CHANNEL-GROUP HIERARCHICAL ALLOCATION FOR MIXED PRECISION

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-precision quantization can substantially reduce the memory and computation costs of Transformer inference, but aggressive 4-bit floating point (FP4) quantization often degrades model quality due to highly non-uniform quantization sensitivity across the model. Mixed-precision quantization addresses this challenge by assigning higher precision to sensitive regions while keeping less sensitive regions in lower precision. Existing fine-grained mixed-precision methods often select high-precision regions directly from local or global sensitivity scores, which can over-concentrate the high-precision budget in a subset of the model while leaving other sensitive regions insufficiently protected. We introduce CHAMP (**C**hannel-group **H**ierarchical **A**llocation for **M**ixed **P**recision), a post-training framework combining FP4 and 8-bit floating point (FP8) that hierarchically distributes a global high-precision budget across model components and Transformer layers and constructs the resulting fine-grained channel-group precision mask using sensitivity-aware selection. CHAMP combines sensitivity- and size-aware module allocation, layer-specific adjustment, and local–global channel-group selection. Across language and video generation tasks, CHAMP provides the largest improvements when the FP8 budget is limited, outperforming prior fine-grained sensitivity-based mixed-precision allocation in this regime. CHAMP narrows the gap to full FP8 in our primary language-model settings, improves video frame-level fidelity, maintains comparable generation quality, and achieves higher theoretical throughput.

1 INTRODUCTION

Modern Transformer models, including large language models (LLMs) and video diffusion models, continue to grow in model size and computational demand, creating significant challenges for efficient inference Liu et al. (2024). To reduce these costs, a growing body of work has explored low-precision quantization of model weights and activations Ashkboos et al. (2024); Xiao et al. (2023). By representing model weights $\mathbf{W}$ and intermediate activations $\mathbf{A}$ in low-precision numerical formats, particularly 4-bit floating point (FP4), memory footprint and computational cost can be substantially reduced, making FP4 an attractive choice for efficient LLM inference Li et al. (2025a); Egiazarian et al. (2025). However, aggressive FP4 quantization of both weights and activations can introduce substantial accuracy degradation, limiting its practical use for low-precision inference.

A key challenge is that quantization sensitivity is highly non-uniform across the model. Different Transformer layers, projection modules within each layer (e.g., Q, K, V, and FFN projections), and individual feature channels can respond very differently to quantization Li et al. (2025b); Duanmu et al. (2025); Hooper et al. (2025). Mixed-precision post-training quantization addresses this problem by assigning higher precision, such as 8-bit floating point (FP8), to sensitive regions while keeping less sensitive regions in FP4. Under a limited FP8 budget, this raises two key questions: how should the high-precision budget be distributed across modules and Transformer layers, and which fine-grained regions within those allocations should be assigned FP8?

Existing methods address these questions at different granularities. Coarse-grained approaches allocate precision at the Transformer-layer or linear-projection-module level Li et al. (2025b); Duanmu et al. (2025), providing explicit control over coarse allocations but overlooking sensitivity variation within each module or layer. Finer-grained methods instead select individual feature channels Zheng et al. (2026); Chen et al. (2026) or tensor blocks. In particular, Fine-Grained Mixed-Precision

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.

(FGMP) Hooper et al. (2025) assigns higher precision to fine-grained weight and activation blocks based on sensitivity under a fixed global precision budget, better capturing local sensitivity variation. However, such fine-grained methods typically induce their module- and layer-level budgets implicitly through ranking or thresholding. Consequently, modules or layers with many highly ranked regions can absorb a disproportionate fraction of the global FP8 budget, leaving other sensitive components under-allocated. At finer granularities, this problem can be further amplified, as module- and layer-level budgets emerge from many individual ranking decisions and may become sensitive to small variations in calibration-based sensitivity estimates. This makes explicit control of the high-precision budget particularly important when FP8 is limited, since quantization sensitivity can vary substantially across model components Dong et al. (2020); Li et al. (2025b). These considerations motivate explicit control of the FP8 budget across modules and Transformer layers, together with an intermediate granularity that preserves fine-grained sensitivity variation.

To examine the allocation problem directly, we perform a diagnostic based on a global fine-grained ranking. We use *channel-groups*, i.e., fixed-size groups of adjacent feature channels that retain within-module sensitivity variation while reducing dependence on individual channel-level sensitivity estimates. Starting from a flat global ranking of channel-group sensitivities, we assign FP8 to the highest-scoring groups following the global ranking principle of FGMP Hooper et al. (2025). We then perform two greedy oracle diagnostics (Appendix B.5). At the module level, the oracle promotes a HiF4 channel-group in one module to FP8 and demotes an FP8 channel-group from another module, selecting the reallocation that yields the largest model-quality improvement at each step. At the layer level, we fix each module's total FP8 budget and similarly promote a channel-group in one Transformer layer while demoting one from another layer of the same module. The resulting quality improvements confirm that the allocation induced by flat global ranking can be suboptimal at both the module and layer levels, motivating hierarchical control of the high-precision budget.

To this end, we introduce **C**hannel-group **H**ierarchical **A**llocation for **M**ixed **P**recision (CHAMP). Given a global FP8 ratio, CHAMP allocates the high-precision budget hierarchically across modules and Transformer layers using aggregate and layer-specific sensitivity, then selects channel-groups within these budgets using a local–global policy. Each activation channel-group shares its precision with the corresponding weight input-feature group, restricting computation to matched $\mathbf{W}4\mathbf{A}4$ and $\mathbf{W}8\mathbf{A}8$ paths and avoiding the specialized support required by mismatched precisions from independent weight–activation selection Hooper et al. (2025). We evaluate CHAMP on both large language models and video diffusion models under FP4/MXFP8 mixed precision. For language tasks, CHAMP improves downstream accuracy over FGMP in the low-budget regime, with the largest gains when high precision is scarce. For video generation tasks, CHAMP improves frame-level fidelity over FGMP in our primary settings and maintains strong fidelity across a broad range of MXFP8 ratios while preserving comparable VBench generation quality. CHAMP also achieves a lower compute-weighted MXFP8 ratio and higher theoretical throughput than FGMP across the evaluated primary settings. These results show that explicitly structuring the high-precision budget across modules and Transformer layers can improve the effectiveness of fine-grained mixed-precision quantization. Our main contributions are:

- We introduce CHAMP, a hierarchical mixed-precision framework that separates high-precision budget allocation from fine-grained selection, allocating a global budget first across modules and Transformer layers and then across channel-groups.
- We demonstrate the effectiveness of CHAMP across language and video generation models, improving LLM accuracy in the low-budget regime and video frame-level fidelity while maintaining comparable generation quality. CHAMP also achieves higher theoretical throughput than FGMP across the evaluated primary settings.

2 Related Work

Sensitivity-driven spatial allocation. Sensitivity-based methods assign precision according to fine-grained region importance. CMPQ Chen et al. (2026) uses activation L2 norms, SliM-LLM Huang et al. (2025) uses salience, and AutoFQ Xie et al. (2025) combines conditioning, outlier, and Fisher statistics for channel-wise allocation. MixLLM Zheng et al. (2026) uses global feature ranking via Taylor and empirical Fisher information, while QVLA Xu et al. (2026) extends global channel-wise allocation to VLA models using action-space sensitivity. FGMP Hooper et al. (2025),

our closest baseline, scores fine-grained weight and activation blocks using Fisher-weighted quantization perturbation and applies a global threshold across layers. Unlike these methods, CHAMP explicitly structures the high-precision budget across modules and Transformer layers and performs channel-group-level precision selection within these allocations.

Hardware-structured weight–activation precision. Fine-grained mixed precision can introduce irregular execution patterns that require data reordering or specialized kernels Dettmers et al. (2022); Zhao et al. (2024); Zheng et al. (2026). Independent weight and activation selection can also introduce mismatched $\mathbf{W}4\mathbf{A}8$ and $\mathbf{W}8\mathbf{A}4$ paths Hooper et al. (2025). Hardware-oriented methods therefore impose more structured precision patterns. QServe Lin et al. (2025) uses a fixed W4A8KV4 configuration with co-designed GPU kernels, while MicroMix Liu et al. (2026) assigns matched MXFP4/6/8 precision to weight and activation channels. FGMP Hooper et al. (2025), in contrast, selects weight and activation blocks independently. CHAMP uses channel-group precision assignment and couples corresponding weight and activation groups, restricting computation to matched $\mathbf{W}4\mathbf{A}4$ and $\mathbf{W}8\mathbf{A}8$ paths. This yields a more regular execution pattern and avoids support for mismatched weight–activation precisions.

Temporal precision adaptation. Mixed precision has also been explored temporally in autoregressive and diffusion models. PMPD Chen et al. (2025) varies weight precision during decoding, while MoQAE Tao et al. (2025) adapts precision for KV-cache segments. For diffusion models, TMPDiff Lewandowski et al. (2026) varies precision across denoising timesteps, while 6Bit-Diffusion Su et al. (2026) adjusts precision across Transformer layers during denoising. MPQ-DMv2 Feng et al. (2025) instead compensates low-bit quantization with higher-precision residual components. These approaches are complementary to CHAMP, which focuses on spatially distributing a fixed high-precision budget across modules, Transformer layers, and channel-groups.

3 PRELIMINARIES

Low-Precision Quantization. We consider floating-point quantization of both weights and activations. In particular, HiF4 Luo et al. (2026) and MXFP8 Darvish Rouhani et al. (2023) are used as FP4 and FP8, respectively. Consider a linear layer with input activation $\mathbf{X} \in \mathbb{R}^{N \times F_{\text{in}}}$ and weight matrix $\mathbf{W} \in \mathbb{R}^{F_{\text{out}} \times F_{\text{in}}}$, where N, F_{in}, and F_{out} denote the sequence length, input feature dimension, and output feature dimension, respectively. The full-precision output is given by $\mathbf{Y} = \mathbf{X}\mathbf{W}^\top$. Let $Q(\cdot)$ denote a quantization operator. Quantizing activations and weights gives

$$\widehat{\mathbf{Y}} = Q(\mathbf{X})Q(\mathbf{W})^\top. \tag{1}$$

Following common notation, we denote a configuration with w-bit weights and a-bit activations as $WwAa$. Thus, $W4A4$ and $W8A8$ denote matched weight–activation computation using FP4 (HiF4) and FP8 (MXFP8), respectively. Lower-precision computation introduces quantization error,

$$\Delta Q(\mathbf{X}) = Q(\mathbf{X}) - \mathbf{X}, \tag{2}$$

which can lead to substantial output degradation for quantization-sensitive components.

Channel-level Mixed Precision Quantization. We consider a Transformer composed of L Transformer layers, indexed by l. Each Transformer layer contains M modules, indexed by m. In this paper, we use the term *module* to denote a quantized unit considered by our allocation framework, including linear projection modules as well as the query, key, value, and output tensors in self-attention. For each module m, let $\mathbf{X} \in \mathbb{R}^{N_m \times F_m}$ denote the tensor to be quantized, where F_m is the feature dimension and N_m denotes the number of tokens or tensor elements along the other dimension. This notation applies to both activations of linear projection modules and the tensors in self-attention. We partition the feature dimension F_m into non-overlapping groups of 128 features, referred to as *channel-groups* and indexed by c. This results in $C_m{=}F_m/128$ channel-groups for module m. Channel-level mixed-precision quantization assigns each channel-group either the high-precision MXFP8 format or the low-precision HiF4 format.

