RESIDUAL-AWARE NEURAL SPECTRAL CAPACITY: ALLOCATING CAPACITY ACROSS THE NETWORK FROM ARCHITECTURE SPECIFICATION ALONE

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting relative trained performance from architectural specifications can guide design when training costs limit experimentation. Neural Spectral Capacity (NSC) provides such a heuristic, but scores projections independently and assigns identical scores to compatible reorderings of the same layer configurations. We introduce Residual-Aware Neural Spectral Capacity (ResNSC) for Transformers, using a specification-derived statistical reference for accumulated residual updates to determine the input and residual-state statistics in each projection's capacity calculation, while retaining additive evaluation without data or instantiated weights. When residual widths vary, earlier choices affect later scores, so we augment NSC's allocator (NSC-DP) with residual history beyond its layer-and-budget state. ResNSC-DP jointly allocates residual and feed-forward widths under a resource budget, with certified bounds on the best numerically evaluated ResNSC score within a specified finite candidate set. At the 121M and 374M pretraining scales, selected architectures achieve lower mean perplexity than uniform, variable-width, and tapered allocations on FineWeb-Edu, WikiText-2, and LAMBADA at comparable parameter budgets. In inherited-weight LLaMA-7B pruning using the LoNAS supernet, ResNSC-DP improves eight-task average accuracy over NSC by 3.28–5.46 percentage points across eight matched parameter budgets. Paired reversals preserve the projection shapes and NSC scores yet worsen performance in both settings, exposing performance differences hidden by position-independent scoring.

1 INTRODUCTION

Predicting a neural network's performance before training can guide architecture development, especially when foundation-model training costs limit experimentation. Empirical scaling laws, calibrated on training runs, forecast loss from model and data scale and guide compute allocation (Kaplan et al., 2020; Hoffmann et al., 2022). However, these aggregate quantities leave structural differences at a given scale unresolved: equally sized networks can differ in depth, projection dimensions, and arrangement. Neural Spectral Capacity (NSC; Zhu et al., 2026) provides a heuristic for predicting relative trained performance: it analytically computes and sums projection-level scores from matrix dimensions and prescribed initialization statistics, without data or instantiated weights. Because NSC evaluates each projection independently of its position, it assigns identical scores to compatible reorderings of the same layer configurations. Recent studies show that distributing capacity differently across depth can change language-model performance at matched parameter budgets (Wu et al., 2026; Bayat et al., 2026). These observations motivate incorporating architectural context into specification-only scoring.

The residual stream provides a concrete basis for modeling this context: in a pre-normalized Transformer, attention and feed-forward network (FFN) branches read a normalized view of the shared state and add their outputs back to it (Elhage et al., 2021; Xiong et al., 2020). Earlier branches therefore shape both the features supplied to later projections and the existing state receiving branch outputs, so identically shaped projections can operate under different conditions. In the variable-width setting considered here, branches operate on a width-dependent subset of the shared residual state, so preceding width choices determine which coordinates have been updated and how often. This

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

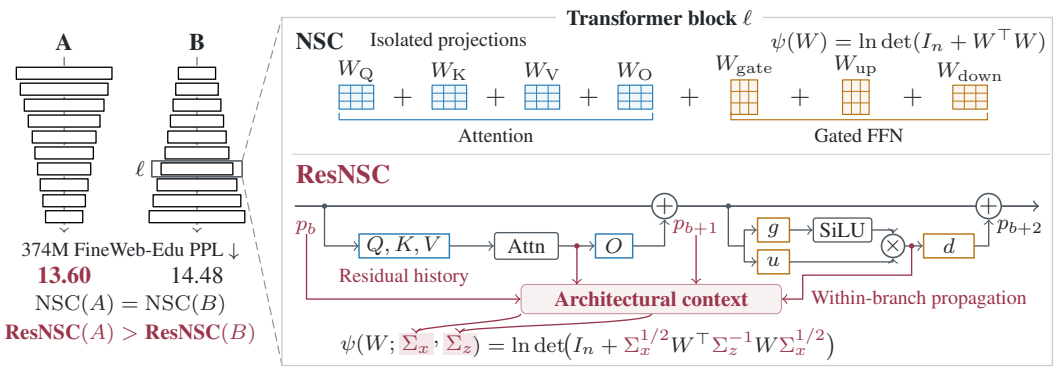

Figure 1: From NSC to ResNSC. Left: schematic ResNSC-DP allocation (A) and reversal (B), with mean 374M FineWeb-Edu perplexities. Reversal preserves projection shapes and NSC scores but changes residual context. Right: additive scoring views of block ℓ. ResNSC's specification-derived covariances use residual history and, for output projections, within-branch propagation. For $b = 2\ell - 2$, p_b, p_{b+1}, p_{b+2} are the pre-attention, post-attention, and post-FFN reference variances. Blue/orange: attention/FFN; normalization omitted.

raises the question: *how can NSC account for this residual context while remaining computable from architecture specifications alone?*

We introduce *Residual-Aware Neural Spectral Capacity* (ResNSC), which uses a specification-derived statistical reference for accumulated residual updates to determine the input and residual-state statistics in each projection's capacity calculation. For output projections, this input model includes earlier within-branch transformations (Figure 1, right). ResNSC can therefore assign different scores to reorderings of the same projections, while retaining NSC's additive aggregation and deterministic evaluation without data or instantiated weights. When residual widths vary, ResNSC's layer scores depend on preceding choices, so the original layer-and-budget state of NSC's allocator (NSC-DP) no longer suffices. ResNSC-DP augments that state with residual history to jointly allocate residual and FFN widths under a resource budget.

