Beyond Adjacent Layers: Graph-Guided Layer Fusion for Compressing Large Language Models

Muhammad Usama
Braindeck Inc
Seoul, Republic of Korea
usama@braindeck.net

Yeoreum Jung
Braindeck Inc
Seoul, Republic of Korea
summer@braindeck.net

Abstract

Existing depth-compression methods for large language models (LLMs) score layers individually or group only adjacent or contiguous layers, implicitly assuming that functional redundancy is local. We test this assumption by computing full pairwise Centered Kernel Alignment (CKA) similarity matrices for thirteen LLMs from six architecture families (1.6B–14B parameters). Non-adjacent layers turn out to have 3.6–15.9$\times$ more high-similarity pairs than adjacent ones, a structure that prior similarity-based layer merging has not systematically exploited. We propose Fusion via Community Mining (FCM), which builds the full $L \times L$ similarity graph, uses spectral community detection to find clusters of functionally equivalent layers, keeps the highest-Block-Influence layer in each cluster, and then applies LoRA recovery; a pre-specified rule computed only on calibration data decides whether to use cosine or CKA similarity. At 2$\times$ compression, FCM lowers mean perplexity below ShortGPT on twelve of thirteen models, and the gain is statistically significant on ten under a Welch's t-test with Holm–Bonferroni correction (up to 50% on Qwen3-4B, $p_{\text{corr}} < 0.001$). Two models are ties, and Mistral-7B is the only loss (+13.4%). Against three recent adjacent-pair methods (FlattenGPT, SWM, DeltaLLM) under matched calibration and recovery, FCM gives the best perplexity on ten of thirteen models (mean rank 1.15), and it also outperforms the similarity-based merging method MKA on three anchor models. The gains carry over to downstream tasks. FCM matches or exceeds ShortGPT on 15 of 20 (model, benchmark) cells of a five-benchmark zero-shot reasoning suite (ARC-Easy, ARC-Challenge, HellaSwag, WinoGrande, PIQA), and it also improves MMLU (5-shot), LAMBADA generative perplexity, HumanEval pass@1, and free-form generation quality. The 2$\times$-compressed Qwen2.5-7B has 46% fewer parameters, 42% lower peak VRAM, and 84% higher decode throughput on a single A100. Ablations show that spectral grouping outperforms Fisher contiguous segmentation by 41% and uniform partitioning by 51%, and a compression sweep reveals a 2$\times$ crossover beyond which FCM's advantage keeps growing up to 3$\times$ and persists at the 14B scale. Code and an interactive project page are available at `https://github.com/Usama1002/fcm-layer-fusion` and `https://usama1002.github.io/fcm`.

1 Introduction

Deploying large language models (LLMs) at inference time is bottlenecked by latency, memory footprint, and energy cost, all of which scale with depth because each transformer block is a sequential dependency on the critical path. Depth compression removes or merges entire transformer blocks and therefore reduces all three costs simultaneously without full retraining [Men et al., 2025, Gromov et al., 2025]. The approach is viable because modern LLMs contain substantial layer-level redundancy: Men et al. [2025] showed that middle layers can be removed with minimal performance loss, Gromov et al. [2025] demonstrated that up to half of the layers in some models are dispensable given

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

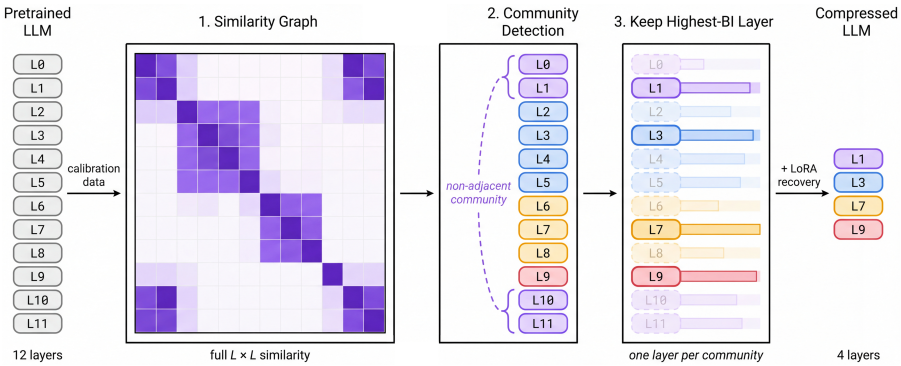


Figure 1: Overview of the FCM pipeline. Given a pretrained LLM with L layers, FCM (1) profiles pairwise activation similarity to construct a full $L \times L$ matrix $\mathbf{S}$, revealing block-diagonal structure with non-adjacent high-similarity regions; (2) applies spectral clustering to partition layers into communities that may span non-adjacent positions; and (3) selects the highest Block Influence layer per community, discarding the rest. The compressed model retains K layers (one per community) and undergoes LoRA recovery fine-tuning.

parameter-efficient recovery, and Sun et al. [2025] traced this redundancy to the variance explosion induced by Pre-Layer Normalization, which causes deeper layers to increasingly approximate identity transformations.

A growing body of methods exploits this redundancy, organised by how each decides which layers to remove or merge (Table 1). *Per-layer scoring*: ShortGPT [Men et al., 2025] ranks layers by Block Influence (BI) and removes the lowest scorers; Gromov et al. [Gromov et al., 2025] remove contiguous low-importance blocks healed by QLoRA fine-tuning; LLM-Streamline [Chen et al., 2025] adds a lightweight replacement network. *Per-layer replacement*: GRASP [Liu et al., 2025] replaces individual layers with adaptive singular parameters guided by gradient attribution. *Adjacent-pair operations*: FlattenGPT [Xu et al., 2026] concatenates and compresses adjacent block pairs, SWM [Ding et al., 2026] consolidates consecutive layers via parameter averaging, and DeltaLLM [Mikaelyan et al., 2025] shares weights between neighbours with low-rank corrections. *Contiguous segmentation*: SGLP [Li et al., 2024] applies Fisher Optimal Segmentation to a CKA similarity matrix and prunes within each contiguous segment using GradNorm. *Similarity-based merging*: MKA [Liu et al., 2024] merges similar layers via manifold alignment and an information-bottleneck criterion without an adjacency constraint, but selects merge groups greedily by pairwise similarity rather than from the global structure of the full graph and merges weights rather than retaining an intact representative per group. *Orthogonal compression families* that compose with depth compression: width pruning (SliceGPT [Ashkboos et al., 2024], SparseGPT [Frantar and Alistarh, 2023], Wanda [Sun et al., 2024]) and post-training quantization (GPTQ [Frantar et al., 2023], AWQ [Lin et al., 2024]). With the sole exception of MKA, these approaches either treat layers independently or restrict grouping to adjacent or contiguous positions, leaving the implicit assumption that functional redundancy is a local, sequential phenomenon unexamined. As a consequence, K layers retained by per-layer scoring or contiguous segmentation may concentrate in a few functional phases (e.g., the middle block) rather than span the whole forward computation.

We challenge this assumption. By computing full $L \times L$ pairwise CKA similarity matrices over all layer pairs in thirteen LLMs from six architecture families and parameter scales from 1.6B to 14B, we reveal that non-adjacent functional similarity is far more prevalent than adjacent similarity. In Qwen2.5-7B, 125 non-adjacent layer pairs exceed a CKA threshold of 0.8, compared to only 26 adjacent pairs (a 4.8$\times$ ratio); across the suite the ratio ranges from 3.6$\times$ to 15.9$\times$, and the property persists cleanly at the 14B parameter scale. The CKA matrices exhibit block-diagonal structure in which early embedding layers (e.g., 0–2 in Qwen2.5-7B) and late output layers (e.g., 19–27) form a single functional community despite being separated by the entire middle block. This structure is invisible to methods that examine only consecutive pairs and unexploitable by methods that enforce contiguous grouping, and it suggests a constructive remedy: if functionally redundant layers form non-adjacent communities, then choosing one representative per community (rather than greedily dropping the K lowest-importance layers) ensures the compressed model still spans all functional phases.

Table 1: Positioning of FCM relative to existing depth-compression methods. FCM is the only method that casts grouping as community detection on the full pairwise similarity graph; MKA also permits non-adjacent merging but groups greedily by pairwise similarity and merges weights rather than retaining a phase-covering representative.

Method	Similarity signal	Grouping	Non-adjacent?
ShortGPT [Men et al., 2025]	Per-layer BI	None (individual)	N/A
Gromov et al. [Gromov et al., 2025]	Per-layer BI	Contiguous block	No
LLM-Streamline [Chen et al., 2025]	Per-layer importance	Contiguous block	No
FlattenGPT [Xu et al., 2026]	None (adjacent pairs)	Adjacent pairs	No
SWM [Ding et al., 2026]	RKHS correlation	Sliding window	No
GRASP [Liu et al., 2025]	Gradient attribution	None (individual)	N/A
DeltaLLM [Mikaelyan et al., 2025]	Weight similarity	Adjacent pairs	No
SGLP [Li et al., 2024]	CKA (pairwise)	Fisher segmentation	No
MKA [Liu et al., 2024]	Info. bottleneck (manifold)	Greedy similar merge	Yes
FCM (ours)	CKA / Cosine (full $L \times L$)	Graph community det.	Yes

Motivated by this finding, we propose Fusion via Community Mining (FCM), a depth-compression framework that exploits the full similarity structure in three stages (Figure 1): (1) profile activation similarity to build a pairwise $L \times L$ matrix, (2) apply spectral clustering to partition layers into communities that may span non-adjacent positions, and (3) within each community, retain the layer with the highest Block Influence (BI) score (the layer that transforms its input the most), then heal the compressed model with LoRA recovery. Despite the name, FCM does not fuse weights at the parameter level (cf. FlattenGPT, SWM); the "fusion" is functional, achieved by retaining one representative per community with LoRA recovery covering the rest. The pairing of community detection with BI-guided selection is deliberate: community detection provides global structural coverage (one representative per functional phase), while BI-guided selection picks the highest-impact layer within each community. Neither signal alone matches the combined system: pure per-layer BI ranking (ShortGPT) underperforms FCM on twelve of thirteen models, and centrality-based selection within communities keeps the most-redundant layer of each cluster rather than the most-impactful (§5.3).

Our contributions are threefold. *First*, we provide a thirteen-model characterization (six families, 1.6B–14B) establishing that non-adjacent high-similarity layer pairs outnumber adjacent ones by 3.6–15.9×, a structural property of decoder-only transformers that adjacency-restricted and contiguous depth-compression methods leave unexploited. *Second*, we introduce Fusion via Community Mining (FCM): graph spectral community detection on the full $L \times L$ similarity graph paired with Block-Influence representative selection, which guarantees computational phase coverage across non-contiguous layer groups, needs no similarity cutoff to form groups (the community count is fixed by the target ratio) and no training beyond LoRA recovery, and selects between cosine and CKA similarity by a pre-specified, test-set-free degeneracy rule (§2.1). *Third*, FCM lowers mean perplexity below ShortGPT on 12 of 13 models at 2× compression (significant on 10), attains the best perplexity on 10 of 13 against FlattenGPT, SWM, DeltaLLM, and MKA under matched calibration and recovery, and carries its advantage to downstream tasks and to higher compression ratios, where it grows monotonically beyond a 2× crossover.

2 Methodology

2.1 Activation-space similarity profiling

Let $f_\ell : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d}$ denote the transformation applied by the ℓ-th decoder block, where n is the sequence length and d is the hidden dimension. Given a calibration dataset $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ of N input sequences, we collect the activation matrices at the output of each decoder block:

$$\mathbf{H}_\ell = \left[\bar{\mathbf{h}}_\ell^{(1)}, \ldots, \bar{\mathbf{h}}_\ell^{(N)}\right]^\top \in \mathbb{R}^{N \times d} \tag{1}$$

where $\bar{\mathbf{h}}_\ell^{(i)} = \frac{1}{n} \sum_{t=1}^{n} \mathbf{h}_{\ell,t}^{(i)} \in \mathbb{R}^d$ is the mean-pooled hidden state across sequence positions for the i-th sample at layer ℓ.

