mean on all metrics: the $(2, 2)$ configuration achieves the highest mean ROUGE-L,

[1] `https://github.com/huggingface/evaluate`
[2] `https://github.com/huggingface/evaluate`

Table 1: Three-seed equal-nominal-product comparison on AlpaGAsus. Values for $(1, 4)$, $(2, 2)$, and $(4, 1)$ are mean $\pm$ sample standard deviation over seeds 42, 43, and 44. The product $B_{\text{nom}} = K \times \texttt{sp_len} = 4$ matches the nominal number of controllable operator and prompt-state slots; it does not imply equivalence in parameters, FLOPs, information capacity, or realized weighting utilization. Bold indicates the highest mean, not statistical significance.

K	sp_len	BLEU	R-L	ME	BS	SB	SBC	Latency (ms)
Equal nominal product $B_{\text{nom}} = 4$								
1	4	0.0741±0.0031	0.2216±0.0032	0.3086±0.0070	0.8676±0.0016	7.409±0.312	0.7686±0.0064	174.41
2	2	0.0771±0.0007	**0.2317**±0.0040	**0.3189**±0.0033	**0.8695**±0.0007	7.710±0.075	0.7740±0.0030	151.43
4	1	**0.0774**±0.0020	0.2283±0.0056	0.3151±0.0065	0.8691±0.0015	**7.738**±0.203	**0.7744**±0.0060	150.74
Nominal product $B_{\text{nom}} = 8$								
1	8	0.07854	0.23141	0.32240	0.87009	7.8542	0.77883	179.24
2	4	0.07946	0.23261	0.31874	0.87095	7.9464	0.78046	178.32
4	2	0.07834	0.23127	0.31519	0.86874	7.8337	0.78551	168.59
Nominal product $B_{\text{nom}} = 16$								
1	16	0.07165	0.21978	0.30063	0.86683	7.1645	0.77211	175.95
2	8	0.07890	0.22962	0.31484	0.86863	7.8902	0.78130	175.62
4	4	0.07503	0.22682	0.31596	0.86865	7.5032	0.77894	170.89

Table 2: Same-model weighting intervention on AlpaGAsus. For each seed, a single $K = 2$, $L = 1$, $\texttt{sp_len} = 0$ model is trained once and evaluated under learned, uniform, and sequence-mean composition. Values are mean $\pm$ sample standard deviation over seeds 42, 43, and 44. All learned parameters, evaluation samples, and decoding settings are held fixed within each seed. Bold indicates the highest mean and does not imply statistical significance.

Weighting	BLEU	R-L	ME	BS	SB	SBC
Learned	0.076±.002	0.224±.004	0.308±.006	0.868±.001	7.564±.213	0.774±.006
Uniform	0.076±.001	**0.231**±.001	**0.310**±.005	**0.868**±.000	7.556±.103	**0.777**±.001
Sequence mean	0.074±.001	0.223±.005	0.306±.002	0.867±.001	7.436±.190	0.772±.003

METEOR, and BERTScore-F1, whereas $(4, 1)$ yields the highest mean BLEU, SacreBLEU, and SBERT similarity. This provides evidence against a metric-independent single-axis allocation rule over the tested $B_{\text{nom}} = 4$ configurations. The $B_{\text{nom}} = 8$ and $B_{\text{nom}} = 16$ rows are exploratory single-run observations. They are not used to establish cross-seed robustness or to support the principal claim. These results provide evidence against treating the nominal product as a sufficient statistic for the evaluated metric profile. They also provide evidence against a metric-independent monotonic allocation rule within the tested configurations.

The uniform result shows that learned nonuniformity is not itself a sufficient indicator of utility. The sequence-mean result separately shows that replacing token-wise allocation with a static within-sequence composition changes the metric profile. These two contrasts identify different properties and should not be collapsed into a single weighting-quality ranking.