At the 121M and 374M pretraining scales, architectures selected by ResNSC-DP achieve lower mean perplexity than uniform, variable-width (VWT) (Wu et al., 2026), and tapered (TLM) (Bayat et al., 2026) allocations at comparable parameter budgets. Relative to the strongest of these baselines for each dataset and scale, the reductions are 0.3–0.4% on FineWeb-Edu, 2.3–2.5% on WikiText-2, and 19.6–24.9% on LAMBADA. In LLaMA-7B pruning using the LoNAS supernet (Munoz et al., 2024), ResNSC-DP improves average accuracy over eight tasks by 3.28–5.46 percentage points relative to NSC across eight matched parameter budgets. In paired controls, reversing the selected width allocations preserves the collection of projection shapes and the NSC score, yet worsens performance in both settings (Figure 1, left, illustrates the 374M case).

Our contributions are:

- **Residual context in capacity evaluation.** A specification-derived residual reference for evaluating Transformer projections in their architectural context.
- **Sufficient history for joint allocation.** Under our reference, counts of preceding layers at each residual width suffice for future score evaluation, enabling ResNSC-DP with bounds on the best ResNSC score within a specified finite candidate set.
- **Controlled evidence on layer order.** Paired reversals reveal performance differences between allocations with identical NSC scores, in both pretraining and inherited-weight pruning.

2 Related Work

Training-free architecture evaluation. Training-free architecture evaluation estimates candidate quality without fully training each model. Representative methods use activation diversity or gradient statistics at initialization (Abdelfattah et al., 2021; Mellor et al., 2021); W-PCA combines parameter

information with principal-component statistics of intermediate activations (Wang, 2025). These methods evaluate instantiated models using input samples. NSC computes a deterministic score from projection dimensions and prescribed initialization statistics, evaluating each projection independently of its position (Zhu et al., 2026). ResNSC retains this specification-only evaluation while conditioning projection-level scores on residual context.

Residual context in Transformers. The Transformer Circuits framework describes Transformer computation through branches that read and update a shared residual representation (Elhage et al., 2021). The behavior of this shared state depends on normalization, studied through signal and gradient propagation (Xiong et al., 2020), and residual mixing, which Hyper-Connections modifies through learned connections (Zhu et al., 2025). ResNSC uses this structural perspective to construct a specification-derived reference for evaluating projections. Its statistical assumptions are modeling choices made for architecture scoring, specified in Section 3.2.

Layer-wise heterogeneous Transformer architectures. Architecture search explores layer-specific Transformer configurations, as in AutoFormer (Chen et al., 2021) and FlexiBERT (Tuli et al., 2022). VWT and TLM prescribe depth-dependent profiles for whole-block and FFN widths, respectively (Wu et al., 2026; Bayat et al., 2026), while geometry-guided allocation uses forward-pass measurements of representation change (Mudarisov et al., 2026). Structured pruning also makes model-dependent choices, using gradient-based importance in LLM-Pruner (Ma et al., 2023) or learning pruning configurations in Sheared LLaMA (Xia et al., 2024). ResNSC supplies a specification-derived objective for width selection in both architecture design and inherited-weight subnetwork selection. In LoNAS (Munoz et al., 2024), ResNSC selects FFN widths while the elastic supernet supplies the selected architecture's weights. When residual widths vary, earlier choices affect later scores, so ResNSC-DP extends NSC's budget-constrained allocation with residual history.

3 Residual-Aware Neural Spectral Capacity

ResNSC extends NSC's local capacity framework with residual history and within-branch propagation. We construct reference covariances from specifications for deterministic additive scoring, then allocate widths with history-aware ResNSC-DP under a resource budget.

3.1 Preliminaries: local capacity and NSC

Local capacity. Consider a linear transformation with weight matrix $W \in \mathbb{R}^{m\times n}$. Conditional on W, we define the Gaussian reference channel $y = Wx + z$, where $x \sim \mathcal{N}(0, \Sigma_x)$ and $z \sim \mathcal{N}(0, \Sigma_z)$ are independent, with $\Sigma_x \succeq 0$ and $\Sigma_z \succ 0$. The input covariance Σ_x describes the variances and correlations of input features, while Σ_z describes the reference noise covariance. The Gaussian mutual-information identity gives (Cover & Thomas, 2006; Telatar, 1999)

$$\psi(W; \Sigma_x, \Sigma_z) := 2I(x; y \mid W) = \ln\det\Big(I_n + \Sigma_x^{1/2} W^\top \Sigma_z^{-1} W \Sigma_x^{1/2}\Big). \tag{1}$$

We use natural logarithms and prescribed reference covariances; Appendix A gives the derivation.