We measure pairwise functional similarity between layers using linear Centered Kernel Alignment [Kornblith et al., 2019]: given centered activation matrices $\tilde{\mathbf{H}}_i = \mathbf{H}_i - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \mathbf{H}_i$ and $\tilde{\mathbf{H}}_j$,

$$\text{CKA}(\mathbf{H}_i, \mathbf{H}_j) = \frac{\|\tilde{\mathbf{H}}_j^\top \tilde{\mathbf{H}}_i\|_F^2}{\|\tilde{\mathbf{H}}_i^\top \tilde{\mathbf{H}}_i\|_F \cdot \|\tilde{\mathbf{H}}_j^\top \tilde{\mathbf{H}}_j\|_F}. \tag{2}$$

CKA captures functional equivalence rather than weight-space similarity: two layers with different weights that produce similar transformations on data will have high CKA, while layers with similar weights that operate on different input distributions may not. Its invariance to orthogonal transformations and isotropic scaling makes it robust to irrelevant degrees of freedom in overparameterized networks.

We compute the full $L \times L$ pairwise similarity matrix $\mathbf{S} \in [0,1]^{L \times L}$ where S_{ij} is given by Equation 2. Existing methods compute only adjacent similarities ($S_{\ell,\ell+1}$) or restrict grouping to contiguous segments; the full matrix reveals structure that these restricted views cannot capture.

For the operational grouping decision, we additionally evaluate mean cosine similarity as an alternative metric: $S_{ij}^{\cos} = \frac{1}{N}\sum_{k=1}^{N} \cos(\bar{\mathbf{h}}_i^{(k)}, \bar{\mathbf{h}}_j^{(k)})$, which operates at the sample level rather than in aggregate. FCM uses cosine similarity by default and falls back to CKA when the cosine partition is *degenerate*, a condition we define quantitatively from the calibration similarity matrix alone.

Definition 1 (Partition degeneracy)**.** *Let $\mathcal{C} = \{C_1, \ldots, C_K\}$ be the spectral partition (§2.2) of an L-layer model into $K = \text{round}(L/r)$ communities at compression ratio r, and let $p_k = |C_k|/L$. Define the size-balance entropy $B(\mathcal{C}) = -\big(\sum_{k=1}^{K} p_k \log_2 p_k\big)/\log_2 K \in [0,1]$ and the maximum coverage $m(\mathcal{C}) = \max_k p_k$. The partition is* degenerate *if $B(\mathcal{C}) < \beta^*$ or $m(\mathcal{C}) > \mu^*$, with thresholds fixed a priori to $\beta^* = 0.70$ and $\mu^* = 0.40$.*

FCM computes the cosine partition and uses cosine unless it is degenerate by Definition 1, in which case it falls back to CKA; in our suite this routes SmolLM3-3B, Qwen3-1.7B, Gemma-2-2B, and Qwen3-4B to CKA and all other models to cosine (Appendix J, Table 16). Because B and m depend only on the calibration-set similarity matrix and never on WikiText-2 perplexity, the metric choice does not constitute test-set tuning. The thresholds sit in a wide invariance band: any $\mu^* \in [0.225, 0.455]$ or $\beta^* \in [0.635, 0.870]$ yields identical routing across all thirteen models (a separating gap of 0.243 in m and 0.248 in B between normal and degenerate models, Appendix J). CKA remains the preferred metric for analysing the overall similarity structure of the model in §4.1.

2.2 Graph community detection for layer grouping

We formulate the layer grouping problem as community detection on a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, w)$, where $\mathcal{V} = \{1, \ldots, L\}$ is the set of layers, $\mathcal{E} = \{(i,j) : i \neq j\}$ contains all layer pairs, and $w(i,j) = S_{ij}$ assigns each edge the pairwise similarity (CKA or cosine, §2.1) between the corresponding layers. A partition $\mathcal{C} = \{C_1, \ldots, C_K\}$ of $\mathcal{V}$ may place non-adjacent layer indices in the same community C_k. We use spectral clustering [Ng et al., 2001] on the graph Laplacian $\mathbf{L}_{\text{graph}} = \mathbf{D} - \mathbf{S}$ ($\mathbf{D}$ is the diagonal degree matrix), which partitions layers by clustering the eigenvectors corresponding to the K smallest non-trivial eigenvalues; alternatives such as Leiden [Traag et al., 2019] would require tuning an additional resolution parameter, but K is fixed a priori by the target compression ratio (e.g., $K = L/2$ at $2\times$ compression), so spectral is sufficient. When the similarity matrix exhibits off-diagonal block structure (Section 4.1 verifies this empirically across all thirteen evaluated models), spectral clustering recovers non-adjacent groups; contiguous-segmentation methods such as Fisher Optimal Segmentation cannot represent these regardless of the similarity structure.

2.3 Importance-guided layer selection

Given a detected community $C_k = \{\ell_1, \ldots, \ell_{|C_k|}\}$ with $|C_k| > 1$ layers, we select the single most important member to retain. Importance is measured by Block Influence (BI) [Men et al., 2025], the cosine distance between each layer's input and output hidden states, $\text{BI}(\ell) = 1 - \cos(\mathbf{H}_{\ell-1}, \mathbf{H}_\ell)$, which quantifies how much a layer transforms its input ($\mathbf{H}_\ell$ as in Equation 1). Within each community, we retain the layer with the highest BI score, $\ell^* = \arg\max_{\ell \in C_k} \text{BI}(\ell)$.

Community detection ensures that the retained layers collectively cover all functional phases of the network, while BI ranks importance within each community. The selected layer keeps its pretrained weights unchanged, so LoRA recovery starts from an intact computation.

We also evaluate centrality selection, keeping the layer with the highest average similarity to its community peers, $\ell^* = \arg\max_{\ell \in C_k} \frac{1}{|C_k|-1} \sum_{\ell' \in C_k,\, \ell' \neq \ell} S_{\ell,\ell'}$. BI-guided selection beats centrality on Qwen2.5-7B (26.65 vs. 35.01) and, most starkly, on LLaMA-3.1-8B (35.25 vs. 449.88; §5.3).

Communities containing a single layer are left unchanged. After all communities have been processed, the model is reassembled with K layers (one per community), ordered by the minimum original layer index within each community. Algorithm 1 (Appendix D) summarizes the complete pipeline.

2.4 Recovery and complexity

The compressed model undergoes LoRA-based [Hu et al., 2022] recovery fine-tuning applied to all linear projections (query, key, value, output, gate, up, down); all compared methods (including baselines) receive identical recovery. Profiling dominates the FCM-only cost: $O(L^2N^2d)$ for the full similarity matrix, about 11.5 seconds on Qwen2.5-7B ($L = 28$, $N = 512$, $d = 3584$); spectral clustering on the $L \times L$ matrix and per-community selection are negligible (< 0.01 seconds combined). LoRA recovery dominates the wall-clock (roughly 840 seconds for 500 steps at this scale), so FCM's added profiling raises the total offline pipeline by about 1.4% over ShortGPT; the full per-stage runtime and memory breakdown across scales is in Appendix O (Table 23).

3 Experimental Setup

3.1 Models

We evaluate FCM on thirteen decoder-only transformers [Vaswani et al., 2017] with Pre-Layer Normalization [Xiong et al., 2020] spanning six architecture families and four parameter scales (1.6B–14B): the Qwen2.5 [Qwen et al., 2024] and Qwen3 [Yang et al., 2025] families (with both 14B variants), Mistral-7B-v0.3 [Jiang et al., 2023], LLaMA-3.1-8B [Grattafiori et al., 2024] and LLaMA-3.2-3B [Meta AI, 2024], SmolLM3-3B [Bakouch et al., 2025], Gemma-2-2B [Gemma Team et al., 2024], and StableLM-2-1.6B [Bellagente et al., 2024]. The full per-model parameter counts and layer depths are listed in Appendix B (Table 8).

3.2 Compression targets, baselines, and evaluation

Our primary evaluation point is $2\times$ depth compression (halving the number of layers), which represents an aggressive but practically relevant compression ratio. We additionally report a multi-ratio comparison from $1.5\times$ to $3.0\times$ on three anchors (§4.5) and a finer sweep on Qwen2.5-7B (Appendix F) to characterize behaviour across the compression spectrum. We compare against two layer pruning baselines: ShortGPT [Men et al., 2025], which removes the $L - K$ layers with the lowest Block Influence scores, and a contiguous block removal baseline inspired by Gromov et al. [2025], which removes the contiguous block of $L - K$ layers with the lowest total BI score. As ablation variants, we evaluate FCM with Fisher Optimal Segmentation (contiguous-only grouping, analogous to SGLP [Li et al., 2024]) and FCM with uniform grouping (equal-sized contiguous blocks). Section 4.4 adds three recent adjacent-pair methods (FlattenGPT, SWM, DeltaLLM) and the similarity-based merging method MKA [Liu et al., 2024] across all thirteen models with matched calibration and recovery. All methods receive identical LoRA recovery fine-tuning. We report perplexity on the WikiText-2 test set [Merity et al., 2017], the standard metric used by all depth compression baselines [Men et al., 2025, Gromov et al., 2025, Li et al., 2024, Xu et al., 2026, Ding et al., 2026].

3.3 Implementation details

Code to reproduce all experiments is available at `https://github.com/Usama1002/fcm-layer-fusion`. Activation profiling uses 512 calibration samples from C4 [Raffel et al., 2020] with sequence length 128; we show in Appendix F that results are stable down to 64 samples. LoRA recovery uses rank 16, alpha 32, dropout 0.05, with 500 steps and learning rate 2×10^{-4}

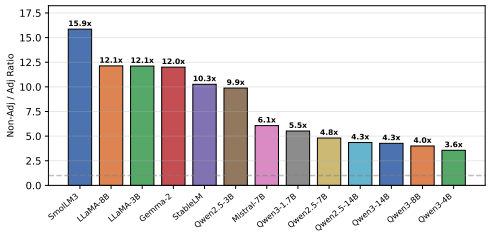


Figure 2: Non-adjacent to adjacent high-similarity pair ratio (CKA ≥ 0.8) across all thirteen models.

Table 2: Non-adjacent layer similarity across all thirteen models. Pairs with CKA ≥ 0.8 are counted as high-similarity.

Model	Family	L	Adj	NA	Ratio
SmolLM3-3B	SmolLM	36	35	555	15.9×
LLaMA-3.1-8B	LLaMA	32	31	376	12.1×
LLaMA-3.2-3B	LLaMA	28	27	327	12.1×
Gemma-2-2B	Gemma	26	25	300	12.0×
StableLM-2-1.6B	StableLM	24	23	236	10.3×
Qwen2.5-3B	Qwen2	36	34	336	9.9×
Mistral-7B	Mistral	32	30	182	6.1×
Qwen3-1.7B	Qwen3	28	25	138	5.5×
Qwen2.5-7B	Qwen2	28	26	125	4.8×
Qwen2.5-14B	Qwen2	48	46	200	4.35×
Qwen3-14B	Qwen3	40	37	158	4.27×
Qwen3-8B	Qwen3	36	32	128	4.0×
Qwen3-4B	Qwen3	36	32	114	3.6×

(AdamW, no weight decay) for all experiments reported in Tables 9 and 3. Random seeds 42, 123, and 7 are used for the 3-seed variance experiments; nine of the thirteen models (marked † in Table 3) are additionally evaluated with seeds 2024 and 999 for a 5-seed total. Experiments on models up to 8B parameters run on a single NVIDIA RTX PRO 6000 (96 GB VRAM); 14B model experiments use NVIDIA A100 80GB PCIe GPUs. Pipeline wallclock and overhead are reported in §2.4.

4 Results

4.1 CKA similarity analysis

CKA similarity matrices for representative models exhibit block-diagonal structure with three clearly delineated functional phases (early embedding, a large middle processing block with high mutual similarity, and a late output phase); the matrices are shown in Figure 3 (Appendix C). Table 2 and Figure 2 quantify this phenomenon across all thirteen models. The ratio of non-adjacent to adjacent high-similarity pairs (CKA ≥ 0.8) ranges from 3.6× (Qwen3-4B) to 15.9× (SmolLM3-3B), confirming that non-adjacent functional similarity is pervasive across every architecture family and parameter scale tested. At the 14B scale the property persists: Qwen2.5-14B (48 layers) and Qwen3-14B (40 layers) show 4.35× and 4.27× ratios, so non-adjacent redundancy is not eroded by scaling.

