

# Motif-Upcycling: Structure-Preserving Adaptation of Transformer Models

Kharki

April 2026

## Abstract

Pretrained Transformers are usually adapted by adding trainable modules around dense blocks, while the internal structure of the original computation remains implicit. We introduce *motif-upcycling*, a preliminary framework for structure-preserving adaptation of pretrained Transformer models. The central observation is that common feed-forward modules, including gated SwiGLU FFNs, can be exactly factorized along their intermediate channel axis into motif-aligned components. With neutral routing, the factorized module computes the same function as the original pretrained block at initialization.

On top of this exact decomposition, motif-upcycling adds lightweight contextual routers and motif-local low-rank interventions. We also introduce *Scale-Aware Residual Control* (SARC), an identity-preserving control motif that modulates trainable residual interventions according to their RMS scale relative to the residual stream. SARC leaves the donor residual path unchanged in its adapter-only form and yields a bounded local perturbation of the new intervention.

Preliminary Qwen-family experiments suggest that motif-upcycled modules can be inserted safely and trained with sub-0.1% trainable budgets in selected settings. In one hybrid Qwen3.5-0.8B run, 692,492 trainable parameters, or 0.092% of the model, reach a best mean validation-loss improvement of 0.458 over the frozen dense baseline at step 4000. We do not claim optimality, universal validity, or frontier-level quality. The main claim is narrower: pretrained Transformer modules can be exactly factorized into motif-structured components and adapted under explicit function-preserving and residual-scale controls.

## 1 Introduction

Pretrained language models are increasingly adapted by low-rank updates, adapters, routing layers, and sparse expert mechanisms. These methods are powerful, but they often treat the pretrained block as a dense object to be perturbed rather than as a structured computation to be exposed. This paper studies a complementary question: when a pretrained Transformer module already contains an internal decomposition, can we make that decomposition explicit, preserve the original function at initialization, and then adapt only a small structured budget?

We propose *motif-upcycling*: a structure-preserving adaptation framework for pretrained Transformers. A motif is an operator role such as projection, comparison, selection, aggregation, expansion, compression, routing, memory-like transformation, or residual iteration. Transformer blocks naturally combine these roles. An attention path can be read as

$$\text{proj} \rightarrow \text{compare} \rightarrow \text{select} \rightarrow \text{aggregate},$$

while a feed-forward path can be read as

expand  $\rightarrow$  select  $\rightarrow$  compress.

The aim is not merely to rename familiar operations. The aim is to expose a level of structure at which pretrained modules can be modified without losing their original function at step zero.

The key architectural observation is simple. In a SwiGLU-style feed-forward network, the intermediate channels can be partitioned into disjoint groups. Because the gate, activation, and Hadamard product act channel-wise before the final down-projection, the original FFN is exactly the sum of the group outputs. A neutral router therefore recovers the dense donor exactly, while contextual routing and motif-local LoRA updates provide a small trainable adaptation budget.

This paper also develops a finite motif-grammar formalism around this operation. The formal results are intended as a structural language for architecture-generated function classes, not as a universal theory of all computation. They show how bounded motif graphs represent architecture-generated functions, how local errors propagate over reachable compact sets, and how motif-specific budget curves can make uniform allocation suboptimal. We include Emergence as Coupled Budget Thresholds (ECBT) as a conditional model: if a capability depends multiplicatively on several motif effectiveness curves, then apparent threshold behavior follows under uniform budget constraints.

Finally, we introduce *Scale-Aware Residual Control* (SARC). SARC is a lightweight residual-control motif that regulates the magnitude of new trainable interventions relative to the residual stream. In its adapter-only form, it leaves the frozen donor update unchanged and applies an identity-preserving bounded scale correction only to the adapter or motif delta.

**Contributions.** This paper makes four preliminary contributions:

1. We introduce motif-upcycling, a structure-preserving adaptation framework for pretrained Transformer modules.
2. We show that SwiGLU-style feed-forward blocks can be exactly factorized into motif-aligned channel components, preserving the pretrained function under neutral routing.
3. We introduce motif-local trainable interventions and Scale-Aware Residual Control, an identity-preserving controller for regulating the magnitude of new residual interventions.
4. We provide preliminary Qwen-family experiments showing validation-loss improvements with sub-0.1% trainable parameter budgets in selected settings.

## 2 Motif grammars and practical representability

### 2.1 Atomic motif families

**Definition 2.1** (Atomic motif family). Let

$$\Phi = \{\Phi^{(1)}, \dots, \Phi^{(m)}\}$$

be a finite family of parameterized operator motifs. Each family  $\Phi^{(i)}$  consists of maps

$$\phi_{\theta}^{(i)} : \mathbb{R}^{d_{\text{in}}} \rightarrow \mathbb{R}^{d_{\text{out}}}, \quad \theta \in \Theta^{(i)},$$

with a finite-dimensional parameter vector. Each family has a parameter cost  $C_{\text{param}}$ , a computational cost  $C_{\text{comp}}$ , and optional regularity constraints.

**Definition 2.2** (Compositional motif class). Fix a finite motif library  $\Phi$ , a graph budget  $s \in \mathbb{N}$ , a local arity bound  $k \in \mathbb{N}$ , and a total parameter budget  $B > 0$ . Let  $F_{k,s,B,\Phi}$  be the class of functions  $f : \mathbb{R}^d \rightarrow \mathbb{R}^r$  for which there exists a directed acyclic graph  $G = (V, E)$  such that  $|V| \leq s$ ,  $\deg^-(v) \leq k$  for all nodes, each node is assigned an operator  $h_v \in \Phi$ ,

$$\sum_{v \in V} C_{\text{param}}(h_v) \leq B,$$

and

$$f = \text{Eval}(G, \{h_v\}_{v \in V}).$$

**Definition 2.3** (Practical function class). Let  $\mu$  be a task-dependent data distribution on  $\mathbb{R}^d$  and  $\epsilon > 0$ . Define

$$P_\epsilon(\mu; k, s, B, \Phi) = \left\{ f : \mathbb{R}^d \rightarrow \mathbb{R}^r \mid \exists \hat{f} \in F_{k,s,B,\Phi} \text{ such that } \|f - \hat{f}\|_{L^2(\mu)} \leq \epsilon \right\}.$$

A function is practically representable if it is approximable on the task distribution by a bounded-complexity motif graph.

This distribution-relative definition is important. It avoids requiring a model to approximate a function over all of  $\mathbb{R}^d$ . Instead, it asks whether the function is representable on the subset of states actually reached by the task distribution.

## 2.2 Exact and abstract grammars

The exact grammar used for representation contains the following operator families:

$$\Phi^{\text{ex}} = \{\Phi^{\text{proj}}, \Phi^{\text{compare}}, \Phi^{\text{select,exact}}, \Phi^{\text{aggregate}}, \Phi^{\text{expand}}, \Phi^{\text{compress}}, \Phi^{\text{iterate}}, \Phi^{\text{compose}}, \Phi^{\text{norm}}\}.$$

Here  $\Phi^{\text{proj}}$  contains affine maps;  $\Phi^{\text{compare}}$  contains bilinear or scaled dot-product similarity maps;  $\Phi^{\text{select,exact}}$  contains row-softmax and the pointwise nonlinearities used by the target architecture;  $\Phi^{\text{aggregate}}$  contains weighted sums and matrix products;  $\Phi^{\text{expand}}$  and  $\Phi^{\text{compress}}$  are affine maps with dimension expansion or compression;  $\Phi^{\text{iterate}}$  contains residual addition;  $\Phi^{\text{compose}}$  contains concatenation and graph composition;  $\Phi^{\text{norm}}$  contains normalization operators such as LayerNorm or RMSNorm with numerical stabilization.