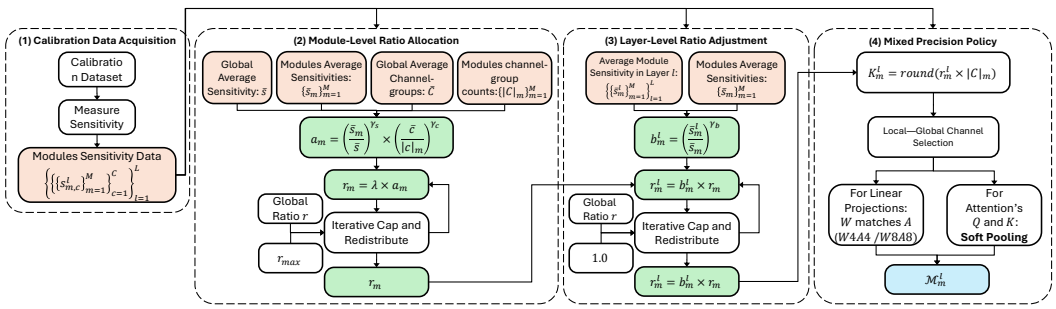


Figure 1: CHAMP Overview.

4 CHAMP ARCHITECTURE

CHAMP (Figure 1) first allocates the high-precision budget across modules based on sensitivity and size, then across Transformer layers, followed by channel-group selection. Let $r \in [0, 1]$ denote the fraction of channel-groups assigned to MXFP8. CHAMP proceeds in four stages: (1) *Calibration Data Acquisition* estimates channel-group quantization sensitivities $s_{m,c}^{l}$ for every module m in each Transformer layer l; (2) *Module-Level Ratio Allocation* distributes the global MXFP8 budget across modules and assigns each module a *base ratio*, which represents its average MXFP8 allocation across Transformer layers; (3) *Layer-Level Ratio Adjustment* adjusts each module's base ratio across Transformer layers according to layer-specific sensitivity; and (4) *Mixed-Precision Mask Construction* determines which channel-groups use MXFP8 and which remain in HiF4.

4.1 CALIBRATION DATA ACQUISITION

The first stage of CHAMP estimates the quantization sensitivity of each channel-group using a small representative calibration dataset $\mathcal{D}$. Similar to FGMP Hooper et al. (2025), for each channel-group c of module m in Transformer layer l, we compute a sensitivity score that combines the squared gradient magnitude of the calibration loss $\mathcal{L}$ with the corresponding quantization error. For sensitivity estimation, we instantiate $Q(\cdot)$ with HiF4 quantization:

$$s_{m,c}^{l} = \sum_{i \in \mathcal{D}} \sum_{j \in c} \left(\frac{\partial \mathcal{L}}{\partial X_{ij}} \right)^2 \left(\Delta Q(X_{ij}) \right)^2, \tag{3}$$

where $j \in c$ denotes the feature indices belonging to channel-group c. A large value of $s_{m,c}^{l}$ indicates that channel-group c is both sensitive to perturbations of the calibration loss and poorly represented under HiF4 quantization. Such channel-groups are stronger candidates for high-precision computation. The resulting sensitivity scores $\{\{\{s_{m,c}^{l}\}_{m=1}^{M}\}_{c=1}^{C_m}\}_{l=1}^{L}$ are used by the subsequent stages.

4.2 MODULE-LEVEL RATIO ALLOCATION

This stage distributes the global high-precision (MXFP8) budget r across module types m. For each module m with C_m number of channel-groups, we define s_m as its average quantization sensitivity across all Transformer layers and channel-groups:

$$s_m = \frac{1}{LC_m} \sum_{l=1}^{L} \sum_{c=1}^{C_m} s_{m,c}^{l}. \tag{4}$$

We then compute the module-level allocation score a_m as

$$a_m = \left(\frac{s_m}{\bar{s}} \right)^{\gamma_s} \left(\frac{\bar{C}}{C_m} \right)^{\gamma_c}, \tag{5}$$

where $\bar{s}$ and $\bar{C}$ are the geometric means of $\{s_m\}_{m=1}^{M}$ and $\{C_m\}_{m=1}^{M}$, respectively. We use geometric means since the allocation depends on relative ratios and to reduce the influence of commonly observed extreme sensitivity values Hooper et al. (2025). The hyperparameters $\gamma_s, \gamma_c \geq 0$ control the

relative influence of sensitivity and module size on the allocation. The first term assigns a larger allocation to modules that are more sensitive to quantization on average, and the second term assigns a relatively larger ratio to modules with fewer channel-groups. This design is motivated by prior work showing that larger layers can often tolerate more aggressive quantization, whereas smaller layers may benefit from higher precision Yang et al. (2021); Singh et al. (2025).

We convert the allocation scores a_m into base MXFP8 ratios r_m using a common scaling factor λ, chosen such that the overall allocation across all modules satisfies the global ratio r. However, highly sensitive modules can receive excessive FP8 allocation. Our sensitivity analysis shows that per-module caps can improve accuracy by reducing this concentration (Appendix B.3). We therefore address this imbalance with a *Cap-and-Redistribute* procedure.

Cap-and-Redistribute. This procedure iteratively caps each module ratio at $r_{\max}$ and redistributes any excess budget among the uncapped modules. We initialize the capped set as $\mathcal{S} = \emptyset$. At each iteration, the scaling factor λ is computed as

$$\lambda = \frac{r \sum_{m=1}^{M} C_m - r_{\max} \sum_{m \in \mathcal{S}} C_m}{\sum_{m \notin \mathcal{S}} a_m C_m}. \tag{6}$$

Each uncapped module is assigned a tentative ratio $\tilde{r}_m{=}\lambda a_m$. If $\tilde{r}_m > r_{\max}$, the corresponding module is added to $\mathcal{S}$ and its ratio is fixed at $r_{\max}$. The scaling factor λ is then recomputed over the remaining uncapped modules. This process is repeated until all tentative ratios satisfy the cap. Let $\lambda^\star$ denote the final scaling factor and $\mathcal{S}^\star$ the final set of capped modules, then resulting base ratios are $\{r_m\}_{m=1}^{M}{=}\{\, r_{\max} \mid m{\in}\mathcal{S}^\star \,\}{\cup}\{\, \lambda^\star a_m \mid m{\notin}\mathcal{S}^\star \,\}$. By construction, the resulting base ratios preserve r before discrete channel-group rounding: $\sum_{m=1}^{M} r_m C_m{=}r \sum_{m=1}^{M} C_m$ (see Appendix A.1).

4.3 Layer-Level Ratio Adjustment

This stage adjusts r_m across Transformer layers according to each layer's sensitivity relative to the module average. For module m in layer l, we define the average channel-group sensitivity:

$$\bar{s}_m^{\,l} = \frac{1}{C_m} \sum_{c=1}^{C_m} s_{m,c}^{\,l}. \tag{7}$$

We then define the layer-level adjustment factor as:

$$b_m^{\,l} = \left(\frac{\bar{s}_m^{\,l}}{\bar{s}_m} \right)^{\gamma_b}, \tag{8}$$

where $\gamma_b \geq 0$ controls the strength of the adjustment, and $\bar{s}_m$ is the geometric mean of the layer-level sensitivities for module m. For each m, the multipliers $\{b_m^{\,l}\}_{l=1}^{L}$ are normalized by their arithmetic mean so that $\frac{1}{L}\sum_{l=1}^{L} b_m^{\,l}{=}1$. The layer-specific MXFP8 ratio is then given by:

$$r_m^{\,l} = b_m^{\,l} r_m. \tag{9}$$

Consequently, $\frac{1}{L}\sum_{l=1}^{L} r_m^{\,l}{=}r_m$, so this stage redistributes the module's base ratio across Transformer layers without changing its average allocation. The overall hierarchical allocation preserves the global MXFP8 budget (see Appendix A.2).

To ensure that all layer-specific ratios remain valid, we apply the same **Cap-and-Redistribute** procedure, separately for each module across its Transformer layers, using a fixed upper bound of 1. The resulting $r_m^{\,l} \in [0, 1]$ denotes the fraction of channel-groups in module m at Transformer layer l assigned to MXFP8. Sensitivity patterns and the resulting module- and layer-specific MXFP8 allocations are visualized in Appendix A.4.

4.4 Mixed-Precision Policy

Given the final ratio $r_m^{\,l}$, we select $K_m^{\,l}{=}\,\mathrm{round}(r_m^{\,l} C_m)$ channel-groups to assign MXFP8 in module m of Transformer layer l. For linear projections, each activation channel-group and its corresponding weight input-feature group share the same precision, yielding only $\mathbf{W}4\mathbf{A}4$ or $\mathbf{W}8\mathbf{A}8$ computation. The same principle requires additional consideration for self-attention query and key. we enforce a shared channel-group precision mask using *soft pooling*.

Soft Pooling. We first apply the channel-group selection procedure independently to query and key, obtaining K^l_{query} and K^l_{key} channel-groups, respectively. We then take the union of the two selections, retaining any channel-group selected by either tensor. When the selections differ, the union may exceed the shared-mask budget, which is set to $\frac{1}{2}(K^l_{\text{query}} + K^l_{\text{key}})$ to preserve the total MXFP8 allocation across query and key. To enforce this budget, each channel-group c in the union is scored by $\min(s^l_{\text{query},c}, s^l_{\text{key},c})$, and the lowest-scoring channel-group is removed until the target budget is met. This favors channel-groups that are sensitive in both query and key and down-weights those that are insensitive in either tensor. The resulting shared mask is applied to both tensors.

Channel Selection. We use a local–global strategy to select channel-groups for MXFP8. For module m in Transformer layer l, half of the allocated K^l_m channel-groups are selected locally within the layer, while the remaining budget is pooled across all Transformer layers of the same module. Specifically, each layer first selects its $\lfloor K^l_m/2 \rfloor$ highest-sensitivity channel-groups. The remaining $K^l_m - \lfloor K^l_m/2 \rfloor$ channel-groups contribute to a global budget, which is allocated across layers by selecting the highest-sensitivity channel-groups that have not already been selected. This design balances local and global sensitivity. The local selection preserves a minimum MXFP8 allocation within each Transformer layer, whereas the global selection allows more sensitive channel-groups to receive MXFP8. A comprehensive ablation of the individual components of CHAMP is provided in Appendix C.2.

5 Experiments

5.1 Experimental Setup

Models, Tasks, and Metrics. We evaluate CHAMP on large language models and video diffusion models. For language modeling, we use Llama-3.1-8B Grattafiori et al. (2024) and, following FGMP Hooper et al. (2025), evaluate zero-shot accuracy with lm-eval-harness on ARC-Challenge, ARC-Easy, BoolQ, PIQA, WinoGrande, OpenBookQA, and RACE. For text-to-video generation, we use Wan 2.2-T2V-A14B Wang et al. (2025) and HunyuanVideo Kong et al. (2024), and generate videos from prompts adapted from the official Sora examples OpenAI (2024) (Appendix B.1). Following prior work Zhao et al. (2025b), we measure quantization-induced degradation relative to BF16 using MSE, PSNR, SSIM Wang et al. (2004), and cosine similarity, averaged over all frames and prompts. In addition, we evaluate absolute generation quality via VBench Huang et al. (2024) on six generation-quality dimensions: overall, subject, and background consistency, aesthetic and imaging quality, and motion smoothness.