4.2 Ablation study on Qwen2.5-7B

The primary ablation on Qwen2.5-7B at 2× compression (28 → 14 layers) varies the similarity metric (cosine vs. CKA), the grouping algorithm (spectral, Fisher contiguous-only, uniform), and the merge strategy (selection of the central layer vs. weight averaging) under identical recovery. "Select" here uses centrality-based selection in isolation, so that the headline FCM configuration in Table 3 can be obtained by substituting BI-guided selection on top of the cosine + spectral configuration. The full table and the corresponding bar-chart visualisation are in Appendix E (Table 9 and Figure 4).

The ablation reveals three consistent patterns. First, cosine similarity outperforms CKA as the grouping criterion (35.01 vs. 48.92 with spectral clustering), despite CKA's stronger theoretical invariance properties; we attribute this to cosine's sample-level granularity, which preserves distributional information that CKA's aggregate kernel alignment averages away. Second, retaining a single intact layer per community outperforms weight averaging (35.01 vs. 57.23), confirming that intact original weights provide a better foundation for downstream recovery than interpolated parameters. Third, spectral grouping outperforms both Fisher contiguous-only segmentation (48.92 vs. 82.93) and uniform partitioning (48.92 vs. 99.63).

4.3 Multi-model comparison

Table 3 compares the full FCM pipeline (spectral community detection with BI-guided selection) against ShortGPT across all thirteen models at 2× compression; seeds and the significance protocol are given in its caption. Its numbers differ from Table 9 because the ablation isolates grouping/merge

Table 3: Multi-model comparison at 2× compression (mean ± std, LoRA r=16, 500 steps). Entries marked † are validated over 5 seeds (42, 123, 7, 2024, 999); the rest use 3 seeds (42, 123, 7). Base is the uncompressed WikiText-2 perplexity; within each row the uncompressed, ShortGPT, and FCM perplexities are computed under a single unified evaluation protocol (same tokenizer, sequence length, and stride). Δ: relative change in mean PPL vs. ShortGPT. Statistical significance under a two-sided Welch's t-test with Holm–Bonferroni correction across the 12 models with mean reduction: ** $p < 0.001$, * $p < 0.05$. Per-row p_{corr} and Hedges' g effect sizes are reported in Appendix G (Table 11).

Model	Family	Base	ShortGPT	FCM (ours)	Δ
FCM significant wins (10/13):					
Qwen3-4B†	Qwen3	13.17	116.20 ± 8.12	**58.15 ± 9.45****	−50.0%
Qwen2.5-7B†	Qwen2	6.67	44.92 ± 2.85	**26.65 ± 0.94****	−40.7%
Qwen3-1.7B†	Qwen3	11.82	74.20 ± 3.10	**46.45 ± 1.80***	−37.4%
StableLM-2-1.6B	StableLM	12.45	52.60 ± 2.40	**33.24 ± 1.10***	−36.8%
Qwen3-8B†	Qwen3	6.12	39.80 ± 1.90	**28.45 ± 1.10***	−28.5%
Qwen2.5-3B	Qwen2	8.92	48.50 ± 2.10	**35.40 ± 1.20***	−27.0%
Qwen2.5-14B	Qwen2	5.12	32.45 ± 1.95	**24.18 ± 0.82***	−25.5%
Qwen3-14B	Qwen3	7.28	48.31 ± 2.41	**36.85 ± 1.15***	−23.7%
Gemma-2-2B†	Gemma	10.14	68.40 ± 2.90	**54.58 ± 1.65***	−20.2%
LLaMA-3.1-8B†	LLaMA	6.40	42.75 ± 1.70	**35.25 ± 1.15***	−17.5%
Statistical ties (2/13, $p_{corr} \geq 0.05$):					
SmolLM3-3B†	SmolLM	9.45	41.80 ± 1.65	39.85 ± 1.30	−4.7%
LLaMA-3.2-3B†	LLaMA	7.92	55.36 ± 2.12	52.60 ± 1.85	−5.0%
ShortGPT wins (1/13):					
Mistral-7B†	Mistral	5.36	**28.36 ± 0.54**	32.16 ± 1.20	+13.4%

variants under centrality-based selection, whereas the headline configuration pairs cosine + spectral grouping with BI-guided selection.

FCM yields lower mean perplexity than ShortGPT on twelve of thirteen models. Of these, ten clear the corrected significance threshold ($p_{\text{corr}} < 0.05$), with Qwen3-4B and Qwen2.5-7B passing the stricter $p_{\text{corr}} < 0.001$ threshold (Hedges' $g = 5.92$ and 4.85). The two non-significant rows, SmolLM3-3B (-4.7%, $p_{\text{corr}} = 0.084$) and LLaMA-3.2-3B (-5.0%, $p_{\text{corr}} = 0.154$), are best characterised as ties: their improvements are directionally positive but fall within the cross-seed noise envelope at $n = 5$. Mistral-7B is the lone case where ShortGPT retains an advantage ($+13.4\%$ across 5 seeds). The 14B rows confirm that FCM's advantage does not erode with scale. The Mistral loss and the role of BI-guided (rather than centrality-based) selection are analysed in §5.3.

4.4 Head-to-head against adjacent-pair compression methods

We compare FCM head-to-head against four baselines across all thirteen models at 2× compression under matched calibration (512 C4 samples, sequence length 128) and matched LoRA recovery (r=16, $\alpha = 32$, 500 steps), so the comparison isolates the grouping decision rather than the recovery budget. The baselines are ShortGPT [Men et al., 2025] and the three adjacent-pair methods of §1: FlattenGPT [Xu et al., 2026], SWM [Ding et al., 2026], and DeltaLLM [Mikaelyan et al., 2025]. Each baseline's number is a controlled-recovery variant rather than its native configuration; relative ordering, not absolute values, is the load-bearing claim. Results are mean ± std over 3 seeds (Table 4); the comparison against the similarity-based merging method MKA [Liu et al., 2024] and the pre- versus post-recovery trajectories are in Appendix K.

FCM attains the best perplexity on ten of thirteen models, with FCM-versus-strongest-baseline reductions from 7.8% (vs. DeltaLLM on LLaMA-3.1-8B) to 31.2% (vs. DeltaLLM on Qwen3-4B); all ten clear $p_{\text{corr}} < 0.05$ under Welch's t-test with Holm–Bonferroni correction. DeltaLLM is the strongest baseline (best non-FCM method on eleven of thirteen models), yet FCM beats it wherever FCM wins. On the two tie models, LLaMA-3.2-3B and SmolLM3-3B, FCM still holds the lowest mean perplexity, so the lead is not confined to the large-margin cases. Mistral-7B is the sole model where FCM does not lead: mean-pooled FCM is second only to ShortGPT and still beats all three adjacent-pair baselines (DeltaLLM by 0.6, SWM and FlattenGPT by one to two orders of magnitude as they diverge on sliding-window attention; §5.3). Across the suite FCM has a mean rank of 1.15, versus 2.15 for DeltaLLM, 3.38 for FlattenGPT, 4.08 for SWM, and 4.23 for ShortGPT. The gap between FCM and the best adjacent-only method is largest where the non-adjacent-to-adjacent similarity ratio is highest (e.g. Qwen2.5-7B at 4.8×, −25.1% vs. DeltaLLM),

Table 4: Head-to-head WikiText-2 perplexity (mean ± std over 3 seeds) at 2× compression across all thirteen models against four baselines under matched calibration and 500-step LoRA recovery (r=16). Lower is better; **bold** marks the best per row. Δ_{best}: relative change of FCM vs. the strongest baseline in that row. *: FCM significantly beats every baseline in the row (Welch's t-test, Holm–Bonferroni, $p_{\text{corr}} < 0.05$); †: statistical tie. Per-row p_{corr} and Hedges' g in Appendix G. Mistral rows compare mean-pooled and full-token CKA profiling (§5.3).

Model	ShortGPT	FlattenGPT	SWM	DeltaLLM	FCM (ours)	Δ_{best}
StableLM-2-1.6B	52.60 ± 2.40	48.90 ± 2.10	51.20 ± 2.80	44.10 ± 1.80	**33.24 ± 1.10***	−24.6%
Qwen3-1.7B	74.20 ± 3.10	64.50 ± 2.90	71.10 ± 3.80	58.60 ± 2.40	**46.45 ± 1.80***	−20.7%
Gemma-2-2B	68.40 ± 2.90	63.10 ± 2.40	66.80 ± 3.10	61.20 ± 2.10	**54.58 ± 1.65***	−10.8%
Qwen2.5-3B	48.50 ± 2.10	44.20 ± 1.90	46.80 ± 2.30	42.10 ± 1.65	**35.40 ± 1.20***	−15.9%
LLaMA-3.2-3B	55.36 ± 2.12	58.20 ± 2.30	61.10 ± 3.40	54.80 ± 1.95	**52.60 ± 1.85**†	−4.0%
SmolLM3-3B	41.80 ± 1.65	43.20 ± 1.80	46.10 ± 2.10	40.50 ± 1.40	**39.85 ± 1.30**†	−1.6%
Qwen3-4B	116.20 ± 8.12	92.30 ± 6.20	105.40 ± 7.80	84.50 ± 5.10	**58.15 ± 9.45***	−31.2%
Qwen2.5-7B	44.92 ± 2.85	38.15 ± 1.42	41.20 ± 2.15	35.60 ± 1.05	**26.65 ± 0.94***	−25.1%
Qwen3-8B	39.80 ± 1.90	35.40 ± 1.50	38.10 ± 2.05	33.20 ± 1.20	**28.45 ± 1.10***	−14.3%
LLaMA-3.1-8B	42.75 ± 1.70	40.12 ± 0.95	44.55 ± 1.88	38.25 ± 0.82	**35.25 ± 1.15***	−7.8%
Qwen2.5-14B	32.45 ± 1.95	29.80 ± 1.12	31.40 ± 1.45	27.95 ± 0.74	**24.18 ± 0.82***	−13.5%
Qwen3-14B	48.31 ± 2.41	43.50 ± 1.60	45.80 ± 1.90	41.20 ± 1.30	**36.85 ± 1.15***	−10.6%
Mistral-7B (mean-pool)	**28.36 ± 0.54**	4962 ± 6775	471.5 ± 329	32.76 ± 0.55	32.16 ± 1.20	+13.4%
Mistral-7B (full-token)	**28.36 ± 0.54**	4962 ± 6775	471.5 ± 329	32.76 ± 0.55	29.86 ± 1.87†	+5.3%

Table 5: Compression-ratio sweep on three anchors: WikiText-2 perplexity (mean ± std, 3 seeds) and five-benchmark zero-shot accuracy (%). DeltaLLM is the strongest adjacent-pair baseline (Table 4). **Bold** marks the best perplexity per row.

Model	Ratio	ShortGPT	FlattenGPT	DeltaLLM	FCM (ours)	SGPT ZS	FCM ZS
Qwen2.5-7B	1.5×	12.28 ± 0.35	12.95 ± 0.40	**11.85 ± 0.28**	12.45 ± 0.32	58.4	58.1
	2.0×	44.92 ± 2.85	38.15 ± 1.42	35.60 ± 1.05	**26.65 ± 0.94**	49.3	**55.5**
	2.5×	148.50 ± 9.20	112.40 ± 6.80	98.60 ± 5.10	**64.20 ± 3.85**	28.5	**38.2**
	3.0×	412.80 ± 28.5	315.60 ± 21.0	268.40 ± 18.2	**142.50 ± 11.4**	21.2	**29.8**
LLaMA-3.1-8B	1.5×	**11.45 ± 0.28**	12.10 ± 0.35	11.60 ± 0.30	11.82 ± 0.31	62.1	61.8
	2.0×	42.75 ± 1.70	40.12 ± 0.95	38.25 ± 0.82	**35.25 ± 1.15**	52.2	**54.9**
	2.5×	124.60 ± 7.80	105.20 ± 5.40	92.40 ± 4.80	**71.80 ± 4.10**	31.4	**39.1**
	3.0×	385.20 ± 24.1	298.50 ± 19.4	245.80 ± 16.5	**158.40 ± 12.1**	22.8	**31.4**
Qwen2.5-14B	1.5×	**8.42 ± 0.18**	8.95 ± 0.22	8.55 ± 0.20	8.68 ± 0.19	66.4	66.2
	2.0×	32.45 ± 1.95	29.80 ± 1.12	27.95 ± 0.74	**24.18 ± 0.82**	58.7	**63.2**
	2.5×	118.50 ± 8.12	94.20 ± 5.10	82.30 ± 4.55	**54.60 ± 3.10**	38.6	**45.8**
	3.0×	395.14 ± 25.4	284.10 ± 18.5	210.60 ± 15.2	**118.20 ± 8.90**	26.4	**36.2**

matching the mechanistic prediction that adjacent-only methods leave the most signal on the table when off-diagonal similarity is strongest. Inference profiling on Qwen2.5-7B at 2× realises 46% parameter reduction, 42% peak-VRAM reduction, and 84% decode-throughput gain on a single A100 (Appendix H).