The abstract grammar replaces exact nonlinearities by a universal select family. For a bounded continuous nonconstant activation  $\rho$ , define

$$\Phi_\rho^{\text{select}} = \left\{ x \mapsto \sum_{j=1}^N a_j \rho(w_j^\top x + b_j) + c \right\}_{N < \infty}.$$

Hornik’s universal approximation theorem gives density on compact sets and in  $L^p(\mu)$  for finite input measures [2].

**Assumption 2.4** (Local regularity). For every motif family, every compact input set  $K$ , and every compact parameter set  $\Theta_0$ , there exist constants  $L_x, L_\theta < \infty$  such that

$$\|\phi_{\theta_1}(x_1) - \phi_{\theta_2}(x_2)\| \leq L_x \|x_1 - x_2\| + L_\theta \|\theta_1 - \theta_2\|$$

for all  $x_1, x_2 \in K$  and  $\theta_1, \theta_2 \in \Theta_0$ .

This is local rather than global. Global Lipschitz continuity in parameters fails even for affine maps unless input norms are bounded, but local regularity holds on the reachable compact sets considered below.

### 3 Representation and reachable compact sets

Let  $\Psi$  be a finite primitive library consisting of affine maps, convolutional linear maps, scaled dot-product compare, row-softmax, weighted sums, residual addition, binary concatenation, normalization, and a finite set of pointwise nonlinearities. Let  $\mathcal{A}_{L,P}(\Psi)$  denote functions implemented by DAGs over  $\Psi$  with at most  $L$  primitive nodes and at most  $P$  trainable parameters.

**Lemma 3.1** (DAG lifting). *Every finite DAG computation can be represented as a sequential composition of state-update maps.*

*Proof.* Take a topological order  $v_1, \dots, v_s$ . The state at step  $t$  stores the input and all outputs of nodes  $v_1, \dots, v_t$ . The update map at step  $t+1$  reads the parent coordinates required by  $v_{t+1}$ , applies  $h_{v_{t+1}}$ , and appends the result. A final projection extracts the sink coordinates.  $\square$

**Lemma 3.2** (Error propagation). *Let  $U_t, \hat{U}_t$  be maps on a compact state set with*

$$\|U_t - \hat{U}_t\|_\infty \leq \delta_t, \quad \text{Lip}(U_t) \leq L_t.$$

*Then*

$$\|U_s \circ \dots \circ U_1 - \hat{U}_s \circ \dots \circ \hat{U}_1\|_\infty \leq \sum_{t=1}^s \left( \prod_{\ell=t+1}^s L_\ell \right) \delta_t.$$

*Proof.* Let  $F_t = U_t \circ \dots \circ U_1$  and  $\hat{F}_t = \hat{U}_t \circ \dots \circ \hat{U}_1$ . Then

$$\|F_t - \hat{F}_t\|_\infty \leq L_t \|F_{t-1} - \hat{F}_{t-1}\|_\infty + \delta_t.$$

Iterating the recurrence proves the bound.  $\square$

**Theorem 3.3** (Exact finite grammar representation). *For every  $f \in \mathcal{A}_{L,P}(\Psi)$ ,*

$$f \in F_{2,L',P,\Phi^{\text{ex}}}$$

*for some  $L' \leq cL$ , where  $c$  accounts only for binary expansion of multi-input concatenations or aggregations. Consequently  $f \in P_\epsilon(\mu; 2, L', P, \Phi^{\text{ex}})$  for every  $\epsilon > 0$  and probability measure  $\mu$ .*

*Proof.* Replace every primitive node by the corresponding exact motif node: affine maps by projection, scaled dot-products by compare, row-softmax and pointwise nonlinearities by exact select, weighted sums by aggregate, residual addition by iterate, concatenation by compose, and normalization by norm. Convolutions on finite grids are linear maps with structured sparsity, hence projection motifs. Parameter counts are preserved, and parameter-free nodes add no trainable parameters. The resulting motif graph computes the same function exactly.  $\square$

**Theorem 3.4** (Approximate finite grammar representation). *Let  $K \subset \mathbb{R}^d$  be compact. For every  $f \in \mathcal{A}_{L,P}(\Psi)$  and every  $\epsilon > 0$ , there exists  $\hat{f} \in F_{2,L',B_\epsilon,\Phi^{\text{abs}}}$  such that*

$$\|f - \hat{f}\|_{\infty,K} \leq \epsilon.$$

*If  $\text{supp } \mu \subset K$ , then  $\|f - \hat{f}\|_{L^2(\mu)} \leq \epsilon$ .*

*Proof.* Lift the primitive DAG to a state-update chain. Since each exact primitive is continuous and the input set is compact, each reachable state set  $K_t$  is compact. Exact affine, compare, aggregate, residual, compose, and norm updates are retained. For each nonlinear select or softmax update, use the universal select family to approximate it uniformly on its reachable compact set  $K_t$  to tolerance  $\delta_t$ . By the error propagation lemma, choosing

$$\delta_t \leq \frac{\epsilon}{L \max_t \prod_{\ell > t} L_\ell}$$

ensures final error at most  $\epsilon$ . The required parameter budget  $B_\epsilon$  is finite by universal approximation.  $\square$

The key point is not merely universality. The approximation occurs on a sequence of reachable compact sets  $K_t$ , not on the entire ambient space. Each motif transforms the data geometry so that the next local operation is simpler on the states that actually occur.

## 4 Composition algebra, identification, and morphology

**Proposition 4.1** (Serial composition). *If  $f = g \circ h$ ,  $g$  is Lipschitz on  $h(K)$  with constant  $L_g$ , and  $\hat{h}, \hat{g}$  approximate  $h, g$  with errors  $\epsilon/(2L_g)$  and  $\epsilon/2$ , then  $\hat{g} \circ \hat{h}$  approximates  $f$  with error at most  $\epsilon$ . Depth and parameter budgets add; width is bounded by the maximum required component width under sequential execution.*

**Proposition 4.2** (Parallel composition). *If  $f = (f_1, \dots, f_m)$ , then independent realization of the components gives*

$$W_f \leq \sum_j W_{f_j}, \quad D_f \leq \max_j D_{f_j}, \quad P_f \leq \sum_j P_{f_j}.$$

**Proposition 4.3** (Shared subroutines). *If several branches reuse a common subgraph  $s$ , then the parameter cost of  $s$  is counted once rather than once per use. Thus motif graph complexity is subadditive under parameter sharing.*

**Theorem 4.4** (Stable motif identification by margin). *Let  $M_i \subset C(K, \mathbb{R}^r)$  be closed motif families. For a target local operator  $f$ , define*

$$d_i(f) = \inf_{g \in M_i} \|f - g\|_{\infty, K}.$$

*Suppose there is a unique best family  $i^*$  with margin  $2\Delta$ :*

$$d_{i^*}(f) + 2\Delta \leq d_j(f), \quad j \neq i^*.$$

*If empirical estimates  $\hat{d}_i$  satisfy  $|\hat{d}_i - d_i(f)| \leq \Delta$  for all  $i$ , then  $\arg \min_i \hat{d}_i = i^*$ .*