NSC's specification-only reference. NSC adopts isotropic inputs and unit noise, $\Sigma_x = I_n$ and $\Sigma_z = I_m$, giving its local spectral-capacity expression $\psi(W; I_n, I_m) = \psi(W) = \ln\det(I_n + W^\top W)$ (Zhu et al., 2026). For specification-only scoring, NSC evaluates this expression through a deterministic spectral formula for random-matrix ensembles specified by projection dimensions and initialization variance conventions. This yields local scores without data or instantiated candidate weights. The reference conditions do not encode projection position or residual history. ResNSC retains Equation (1) while constructing the input and noise covariances from architectural context.

3.2 Residual reference from architectural specifications

Architecture and reference assumptions. Consider L pre-normalized Transformer blocks, each containing an attention branch followed by a feed-forward network (FFN) branch. Both branches in block ℓ read a normalized view of the first d_ℓ residual coordinates and add their outputs to those coordinates; the remaining coordinates are preserved, as in VWT (Wu et al., 2026). The residual state

has width $M = \max(E, d_1, \ldots, d_L)$, where E is the embedding width. In our statistical reference, embedding coordinates are centered, uncorrelated, and have unit variance, with zeros elsewhere. Each branch adds a centered update with identity covariance on its active coordinates, uncorrelated with the initial state and other updates.

Accumulated residual variance. Index the attention and FFN branches of block ℓ by $b = 2\ell - 2$ and $b = 2\ell - 1$, respectively. Denote their active widths by $\widetilde{d}_{2\ell-2} = \widetilde{d}_{2\ell-1} = d_\ell$. The residual ledger p_b records the reference variance of each coordinate before branch b. The ledger is an auxiliary residual reference, with unit increments prescribed separately from the projection ensembles used to evaluate local capacity. Uncorrelated updates add their variances, giving

$$p_{0,i} = \mathbf{1}\{i \le E\}, \quad i = 1, \ldots, M, \qquad p_{b+1,i} = p_{b,i} + \mathbf{1}\{i \le \widetilde{d}_b\}. \tag{2}$$

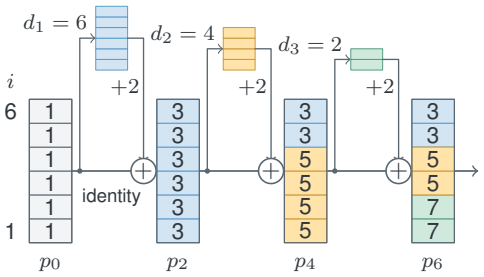

Figure 2: Illustrative reference variances ($E = M = 6$). Each block adds 2 (attention, then FFN) on its first d_ℓ coordinates; colors mark the last updating block.

Each block contributes two updates, with the FFN evaluated after the attention update (Figure 2). The ledger depends on preceding residual widths; FFN intermediate widths affect projection scores and resource costs but not these variance increments. Appendix B.1 gives an auxiliary process realizing this reference.

Reference covariances for scoring. Let $\bar{p}_b = \widetilde{d}_b^{-1} \sum_{i=1}^{\widetilde{d}_b} p_{b,i}$. Under a mean-field approximation, we replace the sample root-mean-square scale by $\sqrt{\bar{p}_b}$, giving normalized input variances $p_{b,i}/\bar{p}_b$. Read projections transform normalized residual features into branch features; write projections produce outputs added to the residual state. Normalization removes the overall variance scale while preserving relative coordinate variances. To retain sensitivity to the accumulated reference variance scale in capacity evaluation, we additionally adopt inverse-background weighting, with coordinate-wise background $1 + p_{b,i}$. This evaluates each projection relative to the accumulated residual background, so the same normalized signal receives less weight when that background is larger. We divide normalized input variances by this background for read scoring and use it as the reference noise variance for write scoring. With a shared scaling constant Z_L, the read input and write noise covariances are

$$\Sigma^{\mathrm{read}}_{x,b} = Z_L \operatorname{diag}_{i \le \widetilde{d}_b}\left(\frac{p_{b,i}}{\bar{p}_b(1 + p_{b,i})}\right), \qquad \Sigma^{\mathrm{write}}_{z,b} = Z_L^{-1} \operatorname{diag}_{i \le \widetilde{d}_b}(1 + p_{b,i}). \tag{3}$$

The unit term retains NSC's unit-noise reference. When $\bar{p}_b = 0$, we set the normalized input covariance and $\Sigma^{\mathrm{read}}_{x,b}$ to zero. The constant $Z_L = 2L / \sum_{b=0}^{2L-1} (b+2)^{-1}$ is shared by all candidates of depth L and normalizes the mean full-width reference precision (inverse noise covariance) to one (Appendix B). When every branch has width E, normalized inputs are isotropic and both $\Sigma^{\mathrm{read}}_{x,b}$ and $(\Sigma^{\mathrm{write}}_{z,b})^{-1}$ equal $Z_L/(b+2)$ times the identity. Fixed residual width makes the reference history depth-determined. When widths vary, different preceding choices can produce different coordinate-wise contexts for the same later layer.