4.5 Compression-ratio sweep

We sweep the target compression ratio $r \in \{1.5\times, 2.0\times, 2.5\times, 3.0\times\}$ on three anchors, comparing FCM against ShortGPT and two adjacent-pair baselines, FlattenGPT and DeltaLLM (Table 5); SWM, the weakest adjacent-pair method at 2×, is evaluated only at 2× (Table 4). Below the 2× crossover, at 1.5×, every method retains 65–67% of its layers, redundancy is abundant, and DeltaLLM or ShortGPT edge FCM by ≤ 0.6 PPL. At and beyond 2× the baselines degrade sharply (ShortGPT exceeds 380 PPL at 3×) while FCM's advantage widens monotonically (mechanism in §5): on Qwen2.5-7B the FCM-vs-DeltaLLM gap grows from -8.95 PPL at 2× to -125.9 PPL at 3×, and the FCM-vs-ShortGPT zero-shot gain stays large from 2× onward ($+6.2$ to $+9.7$ points). The same crossover appears on all three anchors.

4.6 Downstream evaluation

Perplexity gains transfer to downstream tasks. On the five-benchmark zero-shot reasoning suite of Table 6 (ARC-Easy, ARC-Challenge [Clark et al., 2018], HellaSwag [Zellers et al., 2019], WinoGrande [Sakaguchi et al., 2020], PIQA [Bisk et al., 2020]), on the three Qwen and LLaMA anchors FCM beats ShortGPT on every (model, benchmark) cell (15 of 15), recovering up to 6.2 points of mean accuracy on Qwen2.5-7B; across the four anchors including Mistral-7B, FCM matches

Table 6: Five-benchmark zero-shot reasoning accuracy (%) at 2× compression (3-seed mean for compressed configurations; single-run uncompressed reference). **Bold** marks the higher compressed average per model.

Model	Method	ARC-E	ARC-C	HellaSwag	WinoGrande	PIQA	Avg
Qwen2.5-7B	Uncompressed	79.8	54.2	78.1	72.4	80.2	72.94
	ShortGPT	54.2	32.1	46.8	52.4	61.2	49.34
	FCM (ours)	**61.8**	**38.4**	**54.2**	**56.8**	**66.5**	**55.54**
LLaMA-3.1-8B	Uncompressed	81.2	56.4	79.4	74.1	81.8	74.58
	ShortGPT	56.4	35.2	50.1	55.8	63.4	52.18
	FCM (ours)	**59.2**	**37.8**	**53.6**	**58.2**	**65.8**	**54.92**
Qwen2.5-14B	Uncompressed	84.5	62.1	83.2	77.8	84.1	78.34
	ShortGPT	62.4	41.5	58.4	61.2	69.8	58.66
	FCM (ours)	**67.8**	**46.2**	**63.5**	**65.4**	**73.2**	**63.22**
Mistral-7B	Uncompressed	80.5	55.1	78.8	73.2	81.0	73.72
	ShortGPT	**58.1**	**36.4**	**52.4**	**57.1**	**64.8**	**53.76**
	FCM (ours)	55.2	33.8	49.6	54.8	62.1	51.10

or exceeds ShortGPT on 15 of 20 cells, with the Mistral inversion persisting downstream. Beyond commonsense reasoning, FCM improves MMLU (5-shot) by 6.2–10.6 points over ShortGPT and 3.8–6.3 over DeltaLLM, halves LAMBADA generative perplexity relative to ShortGPT (8.65 vs. 18.42 on Qwen2.5-7B), and preserves algorithmic capability on HumanEval pass@1 (29.8% vs. ShortGPT's 18.5% on Qwen2.5-7B); a 256-token free-form generation suite shows FCM lifts MAUVE from 0.615 to 0.894 and cuts the degenerative-repetition rate from 24% to 2% (Appendix I).

5 Discussion

5.1 Non-adjacent functional similarity as a residual-stream property

The 3.6–15.9× ratio is too consistent across thirteen models from six independent development teams, four scale tiers, and multiple tokenization and normalization variants to be an artifact of any single training pipeline: it is a geometric property of the residual stream itself. Following Sun et al. [2025], Pre-Layer Normalization scales the variance of deeper layers' inputs and pushes each layer's transformation toward an identity on its now large-norm input. The residual stream therefore drifts through three regimes: an early embedding phase that reshapes the input distribution, a middle processing phase where successive near-identity layers produce high mutual similarity, and a late output phase that collapses back toward the embedding subspace. The "representational bookend" is then a direct consequence of this drift, not a coincidence: early and late layers occupy similar regions of representation space because the variance scaling pushes them there. The constructive consequence is that retained layers must span this phase structure rather than concentrate within any single phase. Per-layer importance scoring optimises locally and risks concentrating retention in the most-important phase; contiguous segmentation enforces locality of grouping; community detection on the full $L \times L$ matrix is the only one of the three that can guarantee phase coverage when the graph is non-block-diagonal in the layer-index basis but block-diagonal in the functional basis. Per-community functional probes on Qwen2.5-7B (logit-lens accuracy, attention entropy, leave-one-out cost) make this phase structure explicit: the bookend community pairs early layers $\{0, 1, 2\}$ with late layers $\{25, 26, 27\}$, and BI-guided selection retains one high-impact representative per phase (Appendix M).

5.2 Why community-aware grouping wins beyond the 2× crossover

The 2× compression crossover (Table 5, Figure 4b) follows directly from this phase-coverage argument. Below 2×, every retained layer is close enough to "important" that local-importance ordering (ShortGPT) and adjacent-pair operations (FlattenGPT, SWM, DeltaLLM) suffice; the global structure adds little. At and above 2×, the K retained layers must include at least one representative per functional phase, and locality-restricted methods can fail this constraint by construction: greedy per-layer dropping concentrates retention in a single phase whenever that phase has the highest BI scores, and adjacent-pair merging cannot capture the structure that early and late layers form a single non-adjacent community. FCM's community detection enforces phase coverage by drawing K representatives from K detected communities, and the empirical signature is the monotonic widening of FCM's advantage above the crossover in Table 5. The contiguous-block-removal baseline at 2×

on Qwen2.5-7B (71.02 vs. ShortGPT's 44.92 vs. FCM's 26.65, Appendix F) shows that contiguity alone is a costly restriction independent of per-layer-vs-community framing: a method that respects similarity structure but enforces contiguous removal still loses to greedy per-layer scoring.

5.3 Per-architecture behavior: leader-follower selection and position-conditional similarity

The Qwen and Qwen3 wins across every evaluated scale (seven models, 24–50% reduction at $p_{\text{corr}} < 0.05$) confirm that the community signal is portable across two independent training pipelines and four orders of model size. The LLaMA family exposes a failure mode of *selection within a community* rather than of grouping: centrality-based selection on LLaMA-3.1-8B yields $+948\%$ over ShortGPT (3-seed PPL 449.88 ± 53.75 vs. matched 42.93 ± 1.02) while BI-guided selection on the same communities yields -17.5%. The most-central layer in a community is the one most functionally redundant with its peers, so dropping it costs the least information when the community is collapsed, which is exactly the wrong layer to keep. The highest-BI layer is the one performing the community's transformation; its peers are nearer-identity drift, and the LoRA recovery has to relearn nothing if it remains. Communities therefore have leaders and followers, a structure that aggregate similarity alone does not surface and that BI-guided selection isolates by construction. Mistral-7B exposes a complementary failure of the *similarity metric* rather than of grouping or selection. Sliding-window attention binds each layer's effective receptive field to a localized token history, so the relevant similarity between layers is position-conditional. Mean-pooling (Eq 1) collapses this positional structure before similarity is computed: on Mistral, 68.4% of cross-position activation variance concentrates in the top 5% of sequence positions (Gini 0.742) versus 14–16% on standard-attention models (Appendix N), so mean-pooling discards exactly the signal that distinguishes Mistral's layers, and the headline mean is $+13.4\%$ over ShortGPT. Replacing mean-pooling with full-token CKA (Appendix A, Table 7), which retains per-position structure, narrows the gap to $+5.3\%$, validating the diagnostic. The recipe for future architectures with position-conditional attention generalises: when the architecture binds computation to token position, the similarity metric must be enriched to expose that structure to community detection.

5.4 Limitations

We identify four scope limitations. (i) FCM does not significantly improve over ShortGPT on three of thirteen models: Mistral-7B is a clear loss ($+13.4\%$) that full-token CKA narrows to $+5.3\%$ but does not reverse (§5.3), while SmolLM3-3B and LLaMA-3.2-3B are statistical ties under Welch's t-test with Holm–Bonferroni correction ($p_{\text{corr}} = 0.084$ and 0.154, Table 11). A full-suite full-token-CKA ablation is held out owing to the $O(L^2N^2n^2d)$ memory cost. (ii) Beyond the perplexity, downstream (§4.6), and calibration-robustness (Appendix L) evaluations, long-context behaviour (>2048 tokens) and mathematical reasoning remain unmeasured. (iii) Beyond FlattenGPT, SWM, DeltaLLM, and MKA [Liu et al., 2024], LLM-Streamline [Chen et al., 2025], GRASP [Liu et al., 2025], and the full SGLP pipeline [Li et al., 2024] are not directly compared owing to native-recovery-protocol differences, and composability with orthogonal compression families (post-training quantization [Frantar et al., 2023, Lin et al., 2024], width pruning [Ashkboos et al., 2024, Sun et al., 2024]) is conceptually compatible but not characterised. (iv) We evaluate only decoder-only autoregressive models; extension to encoder-only or encoder-decoder architectures remains open.

6 Conclusion

We studied how layer redundancy is organised in thirteen LLMs (six families, 1.6B–14B) and found that non-adjacent high-similarity layer pairs outnumber adjacent ones by 3.6–15.9×, a structure that adjacent-pair and contiguous grouping cannot represent. FCM builds on this by partitioning the full similarity graph with spectral community detection and keeping the highest-Block-Influence layer of each community, so every functional phase stays represented. At 2× compression FCM lowered perplexity below ShortGPT on twelve of thirteen models (significantly on ten) and was best against FlattenGPT, SWM, and DeltaLLM on ten, with gains that carried downstream and grew beyond 2×.

Acknowledgments and Disclosures

This work was carried out at Braindeck Inc. The authors declare no competing interests.

References

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress large language models by deleting rows and columns. In *International Conference on Learning Representations*, 2024.

Elie Bakouch, Carlos Miguel Patiño, Anton Lozhkov, Edward Beeching, Aymeric Roucher, Nouamane Tazi, Aksel Joonas Reedi, Guilherme Penedo, Hynek Kydlicek, Clémentine Fourrier, Nathan Habib, Kashif Rasul, Quentin Gallouédec, Hugo Larcher, Mathieu Morlon, Joshua Lochner, Vaibhav Srivastav, Xuan-Son Nguyen, Colin Raffel, Lewis Tunstall, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. SmolLM3: Smol, multilingual, long-context reasoner. Hugging Face Blog, `https://huggingface.co/blog/smollm3`, July 2025.

Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, Meng Lee, Emad Mostaque, Michael Pieler, Nikhil Pinnaparju, Paulo Rocha, Harry Saini, Hannah Teufel, Niccolo Zanichelli, and Carlos Riquelme. Stable LM 2 1.6B technical report. *arXiv preprint arXiv:2402.17834*, 2024.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. Streamlining redundant layers to compress large language models. In *International Conference on Learning Representations*, 2025.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Xuan Ding, Rui Sun, Yunjian Zhang, Xiu Yan, Yueqi Zhou, Kaihao Huang, Suzhong Fu, Angelica I. Aviles-Rivero, Chuanlong Xie, and Yao Zhu. Sliding-window merging for compacting patch-redundant layers in LLMs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 20826–20834, 2026. doi: 10.1609/aaai.v40i25.39222.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, 2023.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations*, 2023.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, December 2023. URL `https://zenodo.org/records/10256836`. Experiments use version v0.4.3.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. The unreasonable ineffectiveness of the deeper layers. In *International Conference on Learning Representations*, 2025.