*Proof.* For all  $j \neq i^*$ ,

$$\hat{d}_{i^*} \leq d_{i^*} + \Delta < d_j - \Delta \leq \hat{d}_j.$$

$\square$

**Theorem 4.5** (Convex budget allocation). *Consider a motif graph with node budgets  $B_v$  and separable error proxy*

$$E(B) = \sum_v \alpha_v \epsilon_{i_v}(B_v), \quad \sum_v B_v = B_{\text{total}},$$

*where  $\alpha_v > 0$ , and each  $\epsilon_i$  is continuous, strictly decreasing, differentiable, and strictly convex on  $(0, \infty)$ . If the optimum is interior, then it is unique and satisfies*

$$\alpha_v \epsilon'_{i_v}(B_v^*) = \lambda$$

*for all active nodes.*

*Proof.* Strict convexity gives uniqueness. The Lagrangian

$$\mathcal{L}(B, \lambda) = \sum_v \alpha_v \epsilon_{i_v}(B_v) + \lambda \left( \sum_v B_v - B_{\text{total}} \right)$$

yields the stationarity condition  $\alpha_v \epsilon'_{i_v}(B_v^*) + \lambda = 0$ .  $\square$

**Corollary 4.6** (Uniform-interface suboptimality). *If a uniform architecture constrains  $B_u = B_v$  for a set of motifs whose weighted marginal utilities  $\alpha_v \epsilon'_{i_v}(B)$  differ at that common budget, then the uniform allocation is not the unconstrained optimum. If the objective is strongly convex, the constrained allocation has a positive error gap proportional to its squared distance from the optimum.*

This theorem formalizes a function-morphology correspondence: efficient architecture is determined not only by total parameter count but by how motif-specific error curves and task weights distribute marginal utility.

## 5 Emergence as coupled budget thresholds

Emergent abilities in language models are often described as capabilities absent in smaller models but present in larger ones [11]. Some apparent discontinuities can arise from metric choices or thresholding [12]. We add a structural explanation: apparent emergence can arise when a capability depends on several motifs that must simultaneously reach sufficient effectiveness under a constrained budget allocation.

### 5.1 Effectiveness curves and coupled capabilities

For motif  $i$ , let  $\epsilon_i(B_i)$  be its error-vs-budget curve and  $\epsilon_i^* = \lim_{B \rightarrow \infty} \epsilon_i(B)$ . Define normalized effectiveness

$$q_i(B_i) = \frac{\epsilon_i(0) - \epsilon_i(B_i)}{\epsilon_i(0) - \epsilon_i^*} \in [0, 1].$$

A convenient soft-threshold model is

$$q_i(B_i) = \frac{B_i^{\gamma_i}}{B_i^{\gamma_i} + (B_{\min}^{(i)})^{\gamma_i}},$$

where  $B_{\min}^{(i)}$  is a characteristic threshold and  $\gamma_i$  controls sharpness.

Let  $\mathcal{C}_R$  be the critical motifs for a capability  $R$ . Define latent capability strength

$$R_{\text{lat}}(B) = A_R \prod_{i \in \mathcal{C}_R} q_i(B_i)^{\rho_i}, \quad A_R > 0, \quad \rho_i > 0.$$

An observed benchmark score may be thresholded:

$$R_{\text{obs}}(B) = \mathbf{1}\{R_{\text{lat}}(B) \geq \theta_R\}.$$

**Theorem 5.1** (Coupled budget threshold). *Let  $R_{\text{lat}}$  and  $R_{\text{obs}}$  be defined as above. Then observed capability appears exactly when*

$$\prod_{i \in \mathcal{C}_R} q_i(B_i)^{\rho_i} \geq \frac{\theta_R}{A_R}.$$

*If any critical motif has sufficiently small  $q_i(B_i)$ , the observed capability is absent even when all other motifs are strong.*

*Proof.* This follows immediately by substituting the product expression for  $R_{\text{lat}}$  into the threshold condition. Since every  $q_i \leq 1$ , a single factor below  $\theta_R/A_R$  after exponentiation upper-bounds the entire product below threshold.  $\square$

## 5.2 Uniform cliffs and heterogeneous bypass

Suppose uniform geometry fixes budget shares

$$B_i^{\text{uni}} = s_i B_{\text{total}}, \quad s_i > 0.$$

In a layer-wise model,  $s_i = \pi_i/L$ , where  $\pi_i$  is the fraction of layer budget devoted to motif  $i$ .

**Theorem 5.2** (Uniform emergence cliff in the hard-threshold limit). *If*

$$q_i(B_i) = \begin{cases} 0, & B_i < B_{\min}^{(i)}, \\ 1, & B_i \geq B_{\min}^{(i)}, \end{cases}$$

*then the uniform architecture observes the capability at*

$$B_{\text{cliff}}^{\text{uni}} = \max_{i \in \mathcal{C}_R} \frac{B_{\min}^{(i)}}{s_i}.$$

*If  $s_i = \pi_i/L$ , then*

$$B_{\text{cliff}}^{\text{uni}} = L \max_{i \in \mathcal{C}_R} \frac{B_{\min}^{(i)}}{\pi_i}.$$

*Proof.* The capability is observed if and only if all critical motifs cross threshold:  $s_i B_{\text{total}} \geq B_{\min}^{(i)}$  for all  $i$ . Solving the inequalities gives the maximum.  $\square$

**Theorem 5.3** (Heterogeneous threshold advantage). *In the hard-threshold model, the minimal heterogeneous budget for the critical motifs is*

$$B_{\text{cliff}}^{\text{het}} = \sum_{i \in \mathcal{C}_R} B_{\min}^{(i)}$$

plus any background budget for noncritical computation. Moreover,

$$B_{\text{cliff}}^{\text{het}} \leq B_{\text{cliff}}^{\text{uni}},$$

with strict inequality unless the uniform shares match the threshold profile.

*Proof.* The heterogeneous allocation assigns exactly  $B_{\min}^{(i)}$  to each critical motif. For uniform allocation,  $B_{\text{total}} \geq \max_i B_{\min}^{(i)}/s_i$ . Since  $\sum_i s_i \leq 1$ , this maximum is at least  $\sum_i B_{\min}^{(i)}$ , with equality only when  $s_i = B_{\min}^{(i)}/\sum_j B_{\min}^{(j)}$  and no critical budget is wasted elsewhere.  $\square$

**Corollary 5.4** (Domain-specific thresholds). *If domain  $X$  rescales motif effectiveness by  $q_i^{(X)}(B_i) = q_i(\alpha_i^{(X)} B_i)$ , then*

$$B_{\text{cliff}}^{\text{uni},X} = \max_i \frac{B_{\min}^{(i)}}{\alpha_i^{(X)} s_i}.$$

Thus math, code, retrieval, and planning can have different apparent thresholds because their critical motif weights differ.

ECBT is conditional. It does not prove that every observed emergence phenomenon in real LLMs has product-form motif structure. It proves that if a capability is coupled across motifs, then apparent cliffs, domain-specific thresholds, and heterogeneous advantage follow mathematically.