3.3 Projection evaluation and architecture scoring

Read and Write Projections. The query, key, value, gate, and up projections use $\Sigma_x = \Sigma^{\mathrm{read}}_{x,b}$ and unit noise $\Sigma_z = I$. For each of Q, K, and V, we evaluate the stacked heads as joint observations of their shared source, retaining the per-head entry variances; Appendix C explains the distinction from separate-head evaluation. The attention output and FFN down projections receive features transformed within their branches. We model the upstream transformations by a prescribed random propagation matrix W^{in}_b, independent of the output matrix being scored. The attention reference is a $d_\ell \times d_\ell$ Gaussian matrix for value features; the FFN reference combines an $f_\ell \times d_\ell$ Gaussian matrix with an independent SiLU gate, where f_ℓ is the FFN intermediate width. For a nonzero normalized source, these reference ensembles are scaled to give unit feature variance on average. Appendix C

specifies their distributions and scaling. Conditional on W_b^{in}, propagating the normalized residual variances gives the input covariance

$$\Sigma_{x,b}^{\text{write}} = W_b^{\text{in}} \operatorname{diag}_{i \le \widetilde{d}_b} \left(\frac{p_{b,i}}{\bar{p}_b} \right) (W_b^{\text{in}})^\top, \tag{4}$$

with $\Sigma_{x,b}^{\text{write}} = 0$ when $\bar{p}_b = 0$. We pair this input covariance with the reference noise covariance $\Sigma_{z,b}^{\text{write}}$. Propagation uses the normalized variances without the additional discount applied in read scoring. Table 1 summarizes the assignments for all seven projections.

Deterministic local evaluation. For each projection, the reference capacity is the expectation of the local functional in Equation (1) over its prescribed matrix ensemble. For write terms, this expectation includes both the output matrix and the upstream reference, retaining the propagated covariance inside the log-determinant. Read and write expectations are evaluated using covariance-ensemble and product-ensemble spectral formulas, respectively, which reduce the calculation to deterministic scalar fixed-point equations. These formulas approximate the finite-dimensional ensemble expectations from architectural dimensions, residual history, and the prescribed reference conventions, without data or instantiated weights (Appendix D; Hoydis et al., 2011). Let $\widehat{\psi}_{\ell,j}(p_b)$ denote the numerical score for projection j in block ℓ with context p_b.

Table 1: Reference contexts for Equation (1): $d = d_\ell$, $f = f_\ell$. Each row uses the ledger immediately before branch b.

Branch index b	Projection j	Shape $W_{\ell,j}$	Input Σ_x	Noise Σ_z
$2\ell-2$	Q, K, V	$d \times d$	$\Sigma_{x,b}^{\text{read}}$	I_d
$2\ell-2$	O	$d \times d$	$\Sigma_{x,b}^{\text{write}}$	$\Sigma_{z,b}^{\text{write}}$
$2\ell-1$	gate, up	$f \times d$	$\Sigma_{x,b}^{\text{read}}$	I_f
$2\ell-1$	down	$d \times f$	$\Sigma_{x,b}^{\text{write}}$	$\Sigma_{z,b}^{\text{write}}$

Architecture-level aggregation. For an architecture $a = ((d_1, f_1), \ldots, (d_L, f_L))$ and projection set $\mathcal{M} = \{\text{Q}, \text{K}, \text{V}, \text{O}, \text{gate}, \text{up}, \text{down}\}$, we define the ResNSC score by summing these numerical local scores:

$$S(a) = \sum_{\ell=1}^{L} \sum_{j \in \mathcal{M}} \widehat{\psi}_{\ell,j}\big(p_{b(\ell,j)}\big) = \sum_{\ell=1}^{L} S_L^{\text{block}}(p_{2\ell-2}; d_\ell, f_\ell)\,. \tag{5}$$

Here $b(\ell, j)$ is given in Table 1. The block score $S_L^{\text{block}}(p; d, f)$ sums the seven projection terms for incoming ledger p, residual width d, and FFN width f, updating the ledger after attention before evaluating the FFN. Fixed embedding width, head count, and reference conventions are suppressed in this notation. Embedding support initializes the ledger; embeddings, normalization, and readout count toward the resource budget but not the seven projection terms. This additive architecture score can distinguish block reorderings because each term uses its own intermediate residual context, even when the final ledger agrees. When residual widths vary, preceding width choices affect subsequent block scores; the next subsection develops an allocator that retains the required history.

3.4 Budget-constrained allocation with residual history

Formulation. We jointly select residual and FFN widths to maximize the numerical ResNSC score under a resource budget. Let $\mathcal{G}$ be a finite set of specifications. Each g fixes the depth L_g, embedding width E_g, head count, reference conventions, a finite menu $\mathcal{A}_g$ of admissible (d, f) pairs, and budget B_g. We solve

$$\max_{g \in \mathcal{G}\,,\; a \in \mathcal{A}_g^{L_g}} S_g(a) \quad \text{subject to} \quad c_{\text{fixed}}(g) + \sum_{\ell=1}^{L_g} c_{g,\ell}(d_\ell, f_\ell) \le B_g, \tag{6}$$

where S_g is the score from Section 3.3 under specification g, and the resource cost comprises a fixed contribution and nonnegative additive block costs. Our experiments use integer parameter counts, including embedding/readout interfaces and normalization.