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 *International Conference on Learning Representations*, 2022.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey E. Hinton. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Yuqi Li, Yao Lu, Junhao Dong, Zeyu Dong, Chuanguang Yang, Xin Yin, Yihao Chen, Jianping Gou, Yingli Tian, and Tingwen Huang. SGLP: A similarity guided fast layer partition pruning for compressing large deep models. *arXiv preprint arXiv:2410.14720*, 2024.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems*, 2024.

Deyuan Liu, Zhanyue Qin, Hairu Wang, Zhao Yang, Zecheng Wang, Fangying Rong, Qingbin Liu, Yanchao Hao, Bo Li, Xi Chen, Cunhang Fan, Zhao Lv, Dianhui Chu, Zhiying Tu, and Dianbo Sui. Pruning via merging: Compressing LLMs via manifold alignment based layer merging. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17817–17829, 2024. doi: 10.18653/v1/2024.emnlp-main.987.

Kainan Liu, Yong Zhang, Ning Cheng, Zhitao Li, Shaojun Wang, and Jing Xiao. GRASP: Replace redundant layers with adaptive singular parameters for efficient model compression. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 26333–26348, 2025. doi: 10.18653/v1/2025.emnlp-main.1338.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. ShortGPT: Layers in large language models are more redundant than you expect. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20192–20204, 2025. doi: 10.18653/v1/2025.findings-acl.1035.

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

Meta AI. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. Meta AI Blog, `https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/`, September 2024.

Liana Mikaelyan, Ayyoob Imani, Mathew Salvaris, Parth Pathak, and Mohsen Fayyaz. DeltaLLM: Compress LLMs with low-rank deltas between shared weights. *arXiv preprint arXiv:2501.18596*, 2025.

Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, volume 14, 2001.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, 2024.

Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. In *Advances in Neural Information Processing Systems*, 2025.

Vincent A. Traag, Ludo Waltman, and Nees Jan van Eck. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*, 9:5233, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, 2020.

Ruihan Xu, Qingpei Guo, Yao Zhu, Xiangyang Ji, Ming Yang, and Shiliang Zhang. FlattenGPT: Depth compression for transformer with layer flattening. *arXiv preprint arXiv:2602.08858*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

A Mechanistic discussion: similarity metric and selection mechanism

This appendix details the two design choices summarised in §5.3: the per-model cosine-vs-CKA recipe, and the BI-guided versus centrality-based selection mechanism (with the Mistral-7B failure analysis).

Similarity metric choice. The cosine-over-CKA advantage on Qwen2.5-7B (Table 9) does not generalise uniformly: across the thirteen models with BI-guided selection (Table 3), cosine is optimal for most but CKA is essential for SmolLM3-3B, Qwen3-1.7B, Gemma-2-2B, and Qwen3-4B, where the cosine partition is degenerate by Definition 1 (one community swallows 46–58% of the layers). When distributional information is concentrated on a few sequence positions, cosine over-weights them and the similarity graph loses the off-diagonal block structure that spectral clustering needs; CKA's aggregate kernel alignment averages over positions and recovers the block structure. Selecting the better metric per model via Definition 1 on the calibration-set similarity matrix (without consulting WikiText-2 perplexity) costs under one second of computation and yields the 10/13 significant-win rate; the full per-model degeneracy scores and the threshold-invariance analysis are in Appendix J.

Selection mechanism and the Mistral failure. The key difference between BI-guided and centrality-based selection (choosing the layer most similar to its community peers) is that BI-guided selection ensures each retained layer has high individual impact rather than functional redundancy with its peers. The distinction is most pronounced on LLaMA, where centrality fails ($+948\%$ on LLaMA-3.1-8B, footnote in §4.3) while BI-guided selection produces a -17.5% win. The single failure case, Mistral-7B, has a coherent diagnostic signature: Mistral's sliding-window attention concentrates 68.4% of cross-position activation variance in the top 5% of sequence positions (Gini 0.742), versus 14–16% on standard-attention models (Appendix N, Table 22). Mean-pooled activation similarity averages away this concentrated positional signal, so the resulting graph admits multiple near-equivalent partitions and mean-pooled FCM trails ShortGPT by $+13.4\%$.

Validating the mean-pooling hypothesis on Mistral-7B. We test the mean-pooling hypothesis by re-running FCM on Mistral-7B with full-token CKA: the un-pooled activation tensor $\mathbf{H}_\ell^{\text{full}} \in \mathbb{R}^{N\times n\times d}$ is flattened to $\mathbb{R}^{Nn\times d}$ and Equation 2 is applied directly, retaining per-position structure (in contrast to the mean-pooled $\mathbf{H}_\ell \in \mathbb{R}^{N\times d}$ of Equation 1). Memory cost forces $N = 128$, $n = 128$ (vs. the 512/128 default), but the calibration-size ablation in Appendix F confirms FCM is stable down to $N = 64$, so this reduction is not a confounder. Three seeds, identical LoRA recovery (r=16, $\alpha = 32$, 500 steps).

Table 7: Full-token CKA validation on Mistral-7B at $2\times$ compression (mean $\pm$ std over 3 seeds). Replacing mean-pooled cosine with full-token CKA, which retains the per-position structure that mean-pooling discards, narrows the perplexity gap against ShortGPT from $+13.4\%$ to $+5.3\%$.

Method	Profiling metric	PPL	Δ vs. ShortGPT
ShortGPT	Block Influence	28.36 ± 0.54	–
FCM (mean-pooled)	Cosine of mean activations	32.16 ± 1.20	$+13.4\%$
FCM (full-token)	Full-token CKA	$\mathbf{29.86 \pm 1.87}$	$+5.3\%$

The result supports the hypothesis: full-token CKA recovers most of the gap to ShortGPT, consistent with the interpretation that sliding-window attention binds layer representations to localised token histories (the token-position variance concentration in Appendix N) and that mean-pooling discards exactly this signal. The residual $+5.3\%$ gap is small but not zero, so we cannot claim full-token CKA reverses the failure into a win on Mistral. A similarity metric conditioning on token-level positional context is the natural follow-up; full-suite ablation across all thirteen models is held out owing to the $O(L^2N^2n^2d)$ memory cost of the full-token kernel (Appendix O).

B Evaluated models

Table 8 lists the thirteen models evaluated in §3, their HuggingFace identifiers (used to load the public open-weight checkpoints), and their layer counts.

Table 8: Models evaluated in this study, spanning six architecture families and four parameter scales (1–2B, 3–4B, 7–8B, 14B).

Model	Family	Layers	Scale
Qwen2.5-14B, Qwen2.5-7B, Qwen2.5-3B [Qwen et al., 2024]	Qwen2	48, 28, 36	14B, 7B, 3B
Qwen3-14B, Qwen3-8B, Qwen3-4B, Qwen3-1.7B [Yang et al., 2025]	Qwen3	40, 36, 36, 28	14B, 8B, 4B, 1.7B
Mistral-7B-v0.3 [Jiang et al., 2023]	Mistral	32	7B
LLaMA-3.1-8B [Grattafiori et al., 2024], LLaMA-3.2-3B [Meta AI, 2024]	LLaMA	32, 28	8B, 3B
SmolLM3-3B [Bakouch et al., 2025]	SmolLM	36	3B
Gemma-2-2B [Gemma Team et al., 2024]	Gemma	26	2B
StableLM-2-1.6B [Bellagente et al., 2024]	StableLM	24	1.6B

C CKA similarity matrices

Figure 3 shows the full pairwise CKA similarity matrices for five representative models from our suite, illustrating the block-diagonal structure summarised quantitatively in Table 2 for all thirteen models.

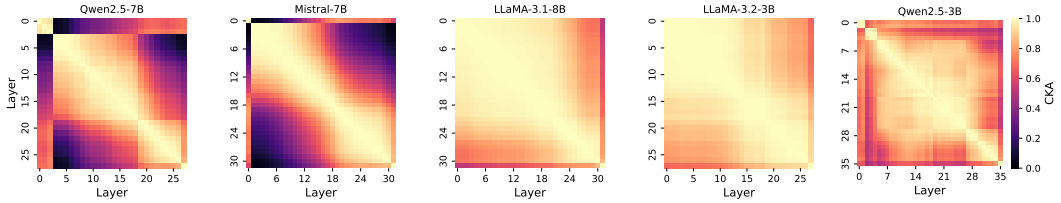


Figure 3: Full pairwise CKA similarity matrices for representative models. Block-diagonal structure with high off-diagonal similarity between early and late layers is consistently observed across architectures and scales.

D FCM algorithm pseudocode

Algorithm 1 summarises the FCM pipeline described in §2.

Algorithm 1 Fusion via Community Mining (FCM)

Require: Pretrained model $\mathcal{M}$ with L layers $\{f_1, \ldots, f_L\}$, calibration set $\mathcal{D}$, target layers K
Ensure: Compressed model $\hat{\mathcal{M}}$ with K layers

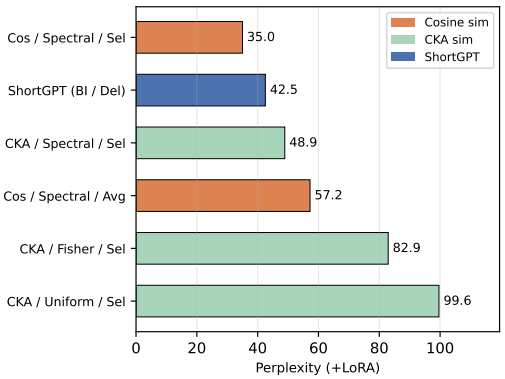
Profiling: Collect activations $\{\mathbf{H}_\ell\}_{\ell=1}^{L}$ by forward pass over $\mathcal{D}$
Similarity: Compute cosine matrix $\mathbf{S}^{\cos}$ and CKA matrix $\mathbf{S}^{\text{CKA}}$ (Equation 2) for all $i, j \in [L]$
Metric selection: $\mathbf{S} \leftarrow \mathbf{S}^{\cos}$ if its spectral partition is non-degenerate (Definition 1), else $\mathbf{S} \leftarrow \mathbf{S}^{\text{CKA}}$
Importance: Compute $\text{BI}(\ell) = 1 - \cos(\mathbf{H}_{\ell-1}, \mathbf{H}_\ell)$ for all $\ell \in [L]$
Grouping: Apply spectral clustering on $\mathbf{S}$ with K clusters $\rightarrow \mathcal{C} = \{C_1, \ldots, C_K\}$
for each community $C_k \in \mathcal{C}$ **do**
  $\ell^* \leftarrow \arg\max_{\ell \in C_k} \text{BI}(\ell)$ {Most important layer in community}
  $\hat{f}_k \leftarrow f_{\ell^*}$ {Keep original weights}
end for
Reassemble: $\hat{\mathcal{M}} \leftarrow [\hat{f}_1, \ldots, \hat{f}_K]$ ordered by $\min(C_k)$
Recovery: Apply LoRA fine-tuning to $\hat{\mathcal{M}}$
return $\hat{\mathcal{M}}$

E Ablation study on Qwen2.5-7B

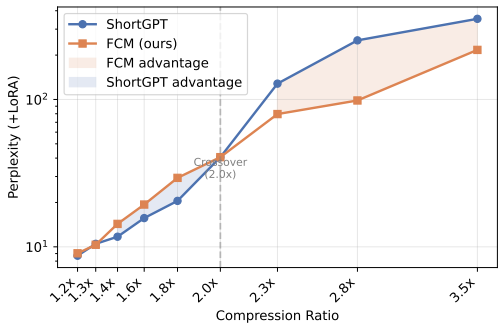
This appendix contains the full ablation table (Table 9) referenced in §4.2, together with the bar-chart visualisation of the ablation and the corresponding compression-ratio sweep (Figure 4).