Reference Configurations and Baseline. We compare CHAMP against: (i) **BF16**, the full-precision reference; (ii) **MXFP8**, which quantizes all channel-groups to MXFP8; (iii) **HiF4**, which quantizes all channel-groups to HiF4; and (iv) **FGMP** Hooper et al. (2025), implemented using FGMP's sensitivity-based mixed-precision selection with 128×128 blocks.

Quantization Settings. We assign each channel-group to either MXFP8 (high precision) or HiF4 (low precision), with a global MXFP8 ratio of $r{=}0.2$ unless stated otherwise. We apply CHAMP to the linear projection modules in the attention and feed-forward sublayers, and to the query, key, value and output tensors in self-attention. For linear projection modules, each activation channel-group and its corresponding input-feature group in the weight matrix share the same precision assignment (W8A8 or W4A4). For self-attention, query and key tensors use the shared precision mask constructed by the Soft Pooling strategy. Further implementation details are provided in Appendix B.4. The hyperparameter selection procedure and the values used for each model are provided in Appendix B.2. We analyze the sensitivity of CHAMP to these hyperparameters in Appendix B.3.

Calibration. Sensitivity scores in Equation 3 are estimated from a calibration set. For Llama-3.1-8B, we use 512 WikiText-2 samples Merity et al. (2017). For Wan 2.2-T2V-A14B and HunyuanVideo, we use 64 prompts from the Disney-VideoGeneration-Dataset Wild-Heart (2024).

Table 1: Zero-shot accuracy (%) of Llama-3.1-8B across seven downstream tasks. Values in parentheses denote the gap to MXFP8. Bold indicates the best result among FGMP and CHAMP.

Method	MXFP8 ratio r	ARC-C	ARC-E	BoolQ	PIQA	Wino	OpenBQ	RACE	Avg.
BF16	–	55.20	82.70	83.12	80.79	74.43	45.20	39.90	65.91
MXFP8	1.0	53.92	81.19	82.02	80.41	71.67	43.80	39.52	64.65
HiF4	0.0	49.66	77.95	79.48	78.24	68.82	42.20	38.85	62.17
FGMP	0.1	50.51	77.69	**80.43**	77.26	70.88	41.80	**39.04**	62.52 (−2.13)
CHAMP	0.1	**51.96**	**79.08**	80.24	**79.11**	**72.06**	**44.20**	38.47	**63.59** (−1.06)
FGMP	0.2	52.56	77.86	**80.31**	77.91	69.69	42.80	38.09	62.75 (−1.90)
CHAMP	0.2	**53.24**	**78.83**	79.02	**78.07**	**70.88**	**43.40**	**38.85**	**63.19** (−1.46)



Figure 2: Qualitative comparison of generated videos across methods at r=0.2. All results use prompt #0 from the official Sora examples OpenAI (2024).

5.2 Main Results

5.2.1 Large Language Models

Table 1 reports zero-shot accuracy on our reasoning benchmarks at MXFP8 ratios of r=0.1 and r=0.2. At the more aggressive r=0.1 setting, CHAMP achieves an average accuracy of 63.59%, improving over FGMP (62.52%) by 1.07 percentage points. Relative to HiF4 (62.17%) and full MXFP8 (64.65%), CHAMP recovers 57.3% of the accuracy gap while using only 10% MXFP8, compared with 14.1% for FGMP. At r=0.2, CHAMP achieves 63.19% average accuracy, compared with 62.75% for FGMP, and outperforms FGMP on six of the seven tasks. At this ratio, CHAMP recovers 41.1% of the HiF4-to-MXFP8 gap, compared with 23.4% for FGMP.

Interestingly, CHAMP achieves slightly higher average accuracy at r=0.1 than at r=0.2. The individual tasks show that the two precision masks induce different prediction changes across benchmarks: ARC-Challenge and RACE improve at r=0.2, whereas the remaining tasks decrease slightly. This non-monotonicity appears to reflect task-dependent changes induced by the different precision masks, rather than a consistent benefit from using a lower MXFP8 ratio.

Additional results across a wider range of MXFP8 ratios are reported in Appendix C.1.1. CHAMP provides its strongest gains at low ratios, while the differences between allocation strategies generally narrow as the high-precision budget increases, indicating that hierarchical allocation is most beneficial when high precision is scarce.

5.2.2 Video Diffusion Models

We evaluate Wan 2.2-T2V-A14B at MXFP8 ratios of r=0.1 and r=0.2, and HunyuanVideo at the higher ratio r=0.5 due to its larger degradation from full MXFP8 to HiF4 (see Figure 2). For HunyuanVideo, we additionally keep the self-attention query, key, and value tensors unquantized for all methods. Figure 2 provides a qualitative comparison of the generated videos. Across both models, CHAMP better preserves subject appearance, scene structure, visual details, lighting, and color characteristics than FGMP, producing outputs closer to the BF16 and full-MXFP8 references.

Tables 2 and 3 evaluate CHAMP on WAN 2.2-T2V-A14B and HunyuanVideo on frame-level fidelity and VBench scores, respectively. Table 2 shows that CHAMP consistently improves over

Table 2: Frame-level fidelity relative to BF16 for text-to-video generation. Values in parentheses denote the gap to MXFP8. Bold indicates the best result among FGMP and CHAMP.

Model	r	Method	Cosine ↑	SSIM ↑	PSNR (dB) ↑	MSE ↓
Wan 2.2-T2V-A14B	–	MXFP8	97.50	73.27	20.71	794
	–	HiF4	94.13	57.27	16.70	1712
	0.10	FGMP	96.01 (−1.49)	64.62 (−8.65)	19.21 (−1.50)	1275 (−481)
		CHAMP	**96.38** (−1.12)	**66.07** (−7.20)	**19.51** (−1.20)	**1139** (−345)
	0.20	FGMP	96.26 (−1.24)	66.56 (−6.71)	19.53 (−1.18)	1197 (−403)
		CHAMP	**96.89** (−0.61)	**67.72** (−5.55)	**19.67** (−1.04)	**999** (−205)
HunyuanVideo	–	MXFP8	95.75	54.07	15.83	2029
	–	HiF4	92.81	48.16	13.01	3474
	0.50	FGMP	94.83 (−0.92)	51.32 (−2.75)	14.57 (−1.26)	2499 (−470)
		CHAMP	**95.42** (−0.33)	**52.98** (−1.09)	**15.24** (−0.59)	**2200** (−171)

Table 3: VBench scores ($\times 100$) for text-to-video generation. AQ/IQ: aesthetic/imaging quality; OC/SC/BC: overall/subject/background consistency; MS: motion smoothness. Values in parentheses denote the gap to MXFP8. Bold indicates the best result among FGMP and CHAMP.

Model	r	Method	OC ↑	SC ↑	BC ↑	AQ ↑	IQ ↑	MS ↑	Avg. ↑
Wan 2.2-T2V-A14B	–	BF16	28.88	97.88	98.27	69.24	73.32	99.03	77.77
	–	MXFP8	29.26	97.99	98.40	69.74	73.02	99.00	77.90
	–	HiF4	28.66	97.44	97.69	67.63	71.36	99.00	76.96
	0.10	FGMP	**29.04**	**97.81**	**98.09**	**69.80**	72.10	**98.99**	**77.64** (−0.26)
		CHAMP	28.93	97.75	97.95	69.11	**72.39**	**98.99**	77.52 (−0.38)
	0.20	FGMP	**29.00**	**97.96**	98.13	70.15	**72.66**	**99.03**	**77.82** (−0.08)
		CHAMP	28.86	97.72	**98.21**	**70.43**	72.15	98.99	77.73 (−0.17)
HunyuanVideo	–	BF16	29.67	94.59	95.17	62.17	66.80	99.14	74.59
	–	MXFP8	29.69	94.60	95.35	63.15	69.45	99.17	75.24
	–	HiF4	20.33	93.36	95.16	50.38	48.33	99.26	67.80
	0.50	FGMP	28.09	93.72	95.44	**61.89**	65.87	**99.17**	74.03 (−1.21)
		CHAMP	**29.02**	**94.25**	**95.84**	61.43	**66.11**	99.13	**74.30** (−0.94)

FGMP under the same MXFP8 budget across both Wan ratios and HunyuanVideo. These results indicate that hierarchical allocation better preserves the BF16 output under quantization.

According to Table 3, CHAMP remains comparable to FGMP on Wan, with average differences of only 0.12 and 0.09 points at $r{=}0.1$ and $r{=}0.2$, respectively. On HunyuanVideo, CHAMP improves the average score by 0.27 points and outperforms FGMP on four of six dimensions, including all three consistency metrics. The largest gains are observed in overall, subject, and background consistency, while motion smoothness remains essentially unchanged.

Additional MXFP8 ratios are reported in Appendix C.1.2, where the broader sweep reveals consistent video-specific trends. On HunyuanVideo, CHAMP improves all four frame-level fidelity metrics at every evaluated ratio and achieves a higher VBench average through $r{=}0.75$. On Wan, CHAMP consistently achieves higher cosine similarity and lower MSE, while SSIM, PSNR, and VBench remain comparable to FGMP. Unlike our language-model results, the video fidelity gains persist across a broader range of MXFP8 budgets. This may be due to denoising, where quantization errors can accumulate across timesteps and make precision allocation critical even at higher ratios.

5.3 Theoretical Efficiency Analysis

To estimate the potential compute efficiency of mixed precision, we analyze theoretical throughput under an idealized compute-bound model. Let $\mathcal{T}$ denote BF16 throughput. Assuming throughput scales inversely with bit-width, MXFP8 and HiF4 achieve $2\mathcal{T}$ and $4\mathcal{T}$, respectively. Since our

experiments use fake quantization and dequantize tensors before PyTorch computation, we report theoretical rather than measured throughput.

Effective mixed-precision throughput. The global ratio r specifies the fraction of channel-groups assigned to MXFP8, but different modules incur different compute costs. We therefore define the compute-weighted MXFP8 ratio as the fraction of computation assigned to MXFP8:

$$\bar{r} = \frac{\sum_{l,m} r_m^l \, \mathrm{FLOPs}(m,l)}{\sum_{l,m} \mathrm{FLOPs}(m,l)}, \tag{10}$$

where $\mathrm{FLOPs}(m,l)$ denotes the computation performed by module m in Transformer layer l. For example, in a linear projection, $\mathrm{FLOPs}(m,l)=2N_m^l F_{\mathrm{in},m}^l F_{\mathrm{out},m}^l$, where N_m^l, $F_{\mathrm{in},m}^l$, and $F_{\mathrm{out},m}^l$ denote the number of input tokens, input feature dimension, and output feature dimension, respectively. The corresponding theoretical mixed-precision throughput is $(\frac{\bar{r}}{2\mathcal{T}} + \frac{1-\bar{r}}{4\mathcal{T}})^{-1} = \frac{4\mathcal{T}}{1+\bar{r}}$.