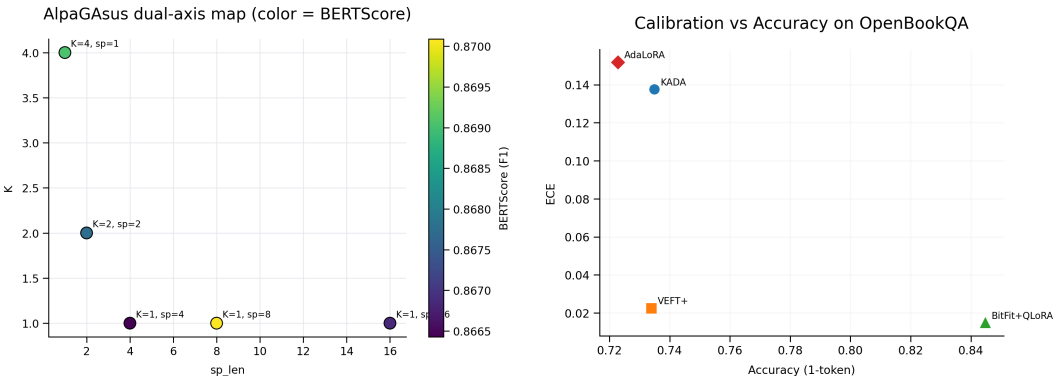
Same-model soft-prompt intervention. We evaluated the $K = 2$, $L = 1$, $\texttt{sp_len} = 8$ configuration across seeds 42, 43, and 44, comparing the learned soft prompt against an evaluation-time zero-prompt intervention while holding all trained parameters and weighting states fixed. The zero-prompt intervention identifies dependence on the final learned prompt value conditional on the trained checkpoint. It does not identify the total training-mediated contribution of the prompt branch, because the remaining parameters were jointly optimized in the presence of that branch. As summarized in Table 3, zeroing the prompt does not reduce the mean of any reported generation metric. Because these differences are small relative to the observed between-seed variation, this intervention tests evaluation-time dependence rather than establishing a performance gain.

Figure 2(a) summarizes the dual-axis landscape over $(K, \texttt{sp_len})$. This panel visualizes the non-monotonic metric and latency profiles across the evaluated configurations.

AlpaGAsus results are consistent with different roles for the two axes. Prompt-side allocation changes the learned context provided to the weighting path, whereas subnetwork-side allocation changes the set of available residual transformations. The weighting intervention further shows that weighting activity and weighting usefulness are distinct: wider weighting freedom does not automatically

Table 3: Same-model soft-prompt intervention on AlpaGAsus over seeds 42, 43, and 44. The weighting mode is fixed to learned, and all trained parameters, evaluation samples, and decoding settings are held fixed. Values report mean ± sample standard deviation. Bold indicates the highest mean, not statistical significance.

Prompt	BLEU	R-L	ME	BS	SB	SBC
Learned	0.07664±0.00267	0.23017±0.00065	0.31220±0.00645	0.86877±0.00038	7.6642±0.2670	0.77233±0.00777
Zero	**0.07683**±0.00297	**0.23376**±0.00164	**0.31394**±0.00688	**0.86924**±0.00036	**7.6830**±0.2966	**0.77279**±0.00848



(a) Dual-axis trade-off map on AlpaGAsus.

(b) Calibration vs accuracy on OpenBookQA.

Figure 2: Complementary evaluations of allocation and predictive behavior. Panel (a) visualizes the AlpaGAsus dual-axis allocation landscape; panel (b) reports the OpenBookQA calibration–accuracy comparison. Because the panels use different tasks and experimental grids, panel (b) is secondary validation and is not evidence for the AlpaGAsus interaction itself.

improve generation quality. The current results do not yet causally identify viewpoint control or semantic specialization of individual operators. The present evidence therefore motivates VEFT as a structured capacity-allocation problem rather than a one-dimensional capacity-allocation rule.

5.2 Secondary Validation on Alpaca: Instruction-Following Comparison

Table 4 summarizes early- and late-stage generation performance across the base model, QLoRA, AdaLoRA, DHKR variants, and VEFT+. This table provides a reported-run comparison on the instruction-following side rather than a strictly update-matched comparison.