Representing residual history. For a fixed g with variable residual widths, two partial architectures (prefixes) at the same depth and spent cost can have different residual ledgers and therefore different scores for the same continuation. The layer-and-budget state used by NSC-DP is therefore insufficient for this history-dependent objective. ResNSC-DP additionally records the counts $\mathbf{n} = (n_1, \ldots, n_K)$

Algorithm 1 ResNSC-DP: allocation with residual history.

Input: Specifications $\mathcal{G}$, menus $\mathcal{A}_g$, budgets B_g, tolerance $\epsilon \geq 0$
Output: Feasible architecture and objective bounds, or infeasibility
$\mathcal{Q}$: valid live prefixes; $\mathbf{e}_{r(d)}$: width-d count increment; burgundy: residual-history operations.
Construct completion bounds U by history-indexed backward DP (Appendix E.3).
Initialize feasible roots $\mathcal{Q}$ and an incumbent with score LB; return infeasible if none exists.
while $\exists \pi \in \mathcal{Q} : U(\pi) > \mathrm{LB} + \epsilon$ and search limits allow **do**
  Remove $\pi = (g, \mathbf{n}, c, s)$ with largest $U(\pi)$ from $\mathcal{Q}$.
  Reconstruct $p \leftarrow p_g(\mathbf{n})$ and $t \leftarrow \sum_r n_r$.
  for each $(d, f) \in \mathcal{A}_g$ **do**
    $s' \leftarrow s + S^{\mathrm{block}}_{L_g}(p; d, f)$.
    $\mathbf{n}' \leftarrow \mathbf{n} + \mathbf{e}_{r(d)}$, $c' \leftarrow c + c_{g,t+1}(d, f)$; $\pi' \leftarrow (g, \mathbf{n}', c', s')$.
    if no affordable completion, dominated within $(g, \mathbf{n}')$, or $U(\pi') \leq \mathrm{LB}$ **then continue**.
    if $t + 1 = L_g$ **then** update the incumbent and LB only on improvement.
    else insert π' into $\mathcal{Q}$ and update the cost–score frontier at $(g, \mathbf{n}')$.
  end for
end while
return incumbent architecture and $(\mathrm{LB}, \mathrm{UB})$, where $\mathrm{UB} = \max\big(\{\mathrm{LB}\} \cup \{U(\pi) : \pi \in \mathcal{Q}\}\big)$.

of preceding blocks at each residual width $d^{(1)} < \cdots < d^{(K)}$. Under the two-update reference, these counts reconstruct the ledger:

$$p_{g,i}(\mathbf{n}) = \mathbf{1}\{i \leq E_g\} + 2\sum_{r=1}^{K} n_r \mathbf{1}\{i \leq d^{(r)}\}, \qquad t = \sum_{r=1}^{K} n_r. \tag{7}$$

The counts determine both the current ledger and the next block index. Appending the same choice gives the same block score and next count vector, so equal counts imply identical additional scores for every common continuation. Under the stated menus and additive resource constraint, feasible continuations also agree at the same remaining budget. We store each prefix as $\pi = (g, \mathbf{n}, c, s)$, where c is spent variable cost and s is accumulated score. Within the same $(g, \mathbf{n})$, a prefix with no greater cost and no smaller score dominates another. The counts retain the history needed for future evaluation, while s retains the effects of past order. FFN choices enter through c and s (Appendix E.2).

Search with completion bounds. ResNSC-DP uses backward dynamic programming to construct completion bounds: upper bounds on the total score of any feasible architecture extending a prefix. These bounds guide a search over nondominated prefixes. For each nonnegative resource price λ, the backward DP maximizes additional score minus λ times additional cost, retaining history transitions while relaxing the hard budget. Adding the earned score s and λ times the remaining budget $B_g - c_{\mathrm{fixed}}(g) - c$ gives an upper bound on the score of any feasible completion. The tightest bound across the prescribed prices defines $U(\pi)$. Initialization tests minimum-cost architectures to find an initial feasible solution or establish infeasibility. Algorithm 1 expands the prefix with largest U and discards infeasible or dominated prefixes and those whose bounds cannot improve the best feasible complete score found so far. Appendix E gives the bound recurrence and detailed procedure (Algorithm 2).

Optimality guarantees. Let S^* be the optimum of the numerical objective in Equation (6). For a nonempty feasible set, under exact arithmetic for the fixed numerical objective and with admissible completion bounds, the best feasible architecture found so far supplies $\mathrm{LB} = S_{\widehat{g}}(\widehat{a})$. At initialization and after each complete prefix expansion, the maximum of LB and the completion bounds of all remaining valid prefixes supplies UB, with

$$\mathrm{LB} \leq S^* \leq \mathrm{UB}, \qquad \mathrm{UB} - \mathrm{LB} \leq \epsilon \implies S^* - \mathrm{LB} \leq \epsilon. \tag{8}$$

A gap of at most ϵ establishes ϵ-optimality, and exhausting the valid queue establishes optimality. If search stops earlier, the bounds quantify the remaining objective gap. Appendix E gives the proof and implementation details. With fixed residual width, depth alone determines the ledger. Precomputing block scores by layer and FFN width recovers ordinary layer-and-budget dynamic programming with ResNSC's contextual score (Appendix E.5).