Table 4 compares the compute-weighted MXFP8 ratios and theoretical throughputs of FGMP and CHAMP under the same MXFP8 budget. Note that throughput estimates are computed over the quantized modules only. CHAMP achieves lower $\bar{r}$ and higher theoretical throughput. The gain over FGMP is largest on Llama-3.1-8B, where theoretical throughput increases from $3.10\mathcal{T}$ to $3.20\mathcal{T}$, corresponding to a 3.2% relative improvement. The parameter γ_c controls the influence of module size during budget allocation, providing a tunable trade-off between sensitivity and efficiency.

6 Conclusion

Key Findings. We introduced CHAMP, a hierarchical mixed-precision framework that jointly structures MXFP8 allocation and fine-grained precision-mask construction across modules, Transformer layers, and channel-groups. Across language and text-to-video generation tasks, CHAMP improves over FGMP in the low-MXFP8 regime, where careful allocation of the limited high-precision budget is most important. For video generation, these gains are most consistent in reconstruction fidelity, while VBench remains broadly comparable. Our analysis further shows that CHAMP can achieve a lower compute-weighted MXFP8 ratio and higher theoretical throughput while maintaining competitive model quality.

Table 4: Theoretical mixed-precision efficiency of CHAMP and FGMP at the evaluated MXFP8 ratios: r=0.20 for Llama-3.1-8B and Wan 2.2-T2V-A14B, and r=0.50 for HunyuanVideo.

Model (r)	Method	$\bar{r}\downarrow$	Throughput $\uparrow$
Llama-3.1-8B	FGMP	0.29	$3.10\mathcal{T}$
	CHAMP	**0.25**	$\mathbf{3.20}\mathcal{T}$
Wan 2.2-T2V-A14B	FGMP	0.19	$3.37\mathcal{T}$
	CHAMP	**0.17**	$\mathbf{3.42}\mathcal{T}$
HunyuanVideo	FGMP	0.57	$2.55\mathcal{T}$
	CHAMP	**0.56**	$\mathbf{2.56}\mathcal{T}$

Limitations and Future Directions. First, CHAMP relies on several allocation hyperparameters (γ_s, γ_c, and $r_{\max}$), whose optimal values can vary across models and workloads. Although our calibration-based proxy makes hyperparameter search inexpensive, it may not perfectly reflect downstream model quality. Grey-box optimization methods Astudillo & Frazier (2022) that combine this proxy with a limited number of downstream evaluations could improve hyperparameter selection while retaining low optimization cost. Second, our efficiency analysis is based on theoretical throughput. Realized speedups depend on hardware support and optimized mixed-precision kernels. Third, for video generation, the mixed-precision allocation remains fixed across denoising steps. A timestep-dependent ratio or precision mask could distribute the high-precision budget more effectively over time. Finally, our sensitivity scores are computed once from the calibration set and remain fixed throughout precision allocation. As a result, the score of a channel-group does not reflect which other channel-groups have already been assigned MXFP8, even though their quantization effects may interact. Sequential decision-making approaches, such as greedy marginal-gain selection, local search, or reinforcement learning, could instead adapt precision decisions to the current allocation and better capture these dependencies.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the models, datasets, calibration procedure, quantization settings, hyperparameter selection, evaluation protocol, and additional experiments to facilitate reproducibility. The source code cannot be publicly released due to restrictions on the release of the underlying implementation. We therefore provide the algorithmic and experimental details needed to enable independent implementation and evaluation of the proposed method.

AI USE STATEMENT

In this paper, we used generative AI tools to assist with the interpretation of empirical results, reference formatting, and revising portions of the manuscript for clarity, readability, and grammar. All AI-assisted suggestions were independently reviewed by the authors. We take responsibility for the final content of this paper.

REFERENCES

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: outlier-free 4-bit inference in rotated llms. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Raul Astudillo and Peter I. Frazier. Thinking inside the box: a tutorial on grey-box bayesian optimization. In *Proceedings of the Winter Simulation Conference*, WSC '21. IEEE Press, 2022.

Hao Mark Chen, Fuwen Tan, Alexandros Kouris, Royson Lee, Hongxiang Fan, and Stylianos I. Venieris. Progressive mixed-precision decoding for efficient llm inference. In *International Conference on Learning Representations*, 2025.

Zihan Chen, Bike Xie, Jundong Li, and Cong Shen. Channel-wise mixed-precision quantization for large language models. *Transactions on Machine Learning Research*, 2026.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Stosic Dusan, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, Levi Melnick, Maral Mesmakhosroshahi, Andres Rodriguez, Michael Schulte, Rasoul Shafipour, Lei Shao, Michael Siu, Pradeep Dubey, Paulius Micikevicius, Maxim Naumov, Colin Verrilli, Ralph Wittig, Doug Burger, and Eric Chung. Microscaling data formats for deep learning. October 2023. URL https://www.microsoft.com/en-us/research/publication/microscaling-data-formats-for-deep-learning/.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 30318–30332. Curran Associates, Inc., 2022. doi: 10.52202/068431-2198. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/c3ba4962c05c49636d4c6206a97e9c8a-Paper-Conference.pdf.

Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael Mahoney, and Kurt Keutzer. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 18518–18529. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/d77c703536718b95308130ff2e5cf9ee-Paper.pdf.

Haojie Duanmu, Xiuhong Li, Zhihang Yuan, Size Zheng, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. MxMoE: Mixed-precision quantization for MoE with accuracy and performance co-design. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International*

Conference on Machine Learning, volume 267 of *Proceedings of Machine Learning Research*, pp. 14793–14806. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/duanmu25a.html.

Vage Egiazarian, Roberto L. Castro, Denis Kuznedelev, Andrei Panferov, Eldar Kurtic, Shubhra Pandit, Alexandre Noll Marques, Mark Kurtz, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Bridging the gap between promise and performance for microscaling fp4 quantization. *CoRR*, abs/2509.23202, September 2025. URL https://doi.org/10.48550/arXiv.2509.23202.

Weilun Feng, Chuanguang Yang, Haotong Qin, Yuqi Li, Xiangqi Li, Zhulin An, Libo Huang, Boyu Diao, Fuzhen Zhuang, Michele Magno, Yongjun Xu, Yingli Tian, and Tingwen Huang. Mpq-dmv2: Flexible residual mixed precision quantization for low-bit diffusion models with temporal distillation, 2025.

Aaron Grattafiori et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Coleman Hooper, Charbel Sakr, Ben Keller, Rangharajan Venkatesan, Kurt Keutzer, Sophia Shao, and Brucek Khailany. Fgmp: Fine-grained mixed-precision weight and activation quantization for hardware-accelerated llm inference, 2025.

Wei Huang, Haotong Qin, Yangdong Liu, Yawei Li, Qinshuo Liu, Xianglong Liu, Luca Benini, Michele Magno, Shiming Zhang, and Xiaojuan Qi. SliM-LLM: Salience-driven mixed-precision quantization for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 25672–25692. PMLR, 2025.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818, June 2024.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Basile Lewandowski, Simon Kurz, Aditya Shankar, Robert Birke, Jian-Jia Chen, and Lydia Y. Chen. Tmpdiff: Temporal mixed-precision for diffusion models, 2026.

Muyang Li, Yujun Lin, Zhekai Zhang*, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. Svdquant: Absorbing outliers by low-rank components for 4-bit diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Xing Li, Zeyu Xing, Yiming Li, Linping Qu, Hui-Ling Zhen, Yiwu Yao, Wulong Liu, Sinno Jialin Pan, and Mingxuan Yuan. KVTuner: Sensitivity-aware layer-wise mixed-precision KV cache quantization for efficient and nearly lossless LLM inference. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 36451–36485. PMLR, 13–19 Jul 2025b. URL https://proceedings.mlr.press/v267/li25dd.html.

Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. In *Proceedings of Machine Learning and Systems*, volume 7, 2025.

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. Qllm: Accurate and efficient low-bitwidth quantization for large language models. In *ICLR*, 2024. URL https://openreview.net/forum?id=FIplmUWdm3.

Wenyuan Liu, Haoqian Meng, Yilun Luo, Peng Zhang, and Xindian Ma. Micromix: Efficient mixed-precision quantization with microscaling formats for large language models. In *International Conference on Learning Representations*, 2026.

Yuanyong Luo, Jing Huang, Yu Cheng, Ziwei Yu, Kaihua Zhang, Kehong Hong, Xinda Ma, Xin Wang, Anping Tong, Guipeng Hu, et al. Hifloat4 format for language model inference. *arXiv preprint arXiv:2602.11287*, 2026.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

OpenAI. Sora: Creating video from text. https://openai.com/index/sora/, 2024. Accessed: 2026-09-01.

Taig Singh, Shreshth Rajan, and Nikhil Jain. Arithmetic-intensity-aware quantization. *arXiv preprint arXiv:2512.14090*, 2025.

Rundong Su, Jintao Zhang, Zhihang Yuan, Haojie Duanmu, Jianfei Chen, and Jun Zhu. 6bit-diffusion: Inference-time mixed-precision quantization for video diffusion models, 2026.

Wei Tao, Haocheng Lu, Xiaoyang Qu, Bin Zhang, Kai Lu, Jiguang Wan, and Jianzong Wang. Moqae: Mixed-precision quantization for long-context llm inference via mixture of quantization-aware experts. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 10810–10820. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.531.

Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

Wild-Heart. Disney-videogeneration-dataset. Hugging Face Datasets, 2024. URL https://huggingface.co/datasets/Wild-Heart/Disney-VideoGeneration-Dataset.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Zhanhao Xie, Yuexiao Ma, Xiawu Zheng, Fei Chao, Wanchen Sui, Yong Li, Shen Li, and Rongrong Ji. Automated fine-grained mixture-of-experts quantization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 27024–27037. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.1386.

Yuhao Xu, Yantai Yang, Zhenyang Fan, Yufan Liu, Yuming Li, Bing Li, and Zhipeng Zhang. Qvla: Not all channels are equal in vision-language-action model's quantization. In *International Conference on Learning Representations*, 2026.

Huanrui Yang, Lin Duan, Yiran Chen, and Hai Li. Bsq: Exploring bit-level sparsity for mixed-precision neural network quantization. *arXiv preprint arXiv:2102.10462*, 2021.

Tianchen Zhao, Tongcheng Fang, Enshu Liu, Wan Rui, Widyadewi Soedarmadji, Shiyao Li, Zinan Lin, Guohao Dai, Shengen Yan, Huazhong Yang, and Xuefei Ning. Vidit-q: Efficient and accurate quantization of diffusion transformers for image and video generation. In *ICLR 2025*, February 2025a.

Tianchen Zhao, Ke Hong, Xinhao Yang, Xuefeng Xiao, Huixia Li, Feng Ling, Ruiqi Xie, SiQi Chen, Hongyu Zhu, Zhang Yichong, and Yu Wang. PAROAttention: Pattern-aware reordering for efficient sparse and quantized attention in visual generation models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=UPELg2oUo3.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate llm serving. In P. Gibbons, G. Pekhimenko, and C. De Sa (eds.), *Proceedings of Machine Learning and Systems*, volume 6, pp. 196–209, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/file/5edb57c05c81d04beb716ef1d542fe9e-Paper-Conference.pdf.