## 6 Transformer decomposition and motif-upcycling

### 6.1 Transformer motif decomposition

A standard attention head computes

$$\begin{aligned} Q &= XW_Q, \quad K = XW_K, \quad V = XW_V, \\ S &= QK^\top / \sqrt{d_k}, \quad A = \text{softmax}(S + M), \quad Y = AV. \end{aligned}$$

This is a direct motif chain:

$$\text{proj} \rightarrow \text{compare} \rightarrow \text{select} \rightarrow \text{aggregate}.$$

The multi-head output adds composition and projection. The feed-forward network is an expand/select/compress motif; residual updates are iterate motifs; RMSNorm/LayerNorm acts as normalization/control.

Modern Qwen2.5 and Qwen3 models are causal Transformers using GQA, RoPE, RMSNorm, and SwiGLU-style MLPs [17, 18, 19]. Qwen2.5-1.5B has 1.54B parameters, 28 layers, GQA with 12 query heads and 2 KV heads, and 32,768 token context [17]. Qwen3-4B has 4.0B parameters, 36 layers, GQA with 32 query heads and 8 KV heads, and native 32,768 token context with YaRN extension to 131,072 tokens [18].



## 6.2 Exact-sum factorization of SwiGLU FFNs

Qwen-style MLPs have the form

$$F(x) = W_{\text{down}} (\phi(W_{\text{gate}}x) \odot W_{\text{up}}x).$$

Let the intermediate channels be partitioned into disjoint sets  $S_1, \dots, S_M$ . Define expert slice

$$E_m(x) = W_{\text{down}}[:, S_m] (\phi(W_{\text{gate}}[S_m, :]x) \odot W_{\text{up}}[S_m, :]x).$$

Then

$$F(x) = \sum_{m=1}^M E_m(x).$$

This is exact because the nonlinearity and Hadamard product are channelwise before the final linear down-projection.

A routed motif FFN is

$$\hat{F}(x) = \sum_{m=1}^M \alpha_m(x) E_m(x), \quad \alpha(x) = M \text{ softmax}(r(x)).$$

If  $r(x) = 0$ , then  $\alpha_m(x) = 1$  for all  $m$ , so  $\hat{F}(x) = F(x)$ . This gives a function-preserving initialization.

## 6.3 Motif-specific adapters

After exact splitting, motif-local adaptation is introduced by low-rank deltas. Low-rank adaptation freezes pretrained weights and adds trainable low-rank matrices, greatly reducing trainable parameter count relative to full fine-tuning [16]. We use motif-specific placements:

- expand: adapters on up/down paths;
- select: adapter on gate path;
- memory: adapters on gate, up, and down paths.

Attention adapters can be added separately as compare adapters on  $Q/K$  projections and memory/aggregate adapters on  $V/O$ , but the strongest preliminary Qwen results reported here come from FFN-only motif LoRA.

This procedure is related to MoE and dense-to-MoE upcycling, but it differs in one important respect: the FFN split is exactly function-preserving before routing or adapter training. Sparse MoE systems scale capacity by routing tokens to experts [13], Soft MoE explores differentiable expert mixtures [14], and recent upcycling work converts dense checkpoints into MoE architectures [15]. Motif-upcycling instead imposes a semantic partition aligned with operator roles and initializes as the exact dense function.

## 7 Scale-Aware Residual Control

Motif-upcycling separates pretrained computation into routed motif slices and trainable local interventions. A remaining question is how strongly a new intervention should be allowed to change the residual stream. Standard pre-normalized Transformer blocks compute

$$x^+ = x + u, \quad u = F(\text{Norm}(x)),$$

but the residual addition does not explicitly compare the update magnitude to the current state magnitude. We introduce *Scale-Aware Residual Control* (SARC) as a lightweight control motif for this role. It is related to residual gating and residual scaling ideas such as ReZero and DeepNorm, but differs by using the relative RMS scale of a trainable intervention and by preserving the donor residual path in the adapter-only form [20, 21].

For  $z \in \mathbb{R}^d$ , define

$$\text{RMS}(z) = \left( \frac{1}{d} \sum_{j=1}^d z_j^2 \right)^{1/2}, \quad R_\varepsilon(z) = \text{RMS}(z) + \varepsilon,$$

with  $\varepsilon > 0$ . For sequence tensors, RMS is computed token-wise over the hidden dimension. Given residual state  $x$  and update  $u$ , define the log relative scale

$$r(x, u) = \log \frac{R_\varepsilon(u)}{R_\varepsilon(x)}.$$

The identity-preserving SARC correction is

$$\text{SARC}_\theta(x, u) = x + (1 + \delta_\theta(r(x, u))) u, \quad \delta_\theta(r) = \lambda \tanh h_\theta(r),$$

where  $0 < \lambda < 1$  and  $h_\theta : \mathbb{R} \rightarrow \mathbb{R}$  is a small token-wise network. In implementation,  $h_\theta$  can be a two-layer MLP  $1 \rightarrow H \rightarrow 1$  whose final affine layer is initialized to zero.

**Proposition 7.1** (Identity preservation). *Let  $B(x) = x + u(x)$  be a residual block and*

$$\widehat{B}_\theta(x) = x + (1 + \lambda \tanh h_\theta(r(x, u(x)))) u(x).$$

*If  $h_\theta(r) = 0$  for all  $r$  at initialization, then  $\widehat{B}_\theta(x) = B(x)$  for all  $x$ .*

*Proof.* If  $h_\theta(r) = 0$ , then  $\tanh h_\theta(r) = 0$ , so the multiplicative scale is one and

$$\widehat{B}_\theta(x) = x + u(x) = B(x).$$

□

**Proposition 7.2** (Bounded residual perturbation). *For the identity-preserving SARC correction with  $|\delta_\theta(r)| \leq \lambda$ ,*

$$\|\text{SARC}_\theta(x, u) - (x + u)\| \leq \lambda \|u\|.$$

*Proof.* The difference is

$$\text{SARC}_\theta(x, u) - (x + u) = \delta_\theta(r)u.$$

Taking norms and using  $|\delta_\theta(r)| \leq \lambda$  gives the result. □

A more restrictive stabilizing variant is

$$g_\tau(r) = \sigma(\log \tau - r) = \frac{\tau}{\tau + e^r}, \quad x^+ = x + g_\tau(r)u.$$

It is not the safest default for pretrained residual branches because it does not exactly preserve  $x + u$  unless initialized to pass through the relevant range, but it gives an explicit relative-update bound:

$$\text{RMS}(g_\tau(r)u) \leq \tau R_\varepsilon(x).$$

Indeed  $e^r = R_\varepsilon(u)/R_\varepsilon(x)$ , so

$$g_\tau(r)e^r = \frac{\tau e^r}{\tau + e^r} \leq \tau.$$

## 7.1 Adapter-only and motif-wise SARC

For pretrained models, the safest variant is to leave the frozen donor update unchanged and apply SARC only to the new trainable intervention. Write

$$u(x) = u_0(x) + \Delta u_\theta(x),$$

where  $u_0$  is the frozen donor update and  $\Delta u_\theta$  is a trainable adapter or motif-local correction. Adapter-only SARC is

$$x^+ = x + u_0(x) + \left(1 + \lambda \tanh h_\phi \left(\log \frac{R_\varepsilon(\Delta u_\theta(x))}{R_\varepsilon(x)}\right)\right) \Delta u_\theta(x).$$

Thus the pretrained residual geometry is preserved at initialization while the new intervention receives a bounded, learnable scale controller.