4 EXPERIMENTS

We evaluate whether residual context captures consequential architectural differences, improves resource allocation, and retains NSC's global ranking utility. Experiments cover from-scratch pretraining at 121M and 374M, inherited-weight pruning of LoNAS-LLaMA-7B, and ranking across language and vision search spaces.

4.1 ARCHITECTURE SEARCH AND PRETRAINING

Setup. We study 121M and 374M decoder-only Transformers with 16 and 24 blocks, jointly varying each block's residual width d_ℓ and FFN width f_ℓ without a prescribed taper. ResNSC-DP selects architectures from specifications alone under the Uniform model's total-parameter ceiling; NSC-DP (Zhu et al., 2026) uses the same menus and ceiling. We also compare constant-width Uniform, whole-block bottleneck VWT (Wu et al., 2026), and FFN-tapered TLM (Bayat et al., 2026) profiles at comparable parameter counts. Reverse reverses the ResNSC-DP sequence of (d_ℓ, f_ℓ) pairs, preserving projection shapes and the NSC score. Model specifications, menus, and configurations appear in Appendix G, and search certificates in Appendix E.6.

All architectures within a scale are trained from scratch on the same FineWeb-Edu subset (Penedo et al., 2024), with a shared μP/AdamW recipe and token budgets of 2.662B at 121M and 8.228B at 374M. We evaluate held-out FineWeb-Edu loss and perplexity, WikiText-2 word perplexity, LAMBADA perplexity, and eight-task zero-shot accuracy at 374M. Means and sample standard deviations use three independent runs for the 121M main comparisons. The 374M experiments and 121M ablations use one full run plus two final-decay continuations, with standard deviations measuring annealing variation (Appendix F).

Table 2: Language-modeling results after final decay at 121M and 374M: FineWeb-Edu perplexity (PPL), WikiText-2 word perplexity, LAMBADA perplexity, and, at 374M, mean eight-task zero-shot accuracy (Avg8). 121M: means and sample standard deviations over three independent runs; 374M: one run plus two final-decay continuations, so standard deviations reflect annealing variation only. Bold: best per column; per-task accuracies in Appendix F.

	121M			374M			
Method	PPL↓	WikiText-2↓	LAMBADA↓	PPL↓	WikiText-2↓	LAMBADA↓	Avg8↑
ResNSC (ours)	**19.39** (±0.01)	**51.13** (±0.21)	**202.8** (±13.5)	**13.60** (±0.002)	**31.20** (±0.03)	**53.0** (±0.4)	**44.60** (±0.07)
uniform	20.25 (±0.02)	55.47 (±0.07)	262.9 (±13.7)	14.27 (±0.002)	33.82 (±0.05)	76.2 (±2.0)	44.07 (±0.19)
VWT	19.48 (±0.06)	52.33 (±0.15)	252.6 (±35.7)	13.65 (±0.002)	31.99 (±0.08)	70.6 (±2.1)	44.26 (±0.11)
TLM	20.38 (±0.02)	55.43 (±0.53)	252.1 (±8.5)	14.43 (±0.001)	34.19 (±0.02)	78.1 (±0.6)	43.73 (±0.05)
NSC	20.00 (±0.03)	53.40 (±0.31)	221.4 (±10.1)	13.82 (±0.003)	32.40 (±0.11)	66.9 (±1.9)	44.26 (±0.09)
Reverse	20.27 (±0.05)	53.22 (±0.55)	234.6 (±9.3)	14.48 (±0.001)	33.93 (±0.01)	77.7 (±1.6)	42.86 (±0.18)

Table 3: Ablating the architecture score at 121M. All variants share the search space and parameter budget; profiles show the widest and final active widths. Rows 2–4 simplify residual history (none, branch position only, coordinate average); the last row removes the propagated write-source spectrum. Entries are one run (seed 42) plus two final-decay continuations; standard deviations reflect annealing variation only. Bold: best per column.

Scoring rule	Profile $\max_\ell d_\ell \to d_{16}$	PPL↓	WikiText-2↓	LAMBADA↓
ResNSC (full)	$1280 \to 480$	**19.39** (±0.01)	**51.04** (±0.20)	211.2 (±2.3)
w/o residual context	$800 \to 640$ (identical to NSC)	20.00 (±0.01)	53.25 (±0.21)	233.6 (±1.7)
Depth only	$1760 \to 320$	19.85 (±0.01)	52.85 (±0.21)	205.9 (±2.5)
Mean history only	$1920 \to 320$	20.20 (±0.01)	54.61 (±0.20)	229.5 (±2.7)
w/o propagated spectrum	$1440 \to 320$	19.72 (±0.01)	52.72 (±0.20)	**197.5** (±3.3)

Results. Reversing the ResNSC-DP allocation increases mean FineWeb-Edu perplexity from 19.39 to 20.27 at 121M and from 13.60 to 14.48 at 374M, despite unchanged NSC scores (Table 2). WikiText-2 and LAMBADA perplexities also increase at both scales. These differences show that the

placement of the same projection shapes affects trained performance, exposing a limitation of NSC's order-insensitive aggregation.

Architectures selected by ResNSC-DP achieve the lowest mean perplexities across all three datasets and both scales in Table 2. Against NSC-DP, FineWeb-Edu perplexity decreases from 20.00 to 19.39 at 121M and from 13.82 to 13.60 at 374M. At 374M, the selected architecture also achieves the highest mean eight-task accuracy, 44.60%, versus 44.26% for the strongest comparators, NSC and VWT (Appendix F).