Zhen Zheng, Xiaonan Song, and Chuanjie Liu. Mixllm: Llm quantization with global mixed-precision between output-features and highly-efficient system design. In *Proceedings of Machine Learning and Systems*, volume 8, 2026.

APPENDIX

A CHAMP PROPERTIES

A.1 BUDGET PRESERVATION OF CAP-AND-REDISTRIBUTE.

Let $\mathcal{S}^\star$ denote the final set of capped modules and $\lambda^\star$ the scaling factor at convergence. The base ratios produced by the Cap-and-Redistribute procedure preserve the requested global MXFP8 budget:

$$\sum_{m=1}^{M} r_m C_m = r \sum_{m=1}^{M} C_m.$$

At convergence, capped modules are assigned $r_{\max}$, while uncapped modules are assigned $\lambda^\star a_m$. Therefore,

$$\sum_{m=1}^{M} r_m C_m = r_{\max} \sum_{m \in \mathcal{S}^\star} C_m + \lambda^\star \sum_{m \notin \mathcal{S}^\star} a_m C_m.$$

From Eq. 6, evaluated at convergence,

$$\lambda^\star = \frac{r \sum_{m=1}^{M} C_m - r_{\max} \sum_{m \in \mathcal{S}^\star} C_m}{\sum_{m \notin \mathcal{S}^\star} a_m C_m}.$$

Thus,

$$\lambda^\star \sum_{m \notin \mathcal{S}^\star} a_m C_m = r \sum_{m=1}^{M} C_m - r_{\max} \sum_{m \in \mathcal{S}^\star} C_m.$$

Substituting this expression into the total allocated budget gives

$$\sum_{m=1}^{M} r_m C_m = r_{\max} \sum_{m \in \mathcal{S}^\star} C_m + r \sum_{m=1}^{M} C_m - r_{\max} \sum_{m \in \mathcal{S}^\star} C_m = r \sum_{m=1}^{M} C_m.$$

Hence, Cap-and-Redistribute changes how the MXFP8 budget is distributed across modules without changing the total budget at the ratio level.

A.2 HIERARCHICAL BUDGET PRESERVATION.

The layer-level adjustment similarly redistributes each module's base ratio across Transformer layers without changing its average allocation. The adjustment factors are normalized such that

$$\frac{1}{L} \sum_{l=1}^{L} b_m^l = 1.$$

Since $r_m^l = b_m^l r_m$,

$$\frac{1}{L} \sum_{l=1}^{L} r_m^l = \frac{r_m}{L} \sum_{l=1}^{L} b_m^l = r_m.$$

The subsequent Cap-and-Redistribute step with $r_{\max} = 1$ is applied separately across the Transformer layers of each module and preserves this module-level budget. Consequently,

$$\sum_{l=1}^{L} r_m^l = L r_m.$$

Summing over all modules gives

$$\sum_{m=1}^{M} \sum_{l=1}^{L} r_m^l C_m = L \sum_{m=1}^{M} r_m C_m = Lr \sum_{m=1}^{M} C_m.$$

Thus, the two-level allocation redistributes the MXFP8 budget first across modules and then across Transformer layers while preserving the global ratio at the continuous-ratio level.

A.3 Special Cases.

The hierarchical allocation reduces to simpler policies under specific hyperparameter choices. When $\gamma_s = \gamma_c = 0$, the module-level allocation scores satisfy $a_m = 1$ for all modules. If the ratio cap is inactive, this gives $r_m = r$ for all m, recovering a uniform module-level allocation. Similarly, when $\gamma_b = 0$, the layer adjustment factors satisfy $b_m^l = 1$, giving $r_m^l = r_m$ for all Transformer layers and removing layer-level redistribution.

A.4 Quantization Sensitivity and Allocation Patterns

Figures 3–5 visualize the calibrated channel-group sensitivities and the resulting MXFP8 allocations identified by CHAMP at r=0.2. All allocations use the model-specific hyperparameters selected as described in Section B.2. The visualizations highlight substantial heterogeneity both across modules and across Transformer layers. Different modules receive markedly different base ratios, while the layer-level adjustment further redistributes each module's allocation according to its sensitivity pattern. These observations motivate the hierarchical allocation used by CHAMP rather than applying a uniform mixed-precision ratio across the model.

B Experimental Details

B.1 Text-to-Video Evaluation Prompts

All qualitative and quantitative video-generation results for Wan 2.2-T2V-A14B and HunyuanVideo use the eight prompts listed in Table 5, which are adapted from the OpenAI Sora demonstration videos OpenAI (2024).

B.2 Hyperparameter Selection.

To avoid expensive end-to-end evaluation during hyperparameter tuning, we select γ_s, γ_c, and $r_{\max}$ using a lightweight proxy computed from the precomputed calibration sensitivities. For every candidate configuration, we first obtain the module-level ratios $\{r_m\}_{m=1}^{M}$ using the allocation and Cap-and-Redistribute procedure described in Section 4.2. Thus, the proxy search uses exactly the same module-level allocation as CHAMP and does not require a separate allocation formulation.

Sensitivity coverage. For each module m, we rank all channel-groups across Transformer layers by their sensitivity scores $s_{m,c}^l$. We define $\mathrm{cov}_m(r_m)$ as the fraction of the module's total calibration sensitivity captured by the top r_m fraction of these channel-groups. We then minimize the sensitivity-weighted uncovered fraction,

$$\mathcal{L}_{\text{proxy}} = \sum_{m=1}^{M} \frac{C_m s_m}{\sum_{m'=1}^{M} C_{m'} s_{m'}} \left(1 - \mathrm{cov}_m(r_m)\right),$$

where $C_m s_m$ is proportional to the total calibration sensitivity of module m, since s_m denotes its mean sensitivity across Transformer layers and channel-groups. Minimizing $\mathcal{L}_{\text{proxy}}$ favors candidate

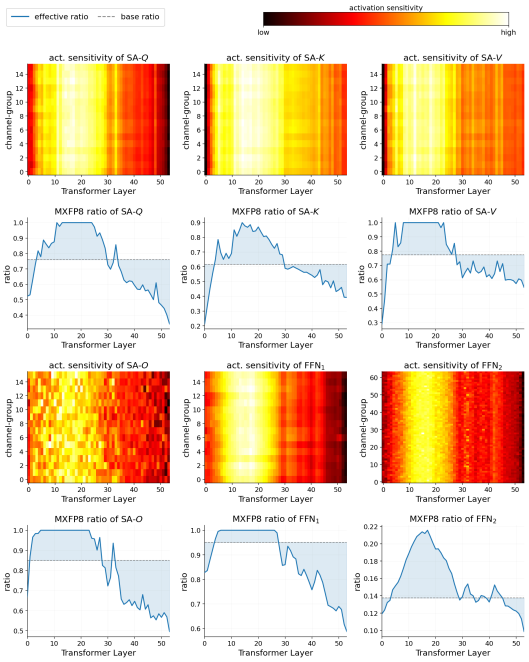


Figure 3: Sensitivity patterns and MXFP8 ratios identified by CHAMP for HunyuanVideo at $r{=}0.2$. Because the sparse-attention configuration produced low-quality videos in our setup, mixed precision is not applied to the attention Q, K, and V input tensors.

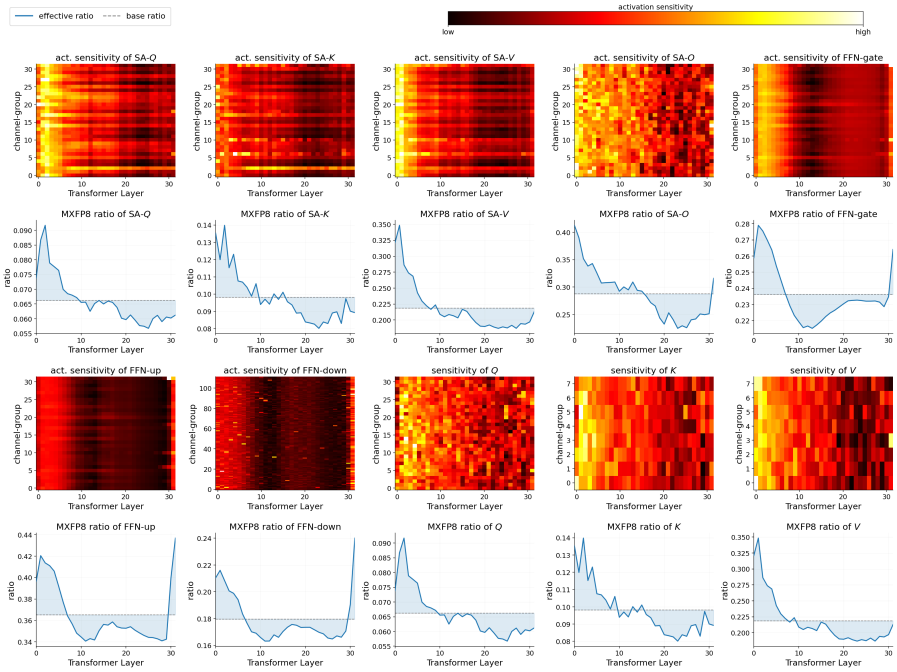


Figure 4: Sensitivity patterns and MXFP8 ratios identified by CHAMP for Llama-3.1-8B at $r{=}0.2$.

module-level allocations whose assigned MXFP8 ratios capture a larger fraction of the calibration sensitivity within each module.

Grid search. For each model, we search $\gamma_s{\in}[0, 2]$ with step 0.05, $\gamma_c{\in}[0, 1]$ with step 0.05, and $r_{\max}{\in}[0.1, 1]$ with step 0.05, for a total of $16{,}359$ configurations. The proxy is evaluated entirely

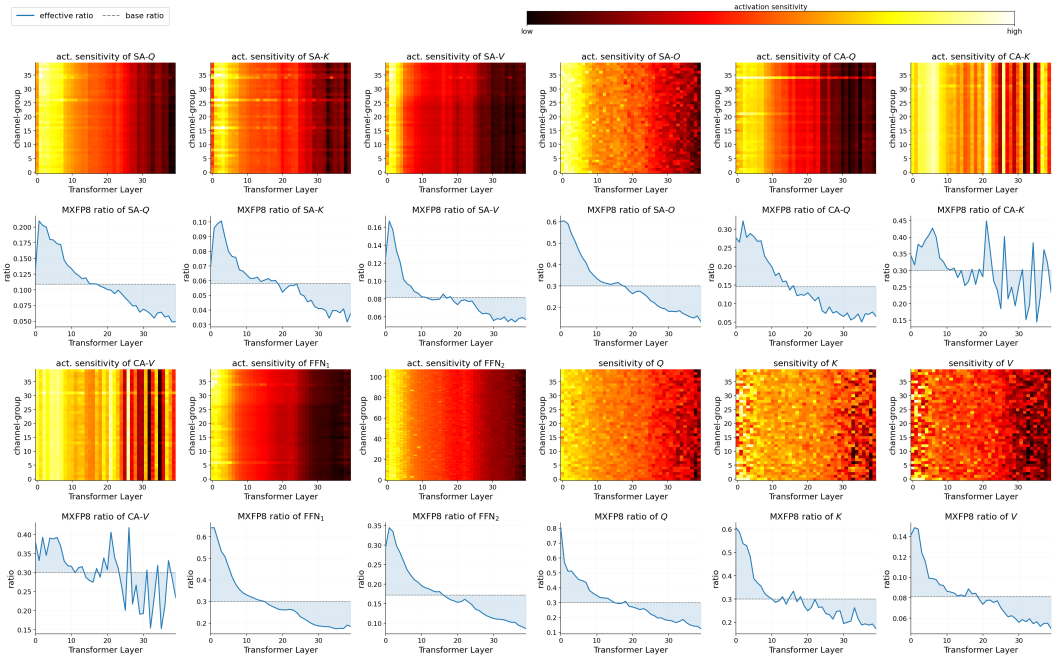


Figure 5: Sensitivity patterns and MXFP8 ratios identified by CHAMP for Wan 2.2-T2V-A14B at r=0.2.

from the precomputed calibration scores and therefore does not require downstream language or video generation for each configuration. The layer-level exponent is fixed to γ_b=0.15, which performed consistently well in our preliminary experiments. We exclude γ_b from the proxy search to limit its dimensionality, since jointly tuning additional hyperparameters can make the selected configuration more sensitive to imperfections in the proxy objective. The local–global channel selection uses a fixed 50/50 split in all experiments. The selected hyperparameters are fixed for all subsequent experiments and are reported in Table 6.