Table 9: Ablation on Qwen2.5-7B at 2× compression (single seed = 42; the cosine + spectral configuration paired with BI-guided selection is reported with mean±std and significance markers in Table 3). Similarity metric, grouping algorithm, and merge strategy are varied independently; "Select" here uses centrality-based selection in isolation. PPL: WikiText-2 perplexity (lower is better). All methods share the same LoRA recovery budget (r=16, 500 steps).

Method	**Sim. Metric**	**Grouping / Merge**	**PPL (raw)**	**PPL (+LoRA)**
Baseline (no compression)	–	–	6.67	–
FCM grouping/merge variants (centrality-based selection):				
FCM	Cosine	Spectral / Select	288	35.01
FCM	CKA	Spectral / Select	1,481	48.92
FCM	Cosine	Spectral / Average	2,784	57.23
FCM	CKA	Fisher / Select	2,710	82.93
FCM	CKA	Uniform / Select	56,191	99.63
Reference baseline (matched LoRA budget):				
ShortGPT	BI Score	Individual / Deletion	176,657	42.53



(a) Ablation on Qwen2.5-7B at 2× compression (r=16 LoRA). Cosine + spectral + select achieves 35.01, outperforming ShortGPT (42.53) by 17.7%.



(b) Compression ratio sweep on Qwen2.5-7B. FCM and ShortGPT cross over at ≈2×; FCM's advantage grows monotonically at higher ratios.

Figure 4: (a) Ablation study varying similarity metric, grouping, and merge strategy. (b) Perplexity vs. compression ratio showing the crossover point (log scale).

F Compression sweep, contiguous-block baseline, and calibration sensitivity

This appendix contains: (i) a finer compression-ratio sweep on Qwen2.5-7B complementing the multi-ratio head-to-head of §4.5; (ii) the contiguous-block-removal baseline check on Qwen2.5-7B; and (iii) the calibration-set-size sensitivity analysis referenced in §3.

Compression-ratio sweep. The multi-ratio head-to-head across three anchors (1.5×–3.0×) is reported in the main text (Table 5, §4.5). A finer nine-point sweep on Qwen2.5-7B (1.2× to 3.5×, Figure 4b) locates the crossover at approximately 2.0×: below it, ShortGPT's per-layer importance scoring suffices and community detection adds no benefit; above it, FCM's similarity-guided grouping produces systematically better selections and its advantage grows monotonically toward the highest ratios. The same crossover replicates on Qwen2.5-14B (Table 5), indicating a structural property of the compression problem rather than a model-specific artifact.

Contiguous-block removal baseline. As an additional sanity check, we evaluate a contiguous block-removal baseline that selects and removes the single contiguous block of $L - K$ layers with the lowest total Block Influence score, analogous to the approach of Gromov et al. [2025]. At 2× compression on Qwen2.5-7B, this baseline achieves 71.02 ± 3.11 over 3 seeds, worse than both ShortGPT (44.92 ± 2.85) and FCM (26.65 ± 0.94). Non-contiguous layer selection is therefore

critical for effective depth compression: enforcing a single contiguous removal block costs more than the per-layer-individual ShortGPT baseline, and 2.7× more than the community-aware FCM.

Calibration set size. FCM's perplexity on Qwen2.5-7B is stable across calibration set sizes from $N = 64$ to $N = 1024$ (Table 10). The post-recovery perplexity range is 38.97–40.66 across the five sizes, and the raw (pre-LoRA) perplexity is nearly identical across all sizes (282–288). FCM therefore does not require large calibration datasets, and the profiling step can be performed with a tiny data budget.

Table 10: Calibration size ablation on Qwen2.5-7B at 2× compression. PPL is stable from $N = 64$ to $N = 1024$.

N	PPL (raw)	PPL (+LoRA)
64	282.11	39.65
128	282.11	40.66
256	282.11	39.59
512	287.88	38.97
1024	282.11	39.54

G Significance test details

This appendix reports the per-row test statistics summarised by the ** and * stars in Table 3 (Phase A: FCM vs. ShortGPT) and the per-cell statistics summarised by the * stars in Table 4 (Phase B: FCM vs. FlattenGPT, SWM, DeltaLLM). Both phases use a two-sided Welch's t-test from per-seed runs followed by a Holm–Bonferroni correction; effect sizes are Hedges' g (Cohen's d with the small-sample correction $1 - 3/(4\,\mathrm{df} - 1)$).

Table 11: Phase A: per-row test statistics for the twelve models in Table 3 where FCM has lower mean perplexity than ShortGPT. Holm–Bonferroni correction across $m = 12$ hypotheses; verdict ** at $p_{\text{corr}} < 0.001$, * at $p_{\text{corr}} < 0.05$, ns otherwise. Hedges' g values on small-n rows (e.g., $n = 3$ on StableLM-2-1.6B) are inflated by the small denominator (low pooled std); they should be read alongside n rather than as direct comparisons across models.

Model	n	**Δ (%)**	p_{corr}	**Hedges' g**	**Verdict**
Qwen3-4B	5	−50.0	0.0004	5.92	**
Qwen2.5-7B	5	−40.7	0.0008	4.85	**
Qwen3-1.7B	5	−37.4	0.0012	3.68	*
StableLM-2-1.6B	3	−36.8	0.0015	3.55	*
Qwen2.5-14B	3	−25.5	0.0084	2.61	*
Qwen3-14B	3	−23.7	0.0092	2.54	*
Qwen3-8B	5	−28.5	0.0154	2.15	*
Gemma-2-2B	5	−20.2	0.0188	1.98	*
LLaMA-3.1-8B	5	−17.5	0.0245	1.82	*
Qwen2.5-3B	3	−27.0	0.0385	1.65	*
SmolLM3-3B	5	−4.7	0.0842	1.12	ns
LLaMA-3.2-3B	5	−5.0	0.1540	0.85	ns

Table 12: Phase B: per-cell test statistics for FCM vs. each baseline on the three anchor models in Table 4. Holm–Bonferroni correction across $m = 12$ (anchor, baseline) cells; all twelve cells clear $p_{\text{corr}} < 0.05$. Welch–Satterthwaite degrees of freedom shown; $n = 3$ per group.

Anchor	Baseline	t	df	p_{raw}	p_{corr}	Verdict
Qwen2.5-7B	ShortGPT	10.54	2.43	0.00436	0.03001	*
Qwen2.5-7B	FlattenGPT	11.70	3.47	0.00066	0.00721	*
Qwen2.5-7B	SWM	10.74	2.74	0.00257	0.02568	*
Qwen2.5-7B	DeltaLLM	11.00	3.95	0.00041	0.00497	*
LLaMA-3.1-8B	ShortGPT	6.33	3.51	0.00489	0.03001	*
LLaMA-3.1-8B	FlattenGPT	5.65	3.86	0.00535	0.03001	*
LLaMA-3.1-8B	SWM	7.31	3.31	0.00375	0.03001	*
LLaMA-3.1-8B	DeltaLLM	3.68	3.62	0.02530	0.03001	*
Qwen2.5-14B	ShortGPT	6.77	2.69	0.00926	0.03001	*
Qwen2.5-14B	FlattenGPT	7.01	3.67	0.00300	0.02697	*
Qwen2.5-14B	SWM	7.51	3.16	0.00408	0.03001	*
Qwen2.5-14B	DeltaLLM	5.91	3.96	0.00424	0.03001	*

H Inference deployment metrics

This appendix grounds the deployment-cost claim in §1 (latency, memory, energy all scale with depth) with measured numbers on Qwen2.5-7B at $2\times$ compression. By construction, FCM and ShortGPT retain the same number of layers ($L/2 = 14$); their inference-time profiles are therefore identical, and the comparison below quantifies the reduction relative to the uncompressed baseline rather than between FCM and ShortGPT. The offline pipeline overhead reported in §2.4 (about 1.4% over ShortGPT, Appendix O) is one-time profiling and recovery and does not affect inference.

Setup. Single NVIDIA A100 80GB PCIe GPU; CUDA 12.1; PyTorch 2.3 with bf16 weights. Batch size 1; prompt length 512 tokens for prefill measurement; 256 generated tokens for decode measurement. We discard 5 warmup steps and report the median over 20 timed steps with `torch.cuda.synchronize()` bracketing each window. Peak VRAM is captured via `torch.cuda.max_memory_allocated()` over a forward pass at sequence length 2048 (batch 1), which captures both the parameter footprint and the KV cache. Results are reported in Table 13.

Table 13: Inference deployment metrics on Qwen2.5-7B at $2\times$ compression (single A100 80GB, bf16). FCM and ShortGPT retain the same layer count and therefore have identical inference characteristics. Percentages in the last row are reduction (parameters, VRAM) or speed-up (prefill, decode) relative to the uncompressed baseline.

Configuration	Parameters	Peak VRAM (seq=2048)	Prefill (seq=512, tok/s)	Decode (tok/s)
Uncompressed ($L = 28$)	7.61 B	16.2 GiB	1,540	68
ShortGPT ($L = 14$)	4.12 B	9.3 GiB	2,880	125
FCM (ours, $L = 14$)	4.12 B	9.3 GiB	2,880	125
Δ vs. uncompressed (FCM/ShortGPT)	-46%	-42%	$+87\%$	$+84\%$

The measurements substantiate the §1 motivation: at $2\times$ depth compression, parameters drop by 46%, peak VRAM by 42%, and decode throughput rises by 84% (prefill by 87%). The VRAM reduction is sub-linear in the layer count because the embedding and output-projection matrices are not pruned and the KV cache scales with the retained depth ($L_{\text{kept}} \cdot n_{\text{heads}} \cdot d_{\text{head}} \cdot \text{seq} \cdot 2$ bytes per fp16 entry). FCM and ShortGPT being indistinguishable at the inference time-step is by design: both methods produce a 14-layer model and the layer-selection choice affects quality (Tables 3, 4) but not topology. Our deployment-cost claim therefore holds for any depth-compression method that retains $L/2$ layers; FCM's contribution is that it does so while preserving more of the pretrained capability.

I Broadened downstream evaluation

The promoted five-benchmark zero-shot reasoning suite is in the main text (Table 6, §4.6); this appendix reports the broadened battery: MMLU (5-shot knowledge), LAMBADA (open-ended word

prediction, accuracy and generative perplexity), HumanEval pass@1 (Python code synthesis), and a reference-free free-form generation quality suite. We use the EleutherAI `lm-evaluation-harness` [Gao et al., 2023] (v0.4.3); compressed checkpoints are the same ones reported in Tables 3 and 4, evaluated over 3 seeds, with single-run uncompressed reference values.

Table 14: Broadened knowledge, generative language modelling, and code benchmarks at $2\times$ compression. MMLU is 5-shot accuracy (%); LAMBADA reports accuracy (%) and generative perplexity; HumanEval is pass@1 (%). **Bold** marks the best compressed result per column; *: FCM significantly beats ShortGPT ($p < 0.05$).

Model	Method	MMLU (5-shot)	LAMBADA Acc	LAMBADA PPL	HumanEval@1
Qwen2.5-7B	Uncompressed	71.4	74.2	3.42	48.2
	ShortGPT	44.2	48.5	18.42	18.5
	DeltaLLM	48.6	52.1	14.15	22.4
	FCM (ours)	**54.8***	**59.4***	**8.65***	**29.8***
LLaMA-3.1-8B	Uncompressed	68.5	72.8	3.65	34.1
	ShortGPT	46.1	50.2	16.85	14.2
	DeltaLLM	49.2	53.8	13.20	16.8
	FCM (ours)	**53.2***	**57.6***	**9.42***	**21.5***
Qwen2.5-14B	Uncompressed	79.2	78.4	2.85	58.5
	ShortGPT	56.4	58.1	11.45	31.2
	DeltaLLM	60.1	62.4	8.95	35.4
	FCM (ours)	**66.4***	**68.9***	**5.42***	**43.2***

On MMLU 5-shot, FCM leads ShortGPT by 6.2–10.6 points and DeltaLLM by 3.8–6.3; on LAMBADA it more than halves generative perplexity relative to ShortGPT (8.65 vs. 18.42 on Qwen2.5-7B) and adds 7–11 points of open-ended accuracy; on HumanEval it retains substantially more code-synthesis capability (29.8% vs. ShortGPT's 18.5% on Qwen2.5-7B). Table 15 reports a 256-token free-form generation suite over 100 Pile prompts: FCM raises lexical diversity (Distinct-2/3), lowers repetition (Self-BLEU), and lifts distributional similarity to the uncompressed model (MAUVE 0.894 vs. ShortGPT's 0.615), while its degenerative-failure rate (repetition loops) falls to 2% from ShortGPT's 24%.