5.3 Secondary Validation on OpenBookQA and SciQ

We evaluate VEFT+ on OpenBookQA and SciQ to examine the stability of predictive performance and calibration across different configurations, as shown in Table 5. OpenBookQA results show that BitFit+QLoRA achieves the highest raw accuracy. Figure 2(b) reports the corresponding accuracy–calibration operating points. VEFT+ does not dominate all external operating points on OpenBookQA. SciQ is near ceiling for all reported methods and therefore provides limited discrimination. These results are reported as secondary accuracy–calibration references.

6 Discussion and Limitations

The configuration and intervention studies answer different questions. The equal-nominal-product comparison characterizes separately trained systems. The routing and prompt interventions characterize dependence within fixed trained checkpoints. Their results show why nominal configuration, realized allocation, intervention dependence, downstream utility, and measured cost should be reported separately. Several limitations should be noted. First, while baseline $B_{\text{nom}} = 4$ comparisons use three seeds, broader $B_{\text{nom}} = 8$ and $B_{\text{nom}} = 16$ sweeps remain descriptive, and evaluations are based on a 4-bit quantized LLaMA-3-8B-Instruct backbone. Second, the experiments do not establish

Table 4: Comparison of model performance on Alpaca at Epoch 1 across multiple evaluation metrics. This table reports model-level comparisons under the available instruction-following runs.

Model	BLEU	R-L	ME	BS	SB	SBC
Base model	0.0524	0.2772	0.3432	0.8734	5.2378	0.8000
QLoRA	0.0770	0.2055	0.2905	0.8725	7.6992	0.7312
AdaLoRA	0.0606	0.2813	0.3533	0.8731	6.0584	0.7997
DHKR and VEFT+ start with $K = 1$ and expand to $K = 2$						
DHKR	0.0474	0.2641	0.3503	0.8719	4.7358	0.8143
VEFT+	0.0796	0.2535	0.3575	0.8738	7.9560	0.7444

Table 5: Accuracy and calibration results on OpenBookQA (upper block) and SciQ (lower block).

Method / Dataset	Acc 1Tok	Acc LblOnly	Acc NLL	ECE	Brier
DHKR ($K = 2$)	0.7349	0.7349	0.6028	0.0207	0.0927
VEFT+ ($K = 2$)	0.7339	0.7329	0.5958	0.0225	0.0923
BitFit+QLoRA	0.8448	0.8448	0.6300	0.0149	0.0558
AdaLoRA	0.7228	0.7218	0.6220	0.1518	0.1072
DHKR ($K = 2, L = 1$)	0.9974	0.9974	0.8878	0.0018	0.0011
VEFT+ ($K = 2, L = 1$, sp_len $= 2$)	0.9979	0.9979	0.8968	0.0023	0.0006
LoRA	0.9940	0.9940	0.8223	0.0027	0.0021

semantic specialization of transformations, a unique viewpoint interpretation of prompt states, mathematical axis orthogonality, training-time prompt necessity, or universal cross-architecture trends. Finally, our analysis is restricted to the $L = 1$ regime, leaving multi-level hierarchical weighting ($L > 1$) for future work. While concurrent work provides a stage-wise transmission analysis from prompt composition to actuation Anonymous (2026b), GASP adopts this decomposition as analytical background to investigate a distinct question: whether heterogeneous installed capacities reduce to a nominal scalar.

7 Conclusion

In this paper, we see that equal nominal products do not determine a unique measured profile in the evaluated VEFT+ configurations. The routing interventions distinguish learned nonuniformity from token-wise allocation variation. The prompt intervention measures dependence on the final prompt value, not the outcome of training without the prompt branch. These results motivate reporting nominal configuration, realized allocation, intervention dependence, utility, and measured cost as separate quantities.

References

AI@Meta. Llama 3 model card. 2024.

Anonymous. Dhkr: Dynamic hierarchical knowledge routing for efficient low-resource alignment. In *Under review*, 2026a.