Limitations. The proxy measures calibration-sensitivity coverage rather than downstream quality directly. It therefore does not capture all interactions between quantization decisions across Transformer layers or their effect on the final model outputs. We use it as an efficient, model-agnostic criterion for hyperparameter selection and evaluate the selected configurations using the downstream metrics reported in the main paper.

B.3 Hyperparameter Sensitivity of CHAMP

We analyze the sensitivity of CHAMP to three module-allocation hyperparameters: the sensitivity exponent γ_s, size exponent γ_c, and allocation cap $r_{\max}$. We perform one-parameter-at-a-time sweeps on Llama-3.1-8B at r=0.20, fixing the other parameters at their selected values (γ_s=1.30, γ_c=0.00, $r_{\max}$=0.40; marked $\star$ in Table 7). We report the seven-task zero-shot average over ARC-Challenge, ARC-Easy, BoolQ, OpenBookQA, PIQA, RACE, and WinoGrande.

Accuracy varies moderately with γ_s, ranging from 62.61% to 63.16%, without a clear monotonic trend. The γ_c sweep shows a narrow range (62.86–63.38%), with the best result at γ_c=0.30. Performance is therefore relatively insensitive to either exponent over this range.

The allocation cap $r_{\max}$ has a larger effect. A tighter cap of $r_{\max}$=0.20 achieves 63.85%, compared with 62.89% at the selected value of 0.30. Performance remains unchanged for $r_{\max}\geq$0.30 in this sweep, indicating that larger caps have little effect on the resulting allocation in this setting. The improvement from tighter caps suggests that preventing individual modules from receiving overly large FP8 allocations can improve the distribution of a limited high-precision budget. CHAMP is relatively insensitive to γ_s and γ_c. But, $r_{\max}$ has a more pronounced effect and benefits from tuning.

Table 5: The eight evaluation prompts used for all text-to-video generation experiments (Wan 2.2-T2V-A14B and HunyuanVideo). Prompts are sourced from the OpenAI Sora demonstration videos OpenAI (2024).

ID	Prompt
0	A stylish woman walks down a Tokyo street filled with warm glowing neon and animated city signage. She wears a black leather jacket, a long red dress, and black boots, and carries a black purse. She wears sunglasses and red lipstick. She walks confidently and casually. The street is damp and reflective, creating a mirror effect of the colorful lights. Many pedestrians walk about.
1	Several giant wooly mammoths approach treading through a snowy meadow, their long wooly fur lightly blows in the wind as they walk, snow covered trees and dramatic snow capped mountains in the distance, mid afternoon light with wispy clouds and a sun high in the distance creates a warm glow, the low camera view is stunning capturing the large furry mammal with beautiful photography, depth of field, cinematic, 4k.
2	A movie trailer featuring the adventures of the 30 year old space man wearing a red wool knitted motorcycle helmet, blue sky, salt desert, cinematic style, shot on 35mm film, vivid colors.
3	Drone view of waves crashing against the rugged cliffs along Big Sur's garay point beach. The crashing blue waters create white-tipped waves, while the golden light of the setting sun illuminates the rocky shore. A small island with a lighthouse sits in the distance, and green shrubbery covers the cliff's edge. The steep drop from the road to the beach is a dramatic feat, with the cliff's edges jutting out over the sea. This is a view that captures the raw beauty of the coast and the rugged landscape of the Pacific Coast Highway.
4	Animated scene features a close-up of a short fluffy monster kneeling beside a melting red candle. The art style is 3D and realistic, with a focus on lighting and texture. The mood of the painting is one of wonder and curiosity, as the monster gazes at the flame with wide eyes and open mouth. Its pose and expression convey a sense of innocence and playfulness, as if it is exploring the world around it for the first time. The use of warm colors and dramatic lighting further enhances the cozy atmosphere of the image.
5	A gorgeously rendered papercraft world of a coral reef, rife with colorful fish and sea creatures.
6	This close-up shot of a Victoria crowned pigeon showcases its striking blue plumage and red chest. Its crest is made of delicate, lacy feathers, while its eye is a striking red color. The bird's head is tilted slightly to the side, giving the impression of it looking regal and majestic. The background is blurred, drawing attention to the bird's striking appearance.
7	Photorealistic closeup video of two pirate ships battling each other as they sail through a stormy sea. Huge waves lash the ships and we see thunderous bolts of lightning in the background as the two ships fire cannons at each other.

Table 6: Hyperparameters used in our experiments. γ_s, γ_c, and $r_{\max}$ are selected using the calibration-based proxy, while γ_b is fixed to 0.15.

Model	γ_s	γ_c	γ_b	$r_{\max}$
Wan 2.2-T2V-A14B	0.90	0.15	0.15	0.30
HunyuanVideo	0.40	0.15	0.15	0.85
Llama-3.1-8B	1.30	0.00	0.15	0.40

B.4 Implementation Details

We implement CHAMP in PyTorch and integrate it into the official inference pipeline of each evaluated model. MXFP8 and HiF4 are simulated using fake quantization: tensors are quantized to the target numerical format and dequantized before the corresponding computation. We use a fixed random seed of 997 throughout all experiments.

Video denoising precision. Following prior work that identifies early denoising timesteps as particularly quantization-sensitive and retains them at higher precision Zhao et al. (2025a), we keep the first three and final three denoising steps in BF16 for all mixed-precision methods and apply mixed precision only to the remaining steps.

Table 7: One-parameter-at-a-time hyperparameter sensitivity on Llama-3.1-8B at r=0.20. Each column varies one hyperparameter while fixing the others at their selected values ($\star$). Bold indicates the best accuracy within each sweep. FGMP achieves 62.75%, while full MXFP8 achieves 64.65%.

γ_s (γ_c=0.00, $r_{\max}$=0.40)		γ_c (γ_s=1.30, $r_{\max}$=0.40)		$r_{\max}$ (γ_s=1.30, γ_c=0.00)	
Value	Acc. (%)	Value	Acc. (%)	Value	Acc. (%)
0.3	63.16	0.00*	63.19	**0.20**	**63.85**
0.5	62.61	0.15	63.14	0.25	63.08
0.7	62.76	**0.30**	**63.38**	0.30	62.89
0.9	62.89	0.50	62.94	0.35	62.89
1.1	63.09	0.70	62.86	0.40*	63.19
1.3*	**63.19**	1.00	63.02	0.50	62.89
1.5	62.83				

Table 8: Per-module MXFP8 allocation before and after Diagnostic 1 at r=0.20 for Llama-3.1-8B. Counts are summed across all 32 Transformer layers, and the total MXFP8 budget is unchanged.

Module	Global (ch)	Oracle (ch)	Δ ch	Global ratio	Oracle ratio
SA-Q	84	85	$+1$	8.20%	8.30%
SA-K	103	104	$+1$	10.06%	10.16%
SA-V	165	166	$+1$	16.11%	16.21%
SA-O	280	279	-1	27.34%	27.25%
FFN-gate	253	254	$+1$	24.71%	24.80%
FFN-up	409	407	-2	39.94%	39.74%
FFN-down	652	651	-1	18.19%	18.16%

B.5 Oracle Diagnostics

Motivation. A flat global ranking (Global) of channel-group sensitivities selects the top-r fraction of channel-groups across all modules and Transformer layers. This unconstrained allocation can concentrate MXFP8 budget in a subset of module–layer pairs, leaving others entirely in HiF4. We refer to such pairs as *starved*. For Llama-3.1-8B at r=0.20, this method starves 95 of the 224 linear module–layer pairs (42.4%), including 25 layers of `q_proj`, 21 layers of `gate_proj`, and 20 layers of `v_proj`. We therefore run two post-hoc oracle diagnostics to test whether the flat global allocation can be improved under the same total MXFP8 budget: one redistributes budget across module types, and the other redistributes budget across Transformer layers within each module.

Diagnostic 1: Cross-module budget reallocation. Starting from Global, we perform a greedy search for beneficial MXFP8 budget swaps across modules. For each target module, we consider promoting one of its K=5 most-sensitive HiF4 channel-groups to MXFP8 and demoting one of the K=5 least-sensitive MXFP8 channel-groups from another module. We evaluate the candidate swaps using WikiText-2 perplexity and accept the best improving swap. This process is repeated until no further improvement is found or five passes are completed.

As shown in Table 8, starting from Global with perplexity of 8.4245, the greedy search converges after two passes and four accepted swaps, reaching perplexity of 8.3814 ($\Delta = -0.0430$). The results show that even a small amount of module-level budget swaps improves perplexity, indicating that the module-level allocation produced by Global is not optimal.

Diagnostic 2: Within-module layer reallocation. Starting again from Global, we examine whether starved module–layer pairs can benefit from receiving a small MXFP8 allocation from another layer of the same module. Global ranks channel-groups individually across the model, so a layer may contain several moderately sensitive groups without any one of them ranking high enough globally to receive MXFP8. Since quantization sensitivity can vary across Transformer layers Li et al. (2025b), we test whether such starved pairs can be improved while keeping both the module-level and total MXFP8 budgets fixed.