At 121M, we test removing residual context, retaining only branch position or mean reference variance, and removing the propagated spectrum, with each variant selecting an architecture under the same menus and parameter ceiling (Table 3; definitions in Appendix F.1). Full ResNSC achieves the lowest mean FineWeb-Edu and WikiText-2 perplexity, supporting coordinate-wise residual history and within-branch propagation for these evaluations. The depth-only and no-propagation variants achieve lower LAMBADA perplexity.

4.2 Structured pruning of LoNAS-LLaMA-7B

Setup. Following NSC (Zhu et al., 2026), we select subnetworks from the LoNAS LLaMA-7B supernet (Munoz et al., 2024) with 32 blocks, fixed residual width 4,096, and variable FFN widths and Q/K/V LoRA ranks. Subnetworks inherit weights from the same supernet using the same channel-slicing procedure. We report Avg8, the mean accuracy over eight commonsense tasks; Appendix H gives the menus, tasks, and LoRA scoring extension.

ResNSC selects from architecture specifications without candidate weights or calibration inputs. The fixed-width specialization of ResNSC-DP uses the same exact layer-and-budget recurrence as NSC-DP, with contextual layer scores. Both methods use identical menus under eight shared parameter ceilings; their selected subnetworks have matching parameter counts at each ceiling. We include the SNIP (Lee et al., 2019), SynFlow (Tanaka et al., 2020), GradNorm (Abdelfattah et al., 2021), and W-PCA (Wang, 2025) evolutionary-search results reported by NSC under a common 5.7B parameter ceiling. Reverse assigns the highest-budget ResNSC FFN widths in reverse order across the original supernet layers, preserving parameter count, FLOPs, and NSC score.

Results. Reverse achieves 59.49% Avg8 versus 68.87% for ResNSC, a 9.38-point difference at the same parameter count and NSC score (Table 4). The placement of retained FFN widths therefore also affects performance under weight inheritance.

At 5.690B parameters, ResNSC exceeds NSC by 3.40 points and the strongest proxy baseline, W-PCA, by 5.04 points in Avg8 (Table 4). This allocation retains 98.8% of the unpruned supernet's Avg8 while using approximately 16% fewer parameters. Across all eight matched NSC budgets, the gain is 3.28–5.46 points (Figure 3).

In this fixed-width setting, layer scoring and selection across all eight budgets take under one second, excluding supernet training and subnetwork evaluation. The NSC study reports subsecond search for NSC and 22–45 minutes for proxy baselines (Table 4).

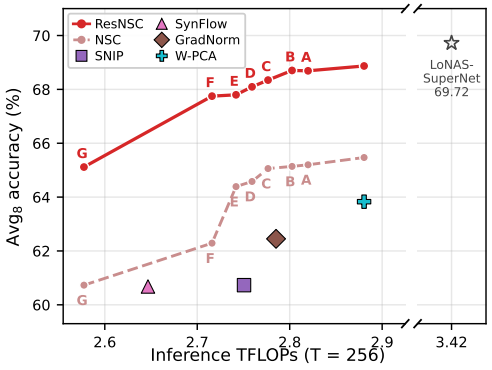

Figure 3: ResNSC improves the accuracy–compute trade-off. Avg8 versus TFLOPs ($T = 256$) across eight matched budgets (A–G: smaller budgets), with proxy baselines and the unpruned supernet (star). Full results in Appendix H.

4.3 Ranking across architecture search spaces

Setup. We examine whether incorporating residual context retains NSC's global ranking utility while distinguishing more candidate architectures. Following the NSC protocol (Zhu et al., 2026), we compare the scores on 500 FlexiBERT architectures (Serianni & Kalita, 2023), 200 GPT-2 architectures (Radford et al., 2019), and 1,001 AutoFormer-Small candidates (Chen et al., 2021),

Table 4: LoNAS LLaMA-7B pruning on eight commonsense tasks (accuracy, %). ResNSC and NSC use the same width/rank menu and matched parameter counts. Params reports each selected model's size. The supernet is a reference; **bold** marks the best pruned result per column. Reverse reassigns FFN widths in reverse order across the original layers, preserving parameter count and NSC score. Cost is search-stage wall-clock time.

Method	Params	BoolQ	PIQA	SIQA	HSwag	WGrnd	ARC-e	ARC-c	OBQA	Avg_8	Cost
LoNAS-SuperNet	6.74 B	66.02	77.58	72.93	57.49	66.85	78.41	62.29	76.20	69.72	—
SNIP	5.44 B	61.93	69.21	64.99	41.26	62.04	66.50	52.47	67.40	60.73	22 min
SynFlow	5.23 B	61.13	70.24	63.25	45.94	59.98	66.84	51.88	66.20	60.68	23 min
GradNorm	5.50 B	61.31	70.78	67.35	46.33	62.59	69.02	55.03	67.20	62.45	22 min
W-PCA	5.69 B	63.82	70.95	67.91	49.24	62.75	69.57	56.40	70.00	63.83	45 min
NSC	5.69 B	**64.01**	72.85	68.94	51.30	64.72	72.43	58.70	70.80	65.47	$<$1 s
ResNSC	5.69 B	61.71	**76.71**	**70.98**	**58.39**	**67.01**	**78.45**	**61.69**	**76.00**	**68.87**	$<$1 s
Reverse	5.69 B	61.25	67.46	63.10	42.64	61.17	65.24	51.45	63.60	59.49	—

using identical candidates, trained-performance labels, and specifications. Table 5 reports rank correlations and distinct-score counts; Appendix I gives the protocol and supplementary results.