Anonymous. Decoding-aware view routing for stable frozen vlm adaptation. In *Under review*, 2026b.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. Alpagasus: Training a better alpaca with fewer data. In *ICLR*, 2024.

Joon-Young Choi, Junho Kim, Jun-Hyung Park, Wing-Lam Mok, and SangKeun Lee. SMoP: Towards efficient and effective prompt tuning with sparse mixture-of-prompts. In *EMNLP*, pp. 14306–14316, 2023.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization. *ICLR*, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *NeurIPS*, 2023.

Chen Dun, Mirian del Carmen Hipolito Garcia, Guoqing Zheng, Ahmed Hassan Awadallah, Robert Sim, and Anastasios Kyrillidis. Sweeping heterogeneity with smart mops: Mixture of prompts for LLM task adaptation. In *AAAI*, pp. 16426–16434, 2025.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23:120:1–120:39, 2022.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, 2017.

Yongxin Guo, Zhenglin Cheng, Xiaoying Tang, Zhaopeng Tu, and Tao Lin. Dynamic mixture of experts: An auto-tuning approach for efficient transformer models. In *ICLR*, 2025.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *ICLR*, 2022.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*. OpenReview.net, 2022.

Haiyang Huang, Newsha Ardalani, Anna Y. Sun, Liu Ke, Shruti Bhosale, Hsien-Hsin S. Lee, Carole-Jean Wu, and Benjamin Lee. Toward efficient inference for mixture of experts. In *NeurIPS*, 2024.

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

Noriaki Kawamae. Knowledge-aligned domain shift tuning for efficient adaptation in large language models. In *KDD*, pp. 1128–1138, 2025a.

Noriaki Kawamae. One-shot multi-view visual conversational recommendation. In *KDD*, pp. 1118–1127, 2025b.

Alon Lavie and Abhaya Agarwal. METEOR: an automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation, WMT@ACL 2007*, pp. 228–231, 2007.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, pp. 3045–3059, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL/IJCNLP*, pp. 4582–4597, 2021.

Zongqian Li, Yixuan Su, and Nigel Collier. A survey on prompt tuning. *CoRR*, abs/2507.06085, 2025a.

Zongqian Li, Yixuan Su, and Nigel Collier. PT-moe: An efficient finetuning framework for integrating mixture-of-experts into prompt tuning. In *NeurIPS*, 2025b.

Chin-Yew Lin. Rouge: a package for automatic evaluation of summaries. In *ACL-workshop*, pp. 25–26, 2004.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pp. 311–318, 2002.

Matt Post. A call for clarity in reporting BLEU scores. In *WMT*, pp. 186–191, 2018.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *ICLR*, 2023.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP-IJCNLP*, 2019.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpaca: A Strong, Replicable Instruction-Following Model, mar 2023. URL `https://crfm.stanford.edu/2023/03/13/alpaca.html`.

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

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, 2017.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *ACL Volume 2: Short Papers*. ACL, 2022.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *ICLR*. OpenReview.net, 2023.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with BERT. In *ICLR*, 2020.

8 AI USE STATEMENT

We used AI tools to edit the manuscript for grammar and readability, assist with literature searches and identify relevant literature, format references, and edit and format tables. All reported results were produced by the authors' experimental pipeline. We manually verified the cited literature, numerical results, code changes, and final claims.

A APPENDIX

A.1 TASK QUALITY

For instruction-following and generation tasks, we report BLEU Papineni et al. (2002), ROUGE-L Lin (2004), METEOR Lavie & Agarwal (2007), BERTScore Zhang et al. (2020), SacreBLEU Post (2018), Sentence-BERT cosine similarity Reimers & Gurevych (2019). This multi-metric approach ensures that evaluation captures changes in both surface-level lexical overlap and underlying semantic fidelity, which can vary independently as capacity shifts. For factual QA tasks, we report Accuracy/OneToken, Accuracy/LabelOnly, Accuracy/NLL(mean), ECE Guo et al. (2017), and Brier score Brier (1950). This combination evaluates both correctness and confidence calibration.