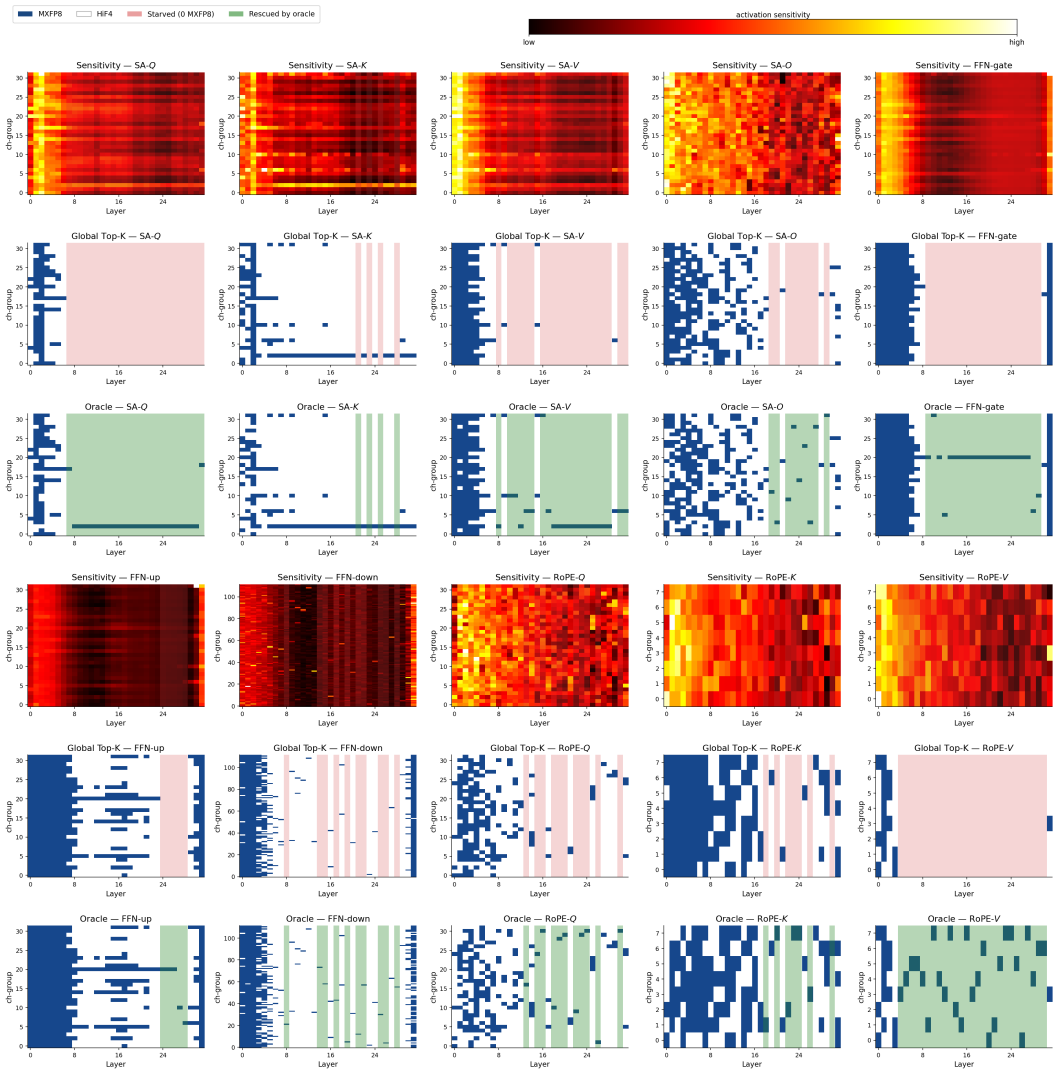


Figure 6: Activation sensitivity and precision-allocation masks for Llama-3.1-8B at r=0.20. Rows show sensitivity, Global allocation, and same-budget within-module reallocation. Blue/white denote MXFP8/HiF4; pink and green mark starved and rescued module–layer pairs, respectively.

For each starved target layer, we consider all layers within the same module that have more than one MXFP8 channel-group as candidate donors. We demote the lowest-sensitivity MXFP8 channel-group in the donor layer to HiF4 and promote the highest-sensitivity HiF4 channel-group in the target layer to MXFP8. We evaluate perplexity on WikiText-2 for each candidate donor and accept the swap with the largest perplexity reduction, if any improvement exists.

Figure 6 shows that some starved module–layer pairs can improve perplexity after receiving MXFP8 budget from another layer of the same module. Such single-group reallocations can reduce perplexity, showing that the layer-level allocation produced by Global ranking is not always optimal. This provides a simple proof-of-concept that the layer-level distribution produced by Global can be improved without changing either the per-module or overall MXFP8 budget.

Limitations. These experiments are intended as diagnostics rather than practical allocation strategies. Diagnostic 1 restricts the search to K=5 sensitivity-ranked candidates and therefore does not evaluate all possible cross-module swaps. Diagnostic 2 considers only starved module–layer pairs and single-group reallocations between layers of the same module. Moreover, both diagnostics use WikiText-2 perplexity to evaluate candidate swaps, requiring repeated model-level evaluation.

Table 9: Zero-shot accuracy (%) of Llama-3.1-8B across different global MXFP8 ratios. BF16, MXFP8, and HiF4 are shown once as reference configurations. FGMP and CHAMP use the same high-precision budget at each ratio. Values in parentheses in the Avg. column denote the gap to full MXFP8. Bold indicates the best result among FGMP and CHAMP at each ratio. Higher is better.

Method	r	ARC-C	ARC-E	BoolQ	PIQA	Wino	OpenBQ	RACE	Avg.
BF16	–	55.20	82.70	83.12	80.79	74.43	45.20	39.90	65.91
MXFP8	1.00	53.92	81.19	82.02	80.41	71.67	43.80	39.52	64.65
HiF4	0.00	49.66	77.95	79.48	78.24	68.82	42.20	38.85	62.17
FGMP	0.10	50.51	77.69	**80.43**	77.26	70.88	41.80	**39.04**	62.52 (−2.13)
CHAMP	0.10	**51.96**	**79.08**	80.24	**79.11**	**72.06**	**44.20**	38.47	**63.59** (−1.06)
FGMP	0.15	50.77	77.69	**80.58**	77.37	70.17	**44.00**	38.47	62.72 (−1.93)
CHAMP	0.15	**51.96**	**78.03**	79.20	**78.56**	**70.88**	43.80	**38.56**	**63.00** (−1.65)
FGMP	0.20	52.56	77.86	**80.31**	77.91	69.69	42.80	38.09	62.75 (−1.90)
CHAMP	0.20	**53.24**	**78.83**	79.02	**78.07**	**70.88**	**43.40**	**38.85**	**63.19** (−1.46)
FGMP	0.25	52.05	77.86	**80.55**	78.29	**71.67**	**43.80**	**39.14**	63.34 (−1.31)
CHAMP	0.25	**53.07**	**78.58**	79.05	**79.22**	71.35	43.60	39.04	**63.42** (−1.23)
FGMP	0.30	**52.13**	78.87	**80.31**	78.24	**70.56**	43.20	38.66	**63.14** (−1.51)
CHAMP	0.30	51.79	**79.71**	78.78	**78.78**	69.69	**44.00**	**39.04**	63.11 (−1.54)
FGMP	0.35	**52.65**	78.70	**80.52**	78.02	**71.98**	42.60	**39.62**	63.44 (−1.21)
CHAMP	0.35	52.56	**79.42**	79.54	**79.27**	71.03	**44.20**	39.33	**63.62** (−1.03)
FGMP	0.40	52.65	78.20	**81.41**	78.89	70.48	**44.80**	**39.90**	**63.76** (−0.89)
CHAMP	0.40	**53.50**	**79.29**	79.48	**79.76**	**71.27**	**44.80**	37.70	63.69 (−0.96)
FGMP	0.45	**53.33**	78.66	**80.86**	78.35	**72.22**	**44.20**	**38.76**	**63.77** (−0.88)
CHAMP	0.45	53.24	**79.88**	80.40	**80.09**	70.88	43.00	38.37	63.69 (−0.96)
FGMP	0.50	**53.24**	79.00	**80.70**	79.43	**71.98**	**45.60**	**39.14**	**64.16** (−0.49)
CHAMP	0.50	52.47	**79.92**	80.46	**79.65**	70.40	43.60	38.95	63.64 (−1.01)

Implications for CHAMP. The two diagnostics highlight complementary limitations of Global ranking. Diagnostic 1 shows that its module-level budget distribution can be improved through cross-module redistribution, and Diagnostic 2 shows that some module–layer pairs left entirely in HiF4 can benefit from within-module layer reallocation. CHAMP addresses these issues using calibration-time sensitivity statistics to control budget across modules and Transformer layers and to construct the channel-group precision mask. Unlike the diagnostics, it does not require repeated model-level evaluation during allocation.

C Ablation Studies

C.1 Effect of the MXFP8 Ratio

C.1.1 Large Language Models

We evaluate Llama-3.1-8B over MXFP8 ratios r=0.10–0.50 (Table 9). CHAMP consistently outperforms FGMP for $r \leq 0.25$, with the largest gain at r=0.10 (62.52%→63.59%). This indicates that hierarchical allocation is most beneficial when high-precision budget is limited. At larger ratios, differences narrow and performance is mixed, as MXFP8 placement becomes less critical.

C.1.2 Video Diffusion Models

We further evaluate Wan 2.2-T2V-A14B across r=0.10–0.50 and HunyuanVideo across r=0.40–0.80. Table 10 shows that the frame-level fidelity advantages of CHAMP persist across a broad range of ratios. On HunyuanVideo, CHAMP improves cosine similarity, SSIM, PSNR, and MSE over FGMP at every evaluated ratio. On Wan, CHAMP consistently achieves higher cosine similarity and lower MSE, while SSIM and PSNR are mixed at some intermediate ratios.

Table 10: Frame-level fidelity relative to BF16 across the full ratio sweep (Wan 2.2-T2V-A14B: r=0.10–0.50; HunyuanVideo: r=0.40–0.80). Values in parentheses denote the gap to MXFP8. Bold indicates the best result among FGMP and CHAMP at each r.