In motif-upcycled modules,

$$F(x) = \sum_{m=1}^M \alpha_m(x) E_m(x).$$

A motif-wise SARC version is

$$F_{\text{SARC}}(x) = \sum_{m=1}^M \alpha_m(x) (1 + \delta_m(r_m)) E_m(x), \quad r_m = \log \frac{R_\varepsilon(E_m(x))}{R_\varepsilon(x)}.$$

If the motif split is exact, the router is neutral  $\alpha_m(x) = 1$ , and each  $\delta_m = 0$  at initialization, then  $F_{\text{SARC}}(x) = F(x)$ . SARC therefore preserves the P0 safety property while separating two roles: the router selects which motif contributes, and SARC controls how strongly that contribution modifies the residual stream.

For a token-wise scalar SARC controller implemented as  $1 \rightarrow H \rightarrow 1$ , the parameter count per gate is

$$P_{\text{SARC}} = (H + H) + (H + 1) = 3H + 1.$$

With  $H = 16$ , this is only 49 parameters per gate. For  $L_p$  patched layers and  $B$  controlled branches per layer,

$$P_{\text{SARC},\text{total}} = L_p B (3H + 1),$$

and for motif-wise SARC with  $M$  experts per patched layer,

$$P_{\text{SARC},\text{total}} = L_p M (3H + 1).$$

$$x^+ = x + u_0(x) + \left(1 + \lambda \tanh h_\phi \left( \log \frac{R_\epsilon(\Delta u_\theta)}{R_\epsilon(x)} \right)\right) \Delta u_\theta(x)$$

Adapter-only SARC controls the trainable intervention while leaving the frozen donor update unchanged.

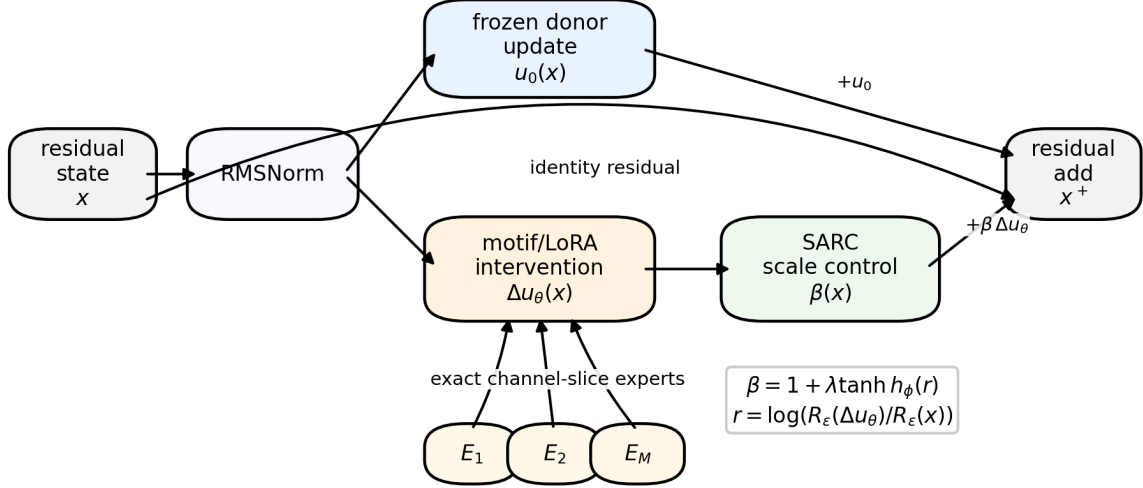


Figure 1: Adapter-only SARC inside a motif-upcycled residual block. The frozen donor update  $u_0(x)$  is left unchanged. SARC controls only the trainable motif/LoRA intervention  $\Delta u_\theta(x)$  through a bounded scale  $\beta(x) = 1 + \lambda \tanh h_\phi(r)$ .

## 8 Experiments

The experiments are preliminary and should be read as feasibility tests rather than full benchmark claims. They ask three practical questions: whether pretrained Transformers can be factorized without changing their function at initialization, whether routing and motif-local LoRA can be trained with a frozen backbone, and whether small motif-local budgets can measurably change validation loss.

### 8.1 Protocol

The staged protocol is:

- P0** Exact-sum FFN factorization with neutral routing; test base-vs-patched logits and hidden states.
- P1** Static router over frozen motif slices.
- P2** Contextual router over frozen motif slices.
- P3** Contextual router over a deeper patched layer set.
- P4** Contextual router plus motif-specific FFN LoRA.
- P5** FFN motif LoRA plus attention compare/memory adapters.

## 8.2 Qwen2.5-1.5B results

On Qwen2.5-1.5B, layers 11–14 were patched in P0, P1, P2, P4, and P5; P3 used layers 10–17. Sequence length was 512, batch size 1, and the validation slice contained 8192 evaluation tokens in the smoke runs. Table 1 summarizes the relevant observations.

Stage	Layers	Trainable params	Best val loss	PPL	Interpretation
Dense baseline	–	0	2.25869	9.57055	frozen donor
P0 exact split	11–14	0	–	–	zero logit/hidden diff
P1 static motif	11–14	12	2.25647	9.54930	safe, tiny effect
P2 contextual motif	11–14	788,492	2.20331	9.05495	routing helps
P3 contextual motif	10–17	1,576,984	2.18388	–	deeper routing helps
P4 motif LoRA	11–14	2,803,724	2.10537	–	strongest FFN result
P4 flat LoRA	11–14	2,803,724	2.11329	–	matched flat control
P5 attention soft	11–14	3,902,996	2.11068	–	improves dense, below P4
P5 attention sparse	11–14	3,902,996	2.10773	–	improves dense, below P4

Table 1: Qwen2.5-1.5B staged motif-upcycling smoke results. P0 exact factorization produced zero measured difference between base and patched outputs in tested logits and hidden states. P4 motif LoRA outperformed a matched flat split in this run. Dashes indicate values not central to the stage summary or not applicable.

The P0 result is the most important safety result: exact-sum factorization with neutral routing is numerically function-preserving in the tested setting. P2 shows that contextual routing can improve validation loss under a frozen backbone. P4 shows that motif-specific LoRA provides a stronger improvement and beats a flat matched split in the smoke run. P5 attention adapters improve the dense baseline but did not exceed the best FFN-only motif-LoRA result.

## 8.3 Qwen3-4B results

For Qwen3-4B, the reported run used layers 15–18, sequence length 1024, batch size 1, three seeds, and a P4-lite configuration: FFN-only exact split, contextual router, motif-specific LoRA, no attention adapters, and no sparse top-k routing. The dataset contained 2,519,040 training tokens and 262,144 validation tokens. The run trained 3,672,076 parameters per seed, less than 0.1% of a 4B-parameter model.

Seed	Trainable params	Best step	Best val loss	PPL
0	3,672,076	799	2.18069	8.85240
1	3,672,076	1499	2.17495	8.80175
2	3,672,076	1299	2.18009	8.84712
Mean	3,672,076	–	2.17858	8.83376
Std.	–	–	0.00258	–

Table 2: Qwen3-4B P4-lite motif-LoRA adaptation on layers 15–18. The average validation loss across seeds was 2.17858 with standard deviation 0.00258.