A.2 WEIGHTING STABILITY AND EFFICIENCY

The dual-axis formulation predicts that nominal capacity may fail to translate into useful realized actuation because allocation, transformation geometry, and downstream sensitivity can attenuate its effect. Weighting concentration is reported descriptively and is not defined as collapse. We therefore track: $p1_{\max}$, the effective number of active subnetworks N_{eff}, normalized weighting entropy $H/\ln K$ for $K > 1$, allocation variability, trust-region magnitude (TR), and overload statistics when hierarchy is enabled. These metrics are motivated by the MoE and DHKR literature, where effective capacity depends on load balance, weighting entropy, and stability rather than nominal expert count alone Guo et al. (2025); Wang et al. (2024); Huang et al. (2024); Anonymous (2026a).

Because logical model size and actual active compute diverge under conditional weighting, we also report train runtime, samples/sec, steps/sec, approximate GPU inference latency, FLOPs/token, and trainable parameter count. These metrics allow us to distinguish between nominal capacity growth, active-path computational cost, and practical deployment efficiency.

B Cross-Backbone Mechanism Audit

We additionally instantiated the audit on a frozen Mistral backbone with persona-specific PersonalLLM supervision. This experiment is not a second capacity grid and is not used to claim personalization superiority. Its purpose is to test whether pathway optimization, realized allocation, same-checkpoint behavioral dependence, and downstream utility remain empirically separable under a different backbone and conditioning interface.

Across three training-seed runs, the gate norm increased by 4.6332 ± 0.6649, and both the VEFT branch and soft-prompt components received nonzero gradients. The final routing distributions were nonuniform but differed in operator identity across runs, consistent with permutation symmetry and run-dependent allocation.

Disabling residual actuation on the same in-memory trained model changed each of the nine logged paired generations across the three runs. This establishes behavioral dependence for the logged intervention examples, not aggregate utility improvement. Preference margin was consistently positive (0.1231 ± 0.0008), whereas pairwise preference accuracy remained close to chance (0.5317 ± 0.0029). The result therefore separates pathway optimization and behavioral dependence from strong downstream utility.

Because the central hypothesis is not merely that more capacity improves performance, but that useful capacity must remain weighting-stable, we also visualize weighting behavior across the equal-nominal-product AlpaGAsus settings. As shown in Figure 3, uniform composition achieves higher mean ROUGE-L, METEOR, BERTScore-F1, and SBERT similarity, while BLEU and SacreBLEU remain comparable to learned weighting. The learned condition has higher three-seed means than the sequence-mean condition on all reported metrics. Because these interventions hold the trained model fixed while altering the weighting strategy, the results distinguish learned nonuniformity from token-wise weighting variation, without establishing semantic operator specialization or universal superiority.

B.1 Runtime Analysis

Table 6 reports trainable parameters and iteration time on OpenBookQA. VEFT+ records an iteration time close to DHKR and approximately 38% of the AdaLoRA iteration time under the reported OpenBookQA protocol. Following DHKR, we interpret trainable-parameter count and runtime as separate systems characteristics because runtime also depends on module placement, execution path, active computation, and software integration (Anonymous, 2026a). The end-to-end measurement does not isolate the contribution of each factor.

Table 6: Trainable parameters are calculated as the sum of parameters with `requires_grad=True`; timing is reported from the same OpenBookQA runtime protocol used in DHKR (Anonymous, 2026a).