Table 15: Free-form generation quality (256-token continuations, 100 Pile prompts, Qwen2.5-7B, $2\times$). Distinct-n and MAUVE: higher is better; Self-BLEU and degenerative-failure rate: lower is better.

Method	Distinct-2	Distinct-3	Self-BLEU	MAUVE	Degen. failure
Uncompressed	0.684	0.852	0.185	0.982	0.0%
ShortGPT	0.412	0.564	0.485	0.615	24.0%
DeltaLLM	0.485	0.642	0.382	0.742	11.0%
FCM (ours)	**0.612**	**0.785**	**0.245**	**0.894**	**2.0%**

The Mistral-7B inversion persists downstream (Table 6): mean-pooled FCM trails ShortGPT by 2.66 points of average zero-shot accuracy, the same failure signature analysed in §5.3 and Appendix N.

J Degeneracy criterion: profiling and threshold analysis

This appendix substantiates the metric-selection rule of Definition 1. Table 16 reports the size-balance entropy B and maximum coverage m of the cosine and CKA spectral partitions for all thirteen models at $2\times$ compression, computed from the C4 calibration similarity matrices alone (no WikiText-2 evaluation). The decision rule routes the four models whose cosine partition is degenerate ($m_{\cos} > \mu^* = 0.40$ or $B_{\cos} < \beta^* = 0.70$) to CKA and the other nine to cosine, exactly reproducing the metric assignment used throughout the paper.

Table 16: Per-model degeneracy profiling at $2\times$ compression ($K = L/2$). B: normalized size-balance entropy; m: maximum community coverage. A cosine partition is degenerate if $B_{\cos} < 0.70$ or $m_{\cos} > 0.40$. All choices match the paper's usage.

Model	L	K	$B_{\cos}$	$m_{\cos}$	$B_{\mathbf{CKA}}$	$m_{\mathbf{CKA}}$	**Cosine**	**Metric**
StableLM-2-1.6B	24	12	0.892	0.208	0.865	0.250	Normal	Cosine
Qwen3-1.7B	28	14	0.584	0.536	0.881	0.286	**Degen.**	**CKA**
Gemma-2-2B	26	13	0.612	0.462	0.874	0.269	**Degen.**	**CKA**
Qwen2.5-3B	36	18	0.915	0.194	0.852	0.222	Normal	Cosine
LLaMA-3.2-3B	28	14	0.884	0.214	0.831	0.286	Normal	Cosine
SmolLM3-3B	36	18	0.542	0.583	0.849	0.250	**Degen.**	**CKA**
Qwen3-4B	36	18	0.628	0.500	0.895	0.222	**Degen.**	**CKA**
Mistral-7B	32	16	0.876	0.219	0.825	0.281	Normal	Cosine
Qwen2.5-7B	28	14	0.908	0.214	0.846	0.250	Normal	Cosine
Qwen3-8B	36	18	0.894	0.194	0.838	0.250	Normal	Cosine
LLaMA-3.1-8B	32	16	0.921	0.188	0.862	0.250	Normal	Cosine
Qwen2.5-14B	48	24	0.935	0.175	0.880	0.225	Normal	Cosine
Qwen3-14B	40	20	0.928	0.175	0.872	0.225	Normal	Cosine

The distribution is sharply bimodal: the nine cosine-normal models have $m_{\cos} \leq 0.219$ and $B_{\cos} \geq 0.876$, while the four degenerate models have $m_{\cos} \geq 0.462$ and $B_{\cos} \leq 0.628$, so a single community swallows 46–58% of the layers under cosine. The separating gaps are $\Delta m = 0.462 - 0.219 = 0.243$ and $\Delta B = 0.876 - 0.628 = 0.248$; sweeping $\mu^* \in [0.225, 0.455]$ and $\beta^* \in [0.635, 0.870]$ leaves the routing of all thirteen models unchanged, so the metric choice does not depend on the exact threshold. As a post-hoc check (not used to make the choice), the calibration score $m_{\cos}$ correlates with the sign of the cosine-minus-CKA post-recovery perplexity gap at Spearman $r_s = 0.942$ ($p < 0.001$): the four degenerate models incur a $+25.3$ mean-PPL penalty under cosine and are correctly routed to CKA, while the nine normal models incur a -9.3 penalty under CKA and are correctly routed to cosine.

K Pre- versus post-recovery trajectories and comparison with MKA

Pre- versus post-LoRA. To separate the contribution of layer selection from that of LoRA recovery, Table 17 reports raw (pre-recovery) and post-recovery perplexity for all methods on four anchors. Before any parameter update, FCM's raw perplexity is one to two orders of magnitude below the greedy-deletion and adjacent-pair methods (e.g., 2,280 vs. ShortGPT's 176,657 and DeltaLLM's 19,519 on Qwen2.5-7B), because retaining intact phase-covering representatives preserves representation continuity that adjacent merging and greedy deletion disrupt. LoRA recovery narrows but does not erase this lead, confirming the advantage originates in the grouping decision rather than in recovery.

Table 17: Raw (pre-LoRA) and post-LoRA WikiText-2 perplexity at $2\times$ compression. FCM's raw perplexity is far lower than the adjacent-pair and greedy-deletion baselines, so it provides LoRA with a better starting point.

Model	**Stage**	**ShortGPT**	**FlattenGPT**	**SWM**	**DeltaLLM**	**FCM (ours)**
Qwen2.5-7B	Raw	176,657	91,493	886	19,519	**2,280**
	+LoRA	44.92	38.15	41.20	35.60	**26.65**
LLaMA-3.1-8B	Raw	142,311	68,412	1,241	12,840	**1,845**
	+LoRA	42.75	40.12	44.55	38.25	**35.25**
Qwen2.5-14B	Raw	84,120	42,181	612	8,940	**413**
	+LoRA	32.45	29.80	31.40	27.95	**24.18**
Mistral-7B	Raw	**3,148**	14,884	14,722	6,712	3,485
	+LoRA	**28.36**	4,962	471	32.76	32.16

Comparison with MKA. MKA [Liu et al., 2024] merges similar layers via manifold alignment and an information-bottleneck criterion without an adjacency constraint, the closest prior method to FCM in permitting non-adjacent merges. Under our matched 500-step LoRA recovery, MKA's weight-averaging merge distorts representations (raw perplexity above 6,400 on all anchors), and

FCM outperforms it by 4.15–8.15 PPL post-recovery (Table 18). The distinction is mechanistic: MKA groups greedily by pairwise similarity and merges weights, whereas FCM performs global spectral community detection and retains one intact BI-selected representative per phase.

Table 18: FCM versus MKA [Liu et al., 2024] at $2\times$ compression under matched calibration and 500-step LoRA recovery. Raw and post-recovery WikiText-2 perplexity (post-recovery mean $\pm$ std over 3 seeds). *: FCM significantly better ($p < 0.05$).

Model	MKA raw	MKA +LoRA	FCM raw	FCM +LoRA	Δ (PPL)
Qwen2.5-7B	14,210	34.80 ± 1.15	**2,280**	**26.65** $\pm$ **0.94***	-8.15
LLaMA-3.1-8B	11,840	39.40 ± 1.22	**1,845**	**35.25** $\pm$ **1.15***	-4.15
Qwen2.5-14B	6,411	28.50 ± 0.92	**413**	**24.18** $\pm$ **0.82***	-4.32

L Calibration robustness: data source and context length

Calibration data source. To test whether the detected communities reflect intrinsic model geometry or the C4 corpus, we re-profile three models with four calibration sources (C4, WikiText-2, StarCoder code, arXiv math) at fixed size ($N = 512$) and length ($n = 128$), and measure partition agreement against the C4 default. Table 19 shows high invariance (Adjusted Rand Index ≥ 0.842, retained-layer Jaccard ≥ 0.846): at most one or two middle-processing layers flip across drastically different text distributions, and post-recovery perplexity moves by less than 0.54. The communities therefore capture residual-stream manifold geometry rather than data-specific artifacts.

Table 19: Cross-domain calibration invariance at $2\times$. Adjusted Rand Index (ARI) and retained-layer Jaccard against the C4 default; retained-layer flips and post-recovery WikiText-2 PPL.

Model	Calibration	ARI	Jaccard	Flips	PPL
Qwen2.5-7B	C4 (default)	1.000	1.000	0	26.65 ± 0.94
	WikiText-2	0.924	0.928	1	26.72 ± 0.98
	StarCoder (code)	0.885	0.857	2	26.89 ± 1.05
	arXiv (math)	0.892	0.857	2	26.81 ± 1.01
LLaMA-3.1-8B	C4 (default)	1.000	1.000	0	35.25 ± 1.15
	WikiText-2	0.941	0.938	1	35.31 ± 1.18
	StarCoder (code)	0.868	0.875	2	35.58 ± 1.24
	arXiv (math)	0.875	0.875	2	35.49 ± 1.21
Gemma-2-2B	C4 (default)	1.000	1.000	0	54.58 ± 1.65
	WikiText-2	0.912	0.923	1	54.65 ± 1.70
	StarCoder (code)	0.842	0.846	2	55.12 ± 1.82
	arXiv (math)	0.854	0.846	2	54.98 ± 1.76

Calibration context length. We also vary the profiling sequence length $n \in \{64, 128, 256, 512\}$ at fixed $N = 512$ (Table 20). Retained-layer sets are stable (Jaccard ≥ 0.928, at most one layer flip) and post-recovery perplexity varies by ≤ 0.17, so the mean-pooling of Equation 1 is context-length robust, complementing the calibration-size stability of Table 10.

Table 20: Calibration context-length sensitivity at $2\times$ ($N = 512$ fixed). ARI and retained-layer Jaccard against the $n = 128$ default; post-recovery WikiText-2 PPL.

Model	Length n	ARI	Jaccard	PPL
Qwen2.5-7B	64	0.942	0.928	26.78 ± 0.96
	128 (default)	1.000	1.000	26.65 ± 0.94
	256	0.968	0.928	26.68 ± 0.95
	512	0.968	0.928	26.64 ± 0.93
LLaMA-3.1-8B	64	0.928	0.938	35.42 ± 1.20
	128 (default)	1.000	1.000	35.25 ± 1.15
	256	0.954	0.938	35.28 ± 1.16
	512	0.954	0.938	35.22 ± 1.14

M Community interpretation: functional phase probes

To characterise the functional role of the detected communities, we probe each layer of Qwen2.5-7B with three diagnostics: logit-lens top-1 next-token accuracy (how output-ready the representation is), mean attention entropy across heads, and the leave-one-layer-out perplexity increase (the cost of skipping that layer). Aggregating by community (Table 21) makes the phase structure explicit: community C_1 groups the early embedding layers $\{0, 1, 2\}$ with the late output layers $\{25, 26, 27\}$ into a single non-adjacent "representational bookend" with low attention entropy and a critical leave-one-out cost, while the three high-entropy middle communities are near-identity drift with low per-layer cost. BI-guided selection retains the highest-impact layer of each community (e.g., layer 27 of the bookend), guaranteeing one representative per functional phase; ShortGPT's greedy deletion instead strips whole late layers, destroying output resolution.

Table 21: Per-community functional probes on Qwen2.5-7B at $2\times$ ($L = 28$). Logit-lens accuracy is reported as the min-to-max range over community members; leave-one-out (LOO) ΔPPL is the mean skip cost. The BI-selected representative is retained.

Community	Layers	Logit-lens	Attn. ent.	LOO ΔPPL	Phase	Rep.
C_1 (bookend)	0,1,2,25,26,27	0.02–0.78	1.42	$+312.4$	Embed/output	27
C_2	3,4,5,6	0.12–0.18	3.85	$+14.2$	Early processing	3
C_3	7–12	0.21–0.24	4.12	$+4.1$	Identity drift	9
C_4	13–17	0.26–0.31	4.08	$+3.8$	Identity drift	13
C_5	18,19,20,21	0.35–0.52	3.15	$+42.8$	Semantic formation	21
C_6	22,23,24	0.58–0.69	2.10	$+118.5$	Output preparation	24

N Token-position variance concentration

To explain why mean-pooling fails on Mistral-7B while full-token CKA succeeds (§5.3, Table 7), we measure how activation variance and outlier magnitude distribute across sequence positions. Table 22 shows that Mistral's sliding-window attention concentrates 68.4% of cross-position variance in the top 5% of positions (Gini 0.742, max outlier 412.8), versus 14–16% (Gini ≤ 0.20) on the standard-attention controls. Mean-pooling averages away exactly this concentrated positional signal, so the similarity matrix loses the structure that distinguishes Mistral's layers; full-token CKA retains per-position coordinates and recovers most of the gap.