Model	r	Method	Cosine ↑	SSIM ↑	PSNR (dB) ↑	MSE ↓
Wan 2.2-T2V-A14B	–	MXFP8	97.50	73.27	20.71	794
	–	HiF4	94.13	57.27	16.70	1712
	0.10	FGMP	96.01 (−1.49)	64.62 (−8.65)	19.21 (−1.50)	1275 (−481)
		CHAMP	**96.38** (−1.12)	**66.07** (−7.20)	**19.51** (−1.20)	**1139** (−345)
	0.15	FGMP	96.04 (−1.46)	65.31 (−7.96)	19.29 (−1.42)	1284 (−490)
		CHAMP	**96.40** (−1.10)	**65.63** (−7.64)	**19.32** (−1.39)	**1141** (−347)
	0.20	FGMP	96.26 (−1.24)	66.56 (−6.71)	19.53 (−1.18)	1197 (−403)
		CHAMP	**96.89** (−0.61)	**67.72** (−5.55)	**19.67** (−1.04)	**999** (−205)
	0.25	FGMP	96.46 (−1.04)	**67.50** (−5.77)	**19.84** (−0.87)	1136 (−342)
		CHAMP	**96.72** (−0.78)	67.00 (−6.27)	19.41 (−1.30)	**1030** (−236)
	0.30	FGMP	96.21 (−1.29)	**66.10** (−7.17)	**19.40** (−1.31)	1218 (−424)
		CHAMP	**96.42** (−1.08)	65.75 (−7.52)	19.26 (−1.45)	**1110** (−316)
	0.35	FGMP	96.29 (−1.21)	**67.29** (−5.98)	**19.68** (−1.03)	1195 (−401)
		CHAMP	**96.73** (−0.77)	66.97 (−6.30)	19.53 (−1.18)	**1027** (−233)
	0.40	FGMP	96.10 (−1.40)	66.70 (−6.57)	19.57 (−1.14)	1259 (−465)
		CHAMP	**96.82** (−0.68)	**68.08** (−5.19)	**19.81** (−0.90)	**1011** (−217)
	0.45	FGMP	96.55 (−0.95)	68.92 (−4.35)	20.15 (−0.56)	1078 (−284)
		CHAMP	**96.84** (−0.66)	**69.06** (−4.21)	**20.34** (−0.37)	**1009** (−215)
	0.50	FGMP	96.76 (−0.74)	69.28 (−3.99)	20.06 (−0.65)	1010 (−216)
		CHAMP	**97.01** (−0.49)	**70.10** (−3.17)	**20.62** (−0.09)	**936** (−142)
HunyuanVideo	–	MXFP8	95.75	54.07	15.83	2029
	–	HiF4	92.81	48.16	13.01	3474
	0.40	FGMP	94.55 (−1.20)	50.42 (−3.65)	14.29 (−1.54)	2649 (−620)
		CHAMP	**95.00** (−0.75)	**52.27** (−1.80)	**14.78** (−1.05)	**2382** (−353)
	0.45	FGMP	94.76 (−0.99)	51.18 (−2.89)	14.53 (−1.30)	2532 (−503)
		CHAMP	**95.19** (−0.56)	**52.08** (−1.99)	**14.95** (−0.88)	**2287** (−258)
	0.50	FGMP	94.83 (−0.92)	51.32 (−2.75)	14.57 (−1.26)	2499 (−470)
		CHAMP	**95.42** (−0.33)	**52.98** (−1.09)	**15.24** (−0.59)	**2200** (−171)
	0.55	FGMP	94.86 (−0.89)	51.32 (−2.75)	14.59 (−1.24)	2506 (−477)
		CHAMP	**95.39** (−0.36)	**52.91** (−1.16)	**15.08** (−0.75)	**2256** (−227)
	0.60	FGMP	95.08 (−0.67)	51.58 (−2.49)	14.75 (−1.08)	2401 (−372)
		CHAMP	**95.57** (−0.18)	**53.47** (−0.60)	**15.31** (−0.52)	**2160** (−131)
	0.65	FGMP	95.18 (−0.57)	52.06 (−2.01)	14.85 (−0.98)	2350 (−321)
		CHAMP	**95.58** (−0.17)	**53.58** (−0.49)	**15.45** (−0.38)	**2147** (−118)
	0.70	FGMP	95.29 (−0.46)	52.38 (−1.69)	14.97 (−0.86)	2295 (−266)
		CHAMP	**95.65** (−0.10)	**53.78** (−0.29)	**15.42** (−0.41)	**2126** (−97)
	0.75	FGMP	95.37 (−0.38)	52.83 (−1.24)	15.08 (−0.75)	2243 (−214)
		CHAMP	**95.72** (−0.03)	**54.15** (+0.08)	**15.55** (−0.28)	**2092** (−63)
	0.80	FGMP	95.43 (−0.32)	53.02 (−1.05)	15.22 (−0.61)	2215 (−186)
		CHAMP	**95.69** (−0.06)	**54.01** (−0.06)	**15.58** (−0.25)	**2102** (−73)

Table 11 shows that these fidelity gains are obtained while maintaining comparable generation quality. For HunyuanVideo, CHAMP achieves a higher VBench average than FGMP at every ratio through r=0.75, with only a small difference at r=0.80. For Wan, the average VBench scores of the two methods remain within 0.18 points throughout the sweep. Unlike the LLM results, where the largest gains occur at low MXFP8 budgets, the video results therefore show persistent fidelity benefits across a broader range of ratios.

Table 11: VBench scores ($\times 100$) for text-to-video generation across the full ratio sweep (Wan 2.2-T2V-A14B: r=0.10–0.50; HunyuanVideo: r=0.40–0.80). OC/SC/BC: overall/subject/background consistency; AQ/IQ: aesthetic/imaging quality; MS: motion smoothness. Parentheses in Avg. denote the gap to MXFP8. Bold indicates the best result among FGMP and CHAMP at each r.

Model	r	Method	OC ↑	SC ↑	BC ↑	AQ ↑	IQ ↑	MS ↑	Avg. ↑
Wan 2.2-T2V-A14B	–	BF16	28.88	97.88	98.27	69.24	73.32	99.03	77.77
	–	MXFP8	29.26	97.99	98.40	69.74	73.02	99.00	77.90
	–	HiF4	28.66	97.44	97.69	67.63	71.36	99.00	76.96
	0.10	FGMP	**29.04**	**97.81**	**98.09**	**69.80**	72.10	98.99	**77.64** (−0.26)
		CHAMP	28.93	97.75	97.95	69.11	**72.39**	98.99	77.52 (−0.38)
	0.15	FGMP	28.65	**97.91**	**98.10**	69.61	72.17	**99.03**	77.58 (−0.33)
		CHAMP	**28.73**	97.79	97.99	**69.81**	**72.53**	98.98	**77.64** (−0.26)
	0.20	FGMP	**29.00**	**97.96**	98.13	70.15	**72.66**	**99.03**	**77.82** (−0.08)
		CHAMP	28.86	97.72	**98.21**	**70.43**	72.15	98.99	77.73 (−0.17)
	0.25	FGMP	28.74	**98.02**	**98.30**	69.65	**72.73**	**99.02**	**77.74** (−0.16)
		CHAMP	**28.93**	97.91	98.10	**70.08**	71.96	98.98	77.66 (−0.24)
	0.30	FGMP	**28.70**	**97.93**	**98.41**	69.45	**72.60**	**99.02**	**77.68** (−0.22)
		CHAMP	28.68	97.71	98.18	**69.64**	71.78	98.98	77.50 (−0.41)
	0.35	FGMP	28.76	**97.99**	**98.25**	69.34	**72.14**	**99.03**	77.59 (−0.32)
		CHAMP	**29.11**	97.67	98.07	**69.97**	71.75	98.99	**77.59** (−0.31)
	0.40	FGMP	28.65	**97.86**	**98.21**	69.70	72.00	**99.03**	77.57 (−0.33)
		CHAMP	**28.99**	97.70	98.08	**69.92**	**72.21**	99.00	**77.65** (−0.25)
	0.45	FGMP	**28.46**	97.89	**98.24**	70.12	72.43	99.02	**77.70** (−0.21)
		CHAMP	28.22	**97.99**	97.90	**70.18**	**72.71**	**99.03**	77.67 (−0.23)
	0.50	FGMP	28.59	97.84	**98.28**	69.74	72.43	99.01	77.65 (−0.26)
		CHAMP	**28.90**	**97.96**	98.09	**70.02**	**72.89**	**99.02**	**77.81** (−0.09)
HunyuanVideo	–	BF16	29.67	94.59	95.17	62.17	66.80	99.14	74.59
	–	MXFP8	29.69	94.60	95.35	63.15	69.45	99.17	75.24
	–	HiF4	20.33	93.36	95.16	50.38	48.33	99.26	67.80
	0.40	FGMP	27.57	93.59	95.08	**61.12**	63.02	99.14	73.25 (−1.98)
		CHAMP	**27.96**	**93.80**	**95.39**	60.78	**65.74**	**99.17**	**73.80** (−1.43)
	0.45	FGMP	27.98	93.85	**95.42**	**61.44**	63.58	**99.18**	73.57 (−1.66)
		CHAMP	**28.43**	**93.97**	95.13	60.27	**66.01**	99.14	**73.82** (−1.41)
	0.50	FGMP	28.09	93.72	95.44	**61.89**	65.87	**99.17**	74.03 (−1.21)
		CHAMP	**29.02**	**94.25**	**95.84**	61.43	**66.11**	99.13	**74.30** (−0.94)
	0.55	FGMP	28.57	**94.20**	95.42	**62.58**	65.32	**99.17**	74.21 (−1.03)
		CHAMP	**29.76**	93.86	**95.85**	62.47	**66.60**	99.14	**74.61** (−0.62)
	0.60	FGMP	28.56	**94.37**	**95.48**	62.26	65.27	**99.17**	74.18 (−1.05)
		CHAMP	**29.73**	93.75	95.34	**62.74**	**67.70**	99.13	**74.73** (−0.50)
	0.65	FGMP	28.92	**94.52**	**95.40**	**63.28**	65.57	**99.19**	74.48 (−0.76)
		CHAMP	**29.59**	93.91	95.34	63.07	**69.24**	99.16	**75.05** (−0.19)
	0.70	FGMP	28.93	**94.14**	95.69	**63.80**	66.88	**99.18**	74.77 (−0.47)
		CHAMP	**29.45**	93.99	**95.95**	63.21	**68.57**	99.16	**75.06** (−0.18)
	0.75	FGMP	28.98	**94.09**	**95.59**	**63.28**	67.13	**99.18**	74.71 (−0.52)
		CHAMP	**29.62**	94.03	95.51	62.85	**69.00**	99.15	**75.03** (−0.21)
	0.80	FGMP	29.42	**94.25**	**95.81**	**63.31**	68.37	**99.18**	**75.06** (−0.18)
		CHAMP	**29.72**	94.10	95.53	62.72	**68.85**	99.17	75.02 (−0.22)

C.2 CHAMP Component Ablation

CHAMP combines three main mechanisms: module-level allocation, layer-level adjustment and local/global blending. We perform a leave-one-component-out ablation on Llama-3.1-8B at r=0.10

Table 12: Leave-one-out ablation for Llama-3.1-8B at $r=0.10$. Each row removes one CHAMP component while holding the others fixed. Δ is relative to Full CHAMP.

Variant	Avg. Acc. (%)	Δ
Full CHAMP	**63.59**	—
w/o module allocation ($\gamma_s=0$, $\gamma_c=0$)	62.91	-0.68
w/o layer adjustment ($\gamma_b=0$)	63.04	-0.55
Local-only ($\alpha=1$)	63.32	-0.27
Global-only ($\alpha=0$)	63.09	-0.50

to assess the contribution of each mechanism. Each variant disables one component while keeping the remaining settings fixed. We report the 7-task zero-shot average accuracy on ARC-Challenge, ARC-Easy, BoolQ, OpenBookQA, PIQA, RACE, and WinoGrande. Results are shown in Table 12. According to the table, all three CHAMP components contribute positively at $r=0.10$.

Module-level allocation. Setting $\gamma_s=\gamma_c=0$ removes sensitivity- and size-aware redistribution across module types and reduces accuracy by 0.68 points, the largest drop among the ablations. This indicates that explicitly controlling the module-level MXFP8 budget is an important component of CHAMP.

Layer-level adjustment. Setting $\gamma_b=0$ removes sensitivity-based redistribution across Transformer layers, assigning each layer the module's base ratio. This reduces accuracy by 0.55 points, the second-largest drop, showing that layer-specific sensitivity provides useful information for distributing the MXFP8 budget within each module.

Local/global blending. Both extreme selection strategies underperform the default $\alpha=0.5$. Global-only selection ($\alpha=0$, $\Delta=-0.50$) allows the remaining budget to concentrate in the highest-sensitivity channel-groups across layers, whereas local-only selection ($\alpha=1$, $\Delta=-0.27$) restricts selection to each layer independently. The mixed strategy outperforms both, indicating that combining layer-local selection with cross-layer competition is beneficial.