Method	Trainable Params	Trainable %	Time / Iter (sec)
BitFit	1,703,936	0.0212%	18.54
AdaLoRA	5,112,576	0.0636%	15.00
DHKR	637,923,344	10.97%	5.67
VEFT+	637,931,536	10.97%	5.68

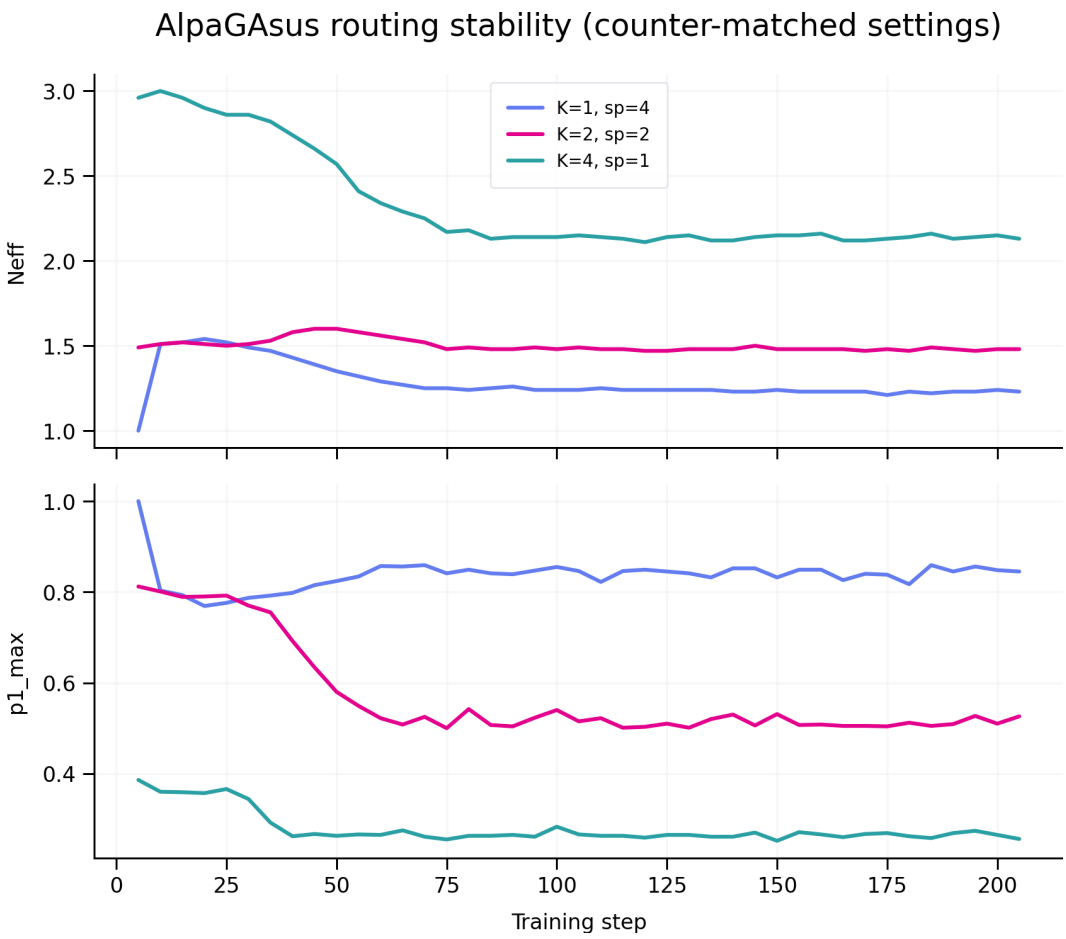


Figure 3: Nominal and realized subnetwork utilization on AlpaGAsus for equal-nominal-product settings $(1, 4)$, $(2, 2)$, and $(4, 1)$. The curves illustrate the evolutionary trajectories of effective active subnetworks (N_{eff}) and top-1 weighting concentration ($p1_{\max}$) throughout training. Although nominal bank capacity (K) remains fixed within each run, realized utilization diverges between training and inference phases and fails to scale proportionally with bank size. Concentration metrics are monitored descriptively to characterize weighting dynamics, avoiding any classification of weight skewness as weighting collapse or as an inherently suboptimal regime.

C TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

This appendix gives the operational definitions used in this paper. These definitions are provided to make the reported VEFT+ experiments self-contained; they summarize the implementation-level regularizers used in our evaluation setting rather than introducing new loss families.