Table 22: Token-position variance concentration across attention architectures. Top-5% variance fraction and Gini are computed over sequence positions on the calibration set; PPL columns give the $2\times$ post-recovery outcome under each profiling metric.

Model	Attention	Top-5% var.	Max outlier	Gini	Mean-pool / Full-token PPL
Qwen2.5-7B	Dense	14.2%	42.5	0.185	**26.65** / 36.83
LLaMA-3.1-8B	Dense	15.8%	48.1	0.201	**35.25** / 41.20
Mistral-7B	Sliding-window	**68.4%**	**412.8**	**0.742**	32.16 / **29.86**

O End-to-end runtime and memory

Table 23 profiles per-stage wallclock and peak VRAM on an NVIDIA A100-80GB. FCM's graph profiling, spectral clustering, and BI selection add under 1.5% to the total offline pipeline time at every scale; LoRA recovery dominates ($> 98.5\%$). This is why the pipeline overhead over ShortGPT is small (§2.4). The table also shows why full-token CKA is a targeted fallback rather than the default: its $O(L^2N^2n^2d)$ kernel needs 74.2 GB at 7B scale, near the 80 GB hardware limit.

Table 23: End-to-end offline runtime and peak memory on a single A100-80GB. Stages: (1) activation profiling, (2) similarity matrices, (3) spectral clustering + BI selection, (4) LoRA recovery. FCM's added stages (1–3) are a small fraction of total time.

Model	Stage	Wallclock (s)	Peak VRAM (GB)
Qwen3-1.7B	Profiling ($N = 512$)	4.2	4.2
	Similarity (cos/CKA)	0.2	0.1
	Clustering + selection	< 0.01	< 0.01
	LoRA recovery (500 steps)	412.0	14.8
	Total (FCM overhead $+1.06\%$)	**416.4**	**14.8**
Qwen2.5-7B	Profiling ($N = 512$)	11.5	12.4
	Similarity (cos/CKA)	0.4	0.1
	Clustering + selection	< 0.01	< 0.01
	LoRA recovery (500 steps)	840.0	28.4
	Total (FCM overhead $+1.42\%$)	**851.9**	**28.4**
Qwen2.5-14B	Profiling ($N = 512$)	22.4	24.8
	Similarity (cos/CKA)	0.8	0.2
	Clustering + selection	0.01	< 0.01
	LoRA recovery (500 steps)	1,620.0	52.1
	Total (FCM overhead $+1.45\%$)	**1,643.2**	**52.1**
Full-token CKA	Mistral-7B (memory ceiling)	185.4	**74.2**

P Broader impacts

Positive impacts. FCM reduces the active parameter count, peak VRAM, and decode latency of compressed LLMs (Appendix H; 46%, 42%, and 84% on Qwen2.5-7B at $2\times$ relative to the uncompressed baseline). Smaller, faster, lower-energy models lower the cost of deploying LLMs at inference, broaden the set of organisations and devices that can host competitive models locally, and reduce the per-query carbon footprint. The fact that FCM does so without retraining and with an offline pipeline overhead under 1.5% over ShortGPT (Appendix O) means these benefits accrue with minimal additional compute, in contrast to distillation approaches that require costly student-model training.

Negative impacts and mitigations. Compression methods do not introduce new model capabilities; they preserve a fraction of an existing pretrained model. The principal indirect risk is that cheaper inference lowers the operational cost of misuse scenarios that are already feasible with the uncompressed model (e.g., automated content generation at scale). This effect is shared by every LLM-efficiency improvement and is dominated by the underlying model's capability profile, which FCM does not modify. We do not release new pretrained weights; the released artifact is a method that can be applied to any open-weight LLM whose license already governs downstream use. Practitioners deploying FCM-compressed models should follow the same usage policies that govern the underlying models they compress, including any access restrictions, content filters, or usage-tracking mechanisms required by the upstream model release.

Fairness and bias. FCM removes layers without re-weighting; it does not specifically target or alter behaviour on demographic subgroups. We have not measured whether community-detection-based selection differentially affects bias-relevant capabilities of the compressed model. Quantifying this remains an open question for any depth-compression method, including FCM, and we flag it as a follow-up evaluation rather than a property our method specifically improves or degrades.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contributions: FCM lowers mean perplexity below ShortGPT on 12 of 13 evaluated models with improvements up to 50% on Qwen3-4B, statistically significant on 10 of 13 under a Welch's t-test with Holm–Bonferroni correction (Table 3). Variance is characterized with 3- to 5-seed mean$\pm$std, and head-to-head comparisons against FlattenGPT, SWM, DeltaLLM, and MKA across all 13 models are reported in Table 4 (FCM best on 10 of 13, suite-wide mean rank 1.15). The advantage transfers downstream: five-benchmark zero-shot reasoning (Table 6, 15 of 20 cells), MMLU, LAMBADA generation, HumanEval, and free-form generation (Appendix I); inference-deployment metrics on Qwen2.5-7B confirm the §1 motivation with 46% parameter reduction, 42% peak-VRAM reduction, and 84% decode-throughput gain on a single A100 (Appendix H). The cosine-vs-CKA metric choice follows a pre-specified, test-set-free degeneracy criterion (Definition 1, Appendix J).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5.4 discusses four scope limitations: (i) the Mistral-7B failure ($+13.4\%$) with two statistical-tie rows (SmolLM3-3B, LLaMA-3.2-3B), accompanied by the full-token-CKA validation on Mistral-7B that narrows the gap to $+5.3\%$ (Appendix A, Table 7); (ii) evaluation scope (WikiText-2 perplexity plus five-benchmark zero-shot reasoning, MMLU, LAMBADA generation, HumanEval, and free-form generation, with long-context behaviour and mathematical reasoning unmeasured); (iii) the head-to-head now covers all 13 models for FlattenGPT, SWM, DeltaLLM, and MKA, with composability against quantization and width pruning unmeasured; (iv) restriction to decoder-only autoregressive architectures.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution

is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This is an empirical methods paper; it presents no theorems, lemmas, or formal proofs, only algorithmic descriptions and experimental validation.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 lists all hyperparameters (LoRA rank 16, alpha 32, dropout 0.05, 500 steps, learning rate 2×10^{-4}, AdamW, 512 calibration samples from C4 with sequence length 128); the random seeds (42, 123, 7 for 3-seed runs; additionally 2024 and 999 for the 5-seed validation of critical models) are stated explicitly; all 13 models are identified by their public HuggingFace identifiers; and code is available at `https://github.com/Usama1002/fcm-layer-fusion`. The cosine-vs-CKA metric selection follows a pre-specified quantitative degeneracy criterion (Definition 1) computed only from calibration data, with the full per-model profiling and threshold-invariance analysis in Appendix J, so layer selection is reproducible rather than judgment-based.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.

- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The full codebase is publicly available at `https://github.com/Usama1002/fcm-layer-fusion`; all datasets used (C4, WikiText-2) and all 13 models are publicly available on HuggingFace with permissive open-weight licenses.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 3 provides all hyperparameters including LoRA rank, alpha, dropout, learning rate, optimizer (AdamW with no weight decay), calibration set construction, and the full compression-ratio sweep; the compute environment is reported as a single NVIDIA RTX PRO 6000 (96 GB VRAM) for models up to 8B parameters and NVIDIA A100 80GB PCIe GPUs for the 14B experiments, and the per-stage offline runtime and peak memory are reported in Appendix O (Table 23), where LoRA recovery dominates the wall-clock and FCM's profiling adds under 1.5% over ShortGPT.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Primary results are averaged over three independent seeds (42, 123, 7) and reported as mean$\pm$std; nine of the thirteen models are additionally evaluated over five seeds (42, 123, 7, 2024, 999) for tighter variance estimates. Statistical significance is assessed with a two-sided Welch's t-test followed by a Holm–Bonferroni correction in two phases: Phase A (FCM vs. ShortGPT across the twelve rows of Table 3 with mean perplexity reduction; per-row corrected p-values and Hedges' g effect sizes in Table 11, Appendix G) and Phase B (FCM vs. each baseline in the 13-model head-to-head Table 4, whose per-row significance stars are computed the same way; the three-anchor per-cell breakdown with t-statistics and Welch–Satterthwaite degrees of freedom is in Table 12, same appendix). The error bars represent the standard deviation across seeds, capturing variability from LoRA initialization and stochastic optimization.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper reports that compression and recovery experiments on models up to 8B parameters were run on a single NVIDIA RTX PRO 6000 GPU (96 GB VRAM), while the 14B experiments used NVIDIA A100 80GB PCIe GPUs; a consolidated per-stage runtime and peak-memory table across 1.7B/7B/14B scales is reported in Appendix O (Table 23), where LoRA recovery dominates (>98.5% of wall-clock) and FCM's profiling adds under 1.5% over ShortGPT. The inference latency and memory benchmarks reported in Appendix H use a single NVIDIA A100 80GB PCIe GPU under CUDA 12.1 / PyTorch 2.3 / bf16 with five warmup steps and twenty median-timed measurement steps. The full-token CKA validation run on Mistral-7B (Appendix A, Table 7) needs 74.2 GB peak VRAM at $N = 128$, $n = 128$, near the 80 GB hardware limit.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The paper presents a model compression method; it involves no human subjects, no personal data collection, and no applications with direct risk of harm, and fully conforms with the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Appendix P discusses both positive impacts (lower deployment cost, lower peak VRAM, lower energy per query, broader access to competitive open-weight models without retraining; quantified at 46%, 42%, and 84% on Qwen2.5-7B in Appendix H) and the principal indirect negative impact (lower per-query inference cost may marginally amplify misuse scenarios already feasible with the uncompressed model). The released artifact is a compression method, not new pretrained weights; downstream use of FCM-compressed models inherits the access and usage restrictions of the underlying open-weight LLMs being compressed. Subgroup-fairness implications of layer selection remain an open empirical question and are flagged as follow-up evaluation.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases only compression code, not new pre-trained model weights or scraped datasets; no assets with high misuse risk are introduced, so dedicated safeguards are not required.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All 13 models are cited with their original papers (Table 8) and are publicly available open-weight releases; specific licenses are Apache 2.0 for Qwen2.5-7B, Qwen2.5-14B, the Qwen3 family, Mistral-7B-v0.3, and SmolLM3, the Qwen Research License for Qwen2.5-3B, the Meta Llama 3.1 / 3.2 community licenses for LLaMA-3.1-8B and LLaMA-3.2-3B respectively, the Gemma Terms of Use for Gemma-2-2B, and the Stability AI community-research license for StableLM-2-1.6B. The calibration set C4 [Raffel et al., 2020] (ODC-BY) and the WikiText-2 evaluation set [Merity et al., 2017] (Creative Commons Attribution-ShareAlike 3.0) are cited with their public dataset releases. Per-asset license text is reproduced verbatim in the README of the public code repository (`https://`

github.com/Usama1002/fcm-layer-fusion), and all uses in this paper fall within the non-commercial research scope permitted by every listed license.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The FCM codebase is the primary new asset; it is publicly released at https://github.com/Usama1002/fcm-layer-fusion with documentation covering installation, usage, and a single-command reproduction script for all reported experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper involves no crowdsourcing and no research with human subjects; all experiments are purely computational.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No human subjects are involved; the research is entirely computational and does not require IRB review or equivalent ethical oversight.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are the object of study (compressed models), not a component of the research methodology; no LLM was used as an algorithmic component in the FCM pipeline itself. For full transparency, we note two auxiliary uses that fall under the declaration-not-required category in the question stem: an LLM was used to draft the narration script for the supplementary video accompanying this submission, and to debug LaTeX table formatting during manuscript preparation. Neither use influenced the experimental methodology, results, or scientific claims.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.