The Qwen3-4B run should not be interpreted as a comparison against the dense donor, because the dense baseline was intentionally skipped in that run. It does show that the motif-upcycling procedure

scales to a 4B dense donor, trains stably across seeds, and produces a consistent validation-loss trajectory under a small trainable budget.

Calibration summaries in both Qwen2.5 and Qwen3 runs show large differences among intermediate-channel scores. In Qwen3-4B layers 15, 17, and 18, the first score component was around 0.44–0.45 while the second and third were roughly 0.03–0.04 and 0.017–0.021; layer 16 exhibited a large outlier in the third component. These differences are consistent with the core theory: local motif difficulty is not uniform across layers or operator roles.

#### 8.4 Qwen3.5-0.8B hybrid motif/LoRA run

We additionally analyzed a more recent hybrid run on `Qwen/Qwen3.5-0.8B-Base`. This run differs from the FFN-only P4-lite setting: the patched layers were 10 and 11, the motif set contained six labels

{expand, select, memory, compose, code\_\_syntax, math\_\_reason},

and the resolved attention targets included both state/attention projections and standard attention projections. We therefore treat the run as a hybrid P4/P5-like motif/LoRA experiment rather than a pure FFN-only P4 experiment.

The configuration used sequence length 128, learning rate  $10^{-4}$ , LoRA rank 8, evaluation every 500 steps, six evaluation domains, and 10,000 training steps. The model contained 753,085,516 total parameters, of which 692,492 were trainable, or 0.092%. At step 0, the mean validation-loss delta against the dense baseline was  $6.83 \times 10^{-4}$ , indicating an approximately neutral initialization. The best mean validation-loss improvement over the dense baseline was 0.4578 at step 4000. The final step retained a positive mean improvement of 0.3874, with a late degradation of 0.0704 relative to the best checkpoint.

Domain	Baseline loss	Final patched loss	Final $\Delta$	Final PPL reduction
Wiki	2.8519	2.4224	0.4295	34.9%
News	4.0368	3.8124	0.2244	20.1%
Reviews	3.9532	3.8898	0.0634	6.1%
Paper	3.2441	2.5875	0.6566	48.1%
Code	1.5638	1.0633	0.5005	39.4%
Math	1.7217	1.2717	0.4501	36.2%
Mean	2.8953	2.5078	0.3874	–

Table 3: Final-domain results for the Qwen3.5-0.8B hybrid motif/LoRA run.  $\Delta$  denotes baseline loss minus patched loss, so positive values indicate improvement.

This run motivates SARC but does not evaluate it: the reported numbers are for the current hybrid motif/LoRA baseline. The late-checkpoint degradation and domain-dependent router weights indicate why a residual scale-control motif may be useful as the next ablation. In particular, adapter-only SARC can test whether residual intervention control improves final stability without changing the frozen donor path.

### Qwen3.5-0.8B hybrid motif/LoRA run: evaluation summary

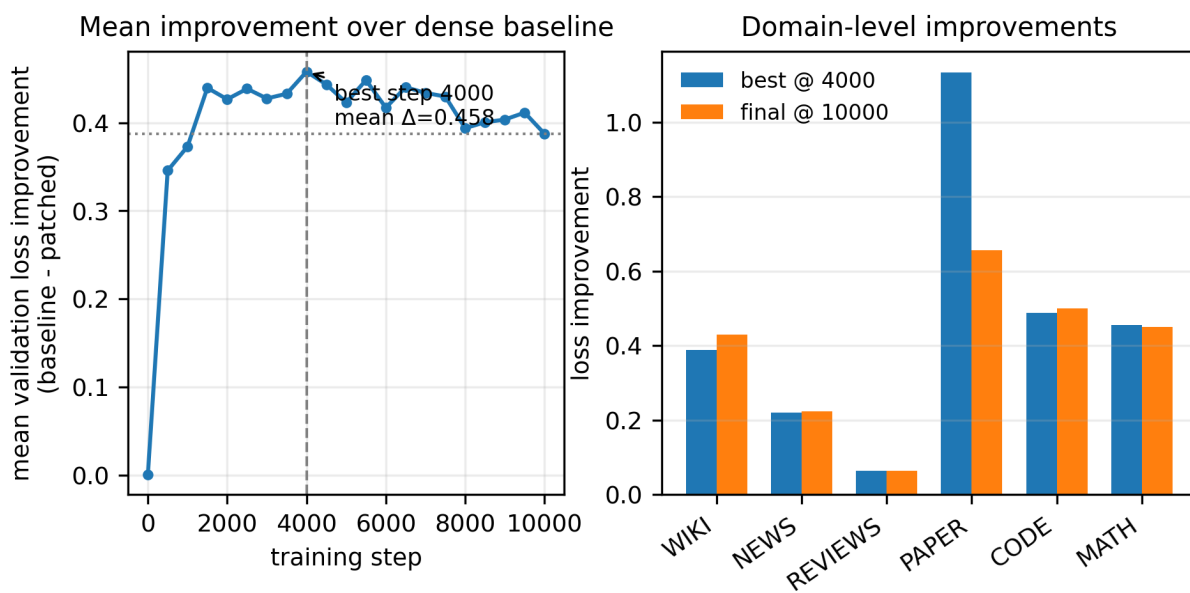


Figure 2: Qwen3.5-0.8B evaluation summary. Left: mean validation-loss improvement over the frozen dense baseline. Right: domain-level improvements at the best mean checkpoint and at the final checkpoint. The largest late degradation occurs in the paper domain, suggesting a useful early-stopping point near step 4000.

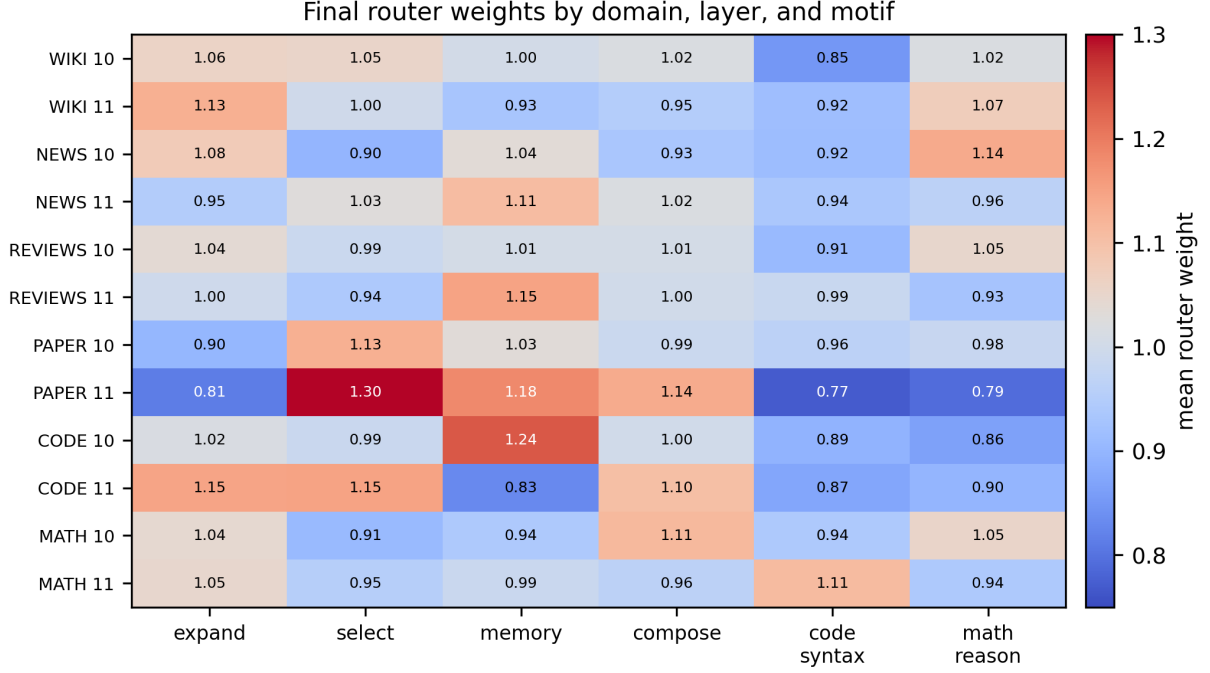


Figure 3: Final router weights by domain, layer, and motif for the Qwen3.5-0.8B hybrid run. The router remains soft rather than collapsing to a single motif, but the learned weights are domain-dependent. The names of motifs should be interpreted as slot labels rather than proven semantic categories.

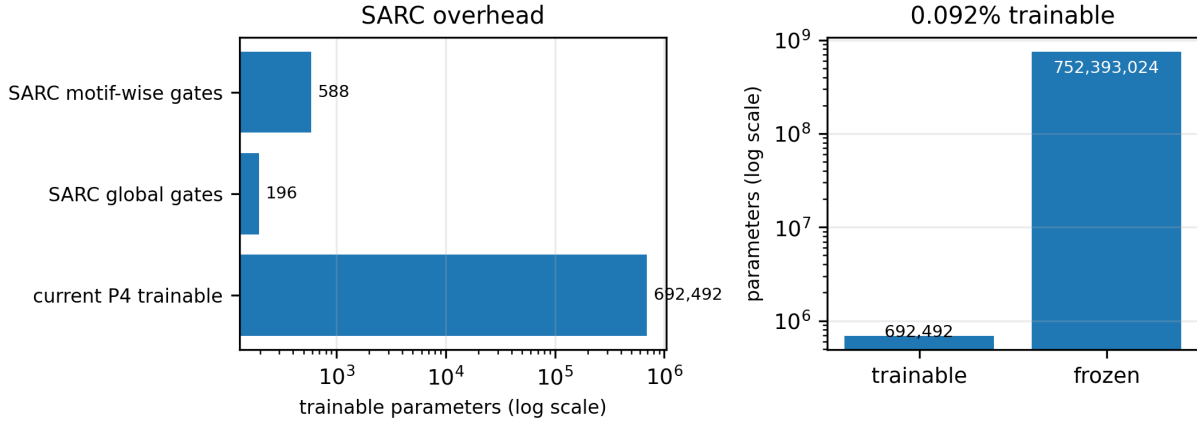


Figure 4: Trainable budget and SARC overhead for the Qwen3.5-0.8B setting. A  $1 \rightarrow 16 \rightarrow 1$  token-wise scalar SARC gate costs 49 parameters. Global attention/FFN SARC over two layers costs 196 parameters; motif-wise SARC over two layers and six motifs costs 588 parameters, which is negligible relative to the 692,492 trainable parameters of the current run.



## 9 Interpretation

The theoretical and empirical pieces support a single conservative interpretation: motif-upcycling is a safe way to expose and adapt structure that is already present inside selected Transformer modules. The exact factorization result is the strongest claim. It shows that a dense SwiGLU FFN can be rewritten as a sum of channel-slice motif experts without changing the function, and that neutral routing preserves the donor at initialization.

The finite-grammar results provide a formal language for this operation. They do not claim that every useful function is best described by a small motif graph. Instead, they show that architecture-generated computations can be represented by bounded motif compositions, that approximation error can be controlled on reachable compact sets, and that motif-specific budget curves naturally lead to non-uniform allocation rules. ECBT should likewise be interpreted conditionally: if a capability requires several coupled motif strengths, then threshold-like behavior and domain-specific cliffs can arise from the product structure.

SARC adds a complementary control layer. Motif routers answer which structured branch should contribute; SARC answers how strongly a new trainable intervention should modify the residual stream. Its identity-preserving initialization makes it compatible with exact-sum upcycling, while its bounded correction gives a local perturbation guarantee. In this draft, SARC is a formal and architectural extension; SARC-specific training runs remain future work.

The experiments support a limited empirical claim. Motif-structured factorization is implementable, can be initialized safely, and can be trained with small parameter budgets to alter validation loss in selected Qwen-family settings. They do not establish frontier-level quality, universal superiority over flat LoRA, or long-context robustness. The matched Qwen2.5 P4 comparison and the Qwen3.5 hybrid run are encouraging, but stronger claims require more seeds, more domains, long-context evaluation, and direct SARC ablations.

## 10 Limitations

This work is preliminary. We explicitly do not claim that motif-upcycling is optimal, universally applicable, or sufficient for frontier-level model quality. The main limitations are:

- The finite-grammar theorems cover architecture-generated classes and reachable compact sets, not all practically relevant functions.
- The approximate theorem depends on local compactness and local Lipschitz constants; it is not a global approximation guarantee over the full ambient space.
- ECBT is a conditional product-coupling model. It is a mechanism by which threshold-like behavior can arise, not a proof that all real LLM emergence follows this mechanism.
- The Qwen experiments are preliminary. They use limited datasets, limited seeds, and do not include a comprehensive benchmark suite.
- SARC is introduced and analyzed formally, but SARC-specific training runs are not included in the current empirical section. The Qwen3.5 results are hybrid motif/LoRA baselines that motivate SARC rather than validate it.
- Qwen3-4B P4-lite results currently lack a dense baseline in the same run configuration.

- Attention motif adapters showed signs of branch collapse in some sparse settings and need stronger regularization before being treated as a stable method.
- Long-context behavior is not established by the short-context experiments reported here and must be evaluated separately.

These limitations are intentional scope boundaries for the present version. The contribution is a structure-preserving adaptation framework with preliminary evidence, not a final recipe for optimal model training.

## 11 Conclusion

We presented motif-upcycling, a preliminary framework for structure-preserving adaptation of pretrained Transformer models. The core idea is to expose motif-aligned components inside selected modules while preserving the donor function at initialization. For SwiGLU FFNs, this is achieved by exact channel-wise factorization: the dense module is rewritten as a sum of motif slices, and neutral routing recovers the original function.

On top of this exact decomposition, motif-upcycling adds contextual routing and motif-local trainable interventions. We also introduced SARC, a lightweight identity-preserving residual controller that regulates the magnitude of new trainable interventions relative to the residual stream. In this view, motif routers choose which branch contributes, while SARC controls the strength of that contribution.

Preliminary Qwen-family experiments suggest that this structure can improve validation loss in selected settings with less than 0.1% trainable parameters. These results do not establish optimality, universal validity, or long-context robustness. They do show that pretrained Transformer modules can be safely factorized, instrumented, and adapted with small structured budgets. We view this as a step toward architecture calculation: not only deciding how many parameters to use, but deciding which computational motifs should receive them.

## A Full proof of soft-threshold ECBT

Let

$$q_i(B_i) = \frac{B_i^{\gamma_i}}{B_i^{\gamma_i} + (B_{\min}^{(i)})^{\gamma_i}}.$$

Its inverse is

$$q_i^{-1}(\eta) = B_{\min}^{(i)} \left( \frac{\eta}{1 - \eta} \right)^{1/\gamma_i}, \quad 0 < \eta < 1.$$

Let  $\Theta = \theta_R/A_R$ . The observed capability condition is

$$\prod_i q_i(B_i)^{\rho_i} \geq \Theta.$$

A necessary condition is

$$q_i(B_i) \geq \Theta^{1/\rho_i} \quad \text{for all } i,$$

because all other factors are at most one. Thus in a uniform architecture  $B_i = s_i B_{\text{total}}$ ,

$$B_{\text{total}} \geq \max_i \frac{q_i^{-1}(\Theta^{1/\rho_i})}{s_i}.$$

Let  $\rho_\Sigma = \sum_i \rho_i$ . A sufficient condition is

$$q_i(B_i) \geq \Theta^{1/\rho_\Sigma} \quad \text{for all } i,$$

since then

$$\prod_i q_i(B_i)^{\rho_i} \geq \prod_i \left(\Theta^{1/\rho_\Sigma}\right)^{\rho_i} = \Theta.$$

Therefore

$$\max_i \frac{q_i^{-1}(\Theta^{1/\rho_i})}{s_i} \leq B_{\text{cliff}}^{\text{uni}} \leq \max_i \frac{q_i^{-1}(\Theta^{1/\rho_\Sigma})}{s_i}.$$

As  $\gamma_i \rightarrow \infty$ ,  $q_i^{-1}(\eta) \rightarrow B_{\min}^{(i)}$  for fixed  $\eta \in (0, 1)$ , recovering the hard-threshold formula.

## B Implementation sketch: Qwen motif-upcycling

For a patched layer  $\ell$ , replace the original MLP by a wrapper containing frozen channel-slice experts  $E_{\ell,m}$ , a router  $r_\ell$ , and optional motif-local low-rank adapters. The neutral forward pass is

$$\hat{F}_\ell(x) = \sum_m M \text{softmax}(r_\ell(x))_m E_{\ell,m}(x),$$

with  $r_\ell(x) = 0$  at initialization. For  $M$  experts, this gives coefficient 1 for every expert. LoRA modules are initialized at zero so that the patched model equals the dense model at step zero.

## C Implementation sketch: SARC

A minimal identity-preserving SARC scaler computes

$$r = \log \frac{\text{RMS}(u) + \varepsilon}{\text{RMS}(x) + \varepsilon}, \quad \beta = 1 + \lambda \tanh h_\theta(r),$$

where  $h_\theta$  is zero-initialized at the final affine layer. In pseudocode:

$$\begin{aligned} R_x &\leftarrow \sqrt{\text{mean}(x^2) + \varepsilon}, \\ R_u &\leftarrow \sqrt{\text{mean}(u^2) + \varepsilon}, \\ r &\leftarrow \log((R_u + \varepsilon)/(R_x + \varepsilon)), \\ \beta &\leftarrow 1 + \lambda \tanh h_\theta(r), \\ &\text{return } x + \beta u. \end{aligned}$$

For adapter-only SARC,  $u$  is replaced by the trainable delta  $\Delta u_\theta$ , and the final residual update is  $x + u_0 + \beta \Delta u_\theta$ . The recommended initial ablation is  $H = 16$ ,  $\lambda = 0.1$ , and adapter-only application to FFN motif deltas.

## D Reproducibility details

Qwen2.5-1.5B smoke experiments used sequence length 512, batch size 1, validation batches 16, learning rate  $2 \times 10^{-4}$ , and patched layers 11–14 except P3, which used layers 10–17. Qwen3-4B P4-lite used sequence length 1024, batch size 1, calibration batches 32, training batches 1500, validation batches 32, learning rate  $10^{-4}$ , patched layers 15–18, and three seeds 0, 1, 2. Qwen3.5-0.8B used sequence length 128, evaluation every 500 steps, six domains, learning rate  $10^{-4}$ , 10,000 training steps, seed 0, layers 10–11, six motif slots, LoRA rank 8, and 692,492 trainable parameters.

## References

- [1] A. Vaswani et al. Attention is all you need. *NeurIPS*, 2017. <https://arxiv.org/abs/1706.03762>.
- [2] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 1991. <https://www.sciencedirect.com/science/article/pii/089360809190009T>.
- [3] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 1989.
- [4] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 1993.
- [5] D. Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 2017.
- [6] H. Mhaskar and T. Poggio. Deep vs. shallow networks: An approximation theory perspective. *Analysis and Applications*, 2016/2017.
- [7] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. arXiv:1607.06450, 2016. <https://arxiv.org/abs/1607.06450>.
- [8] M. Geva et al. Transformer feed-forward layers are key-value memories. *EMNLP*, 2021. <https://arxiv.org/abs/2012.14913>.
- [9] J. Kaplan et al. Scaling laws for neural language models. arXiv:2001.08361, 2020. <https://arxiv.org/abs/2001.08361>.
- [10] J. Hoffmann et al. Training compute-optimal large language models. *NeurIPS*, 2022. <https://arxiv.org/abs/2203.15556>.
- [11] J. Wei et al. Emergent abilities of large language models. arXiv:2206.07682, 2022. <https://arxiv.org/abs/2206.07682>.
- [12] R. Schaeffer, B. Miranda, and S. Koyejo. Are emergent abilities of large language models a mirage? *NeurIPS*, 2023. <https://arxiv.org/abs/2304.15004>.
- [13] W. Fedus, B. Zoph, and N. Shazeer. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. arXiv:2101.03961, 2021. <https://arxiv.org/abs/2101.03961>.
- [14] J. Puigcerver et al. From sparse to soft mixtures of experts. arXiv:2308.00951, 2023. <https://arxiv.org/abs/2308.00951>.
- [15] E. He et al. Upcycling large language models into mixture of experts. arXiv:2410.07524, 2024. <https://arxiv.org/abs/2410.07524>.
- [16] E. J. Hu et al. LoRA: Low-rank adaptation of large language models. arXiv:2106.09685, 2021. <https://arxiv.org/abs/2106.09685>.
- [17] Qwen Team. Qwen2.5-1.5B model card. Hugging Face, 2024. <https://huggingface.co/Qwen/Qwen2.5-1.5B>.

- [18] Qwen Team. Qwen3-4B model card. Hugging Face, 2025. <https://huggingface.co/Qwen/Qwen3-4B>.
- [19] Hugging Face Transformers. Qwen2 model documentation. [https://huggingface.co/docs/transformers/model\\_doc/qwen2](https://huggingface.co/docs/transformers/model_doc/qwen2).
- [20] T. Bachlechner et al. ReZero is all you need: Fast convergence at large depth. arXiv:2003.04887, 2020. <https://arxiv.org/abs/2003.04887>.
- [21] H. Wang et al. DeepNet: Scaling Transformers to 1,000 layers. arXiv:2203.00555, 2022. <https://arxiv.org/abs/2203.00555>.
- [22] Qwen Team. Qwen3.5-0.8B-Base model card. Hugging Face, 2026. <https://huggingface.co/Qwen/Qwen3.5-0.8B-Base>.