Results. ResNSC retains strong global ranking correlations: Spearman ρ is comparable to NSC on FlexiBERT and GPT-2 and higher on AutoFormer-Small (Table 5); it is lower on AutoFormer-Tiny (Appendix I). Distinct-score counts increase from 297 to 421 on FlexiBERT and from 952 to 1,001 on AutoFormer-Small; both assign distinct scores to all 200 GPT-2 candidates. Together with the paired reversals in Sections 4.1–4.2, these distinctions suggest that architectural context can help guide resource allocation.

Table 5: Global ranking correlations and number of distinct scores.

Benchmark	N	τ_b (NSC / ResNSC)	ρ (NSC / ResNSC)	Distinct (NSC / ResNSC)
FlexiBERT	500	0.695 / 0.692	0.884 / 0.883	297 / 421
GPT-2	200	0.849 / 0.854	0.968 / 0.970	200 / 200
AutoFormer-Small	1,001	0.658 / 0.719	0.852 / 0.907	952 / 1,001

5 Conclusion

We introduced Residual-Aware Neural Spectral Capacity (ResNSC), which scores each Transformer projection in the residual context implied by its architecture, using a reference constructed from the architecture specification alone. Because earlier width choices change later scores, ResNSC-DP extends NSC-DP with a residual-history state and bounds the best score within a specified finite candidate set.

Paired reversals show that allocations with identical projection shapes and NSC scores perform differently after from-scratch training and under weight inheritance. Optimizing ResNSC yields lower perplexity than uniform, VWT, TLM, and same-menu NSC allocations at 121M and 374M, and higher eight-task average accuracy than NSC at every matched LoNAS budget. Across broader search spaces, ResNSC retains strong global ranking correlations while distinguishing more candidate architectures. ResNSC keeps NSC's additive, data-free evaluation, providing a basis for studying other architecture families and larger candidate sets. Beyond the residual stream, conditioning each projection on further elements of the computational graph, such as attention mixing across positions and expert routing, is the next step toward architecture-faithful capacity evaluation from the specification alone.

AI use statement

Generative AI tools were used to assist with manuscript drafting and language editing, code development, and checking mathematical derivations. The authors take responsibility for the final manuscript, including its claims, results, and code.

Ethics statement

This work studies neural architecture scoring and capacity allocation using existing datasets and pretrained models. We have not identified ethical issues specific to this study that require special consideration.

Reproducibility statement

Section 3 defines the ResNSC score, reference model, and search algorithm; Appendices A–E provide the assumptions, derivations, and proofs. Section 4 and Appendices F–I describe the search spaces, training and evaluation settings, repeated runs, and statistical reporting. The appendices also list the selected layer-wise architecture configurations and pruning results at each budget to support implementation and experimental reproduction.

References

Mohamed S. Abdelfattah, Abhinav Mehrotra, Łukasz Dudziak, and Nicholas D. Lane. Zero-cost proxies for lightweight NAS. In *International Conference on Learning Representations*, 2021.

Reza Bayat, Ali Behrouz, and Aaron Courville. Tapered language models, 2026. URL https://arxiv.org/abs/2606.23670.

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*, volume 34, 2020.

Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. AutoFormer: Searching transformers for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019.

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, 2018. URL https://arxiv.org/abs/1803.05457.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, 2 edition, 2006.

Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *International Conference on Learning Representations*, 2024.

Nelson Elhage et al. A mathematical framework for transformer circuits. Transformer Circuits Thread, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

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, 12 2023.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Jakob Hoydis, Romain Couillet, and Mérouane Debbah. Iterative deterministic equivalents for the performance analysis of communication systems, 2011. URL https://arxiv.org/abs/1112.4167.

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.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. URL https://arxiv.org/abs/2001.08361.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. LLM-Pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems*, 2023.

Vladimir A. Marčenko and Leonid A. Pastur. Distribution of eigenvalues for some sets of random matrices. *Mathematics of the USSR-Sbornik*, 1(4):457–483, 1967.

Joe Mellor, Jack Turner, Amos Storkey, and Elliot J. Crowley. Neural architecture search without training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*. PMLR, 2021.

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

Timur Mudarisov, Mikhail Burtsev, and Radu State. Geometry-guided layerwise FFN width allocation in transformers. arXiv preprint arXiv:2608.02064, 2026. URL https://arxiv.org/abs/2608.02064.

Juan Pablo Munoz, Jinjie Yuan, Yi Zheng, and Nilesh Jain. LoNAS: Elastic low-rank adapters for efficient large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. ELRA and ICCL, 2024.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems*, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI, 2019.

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*, volume 34, 2020.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019.

Aaron Serianni and Jugal Kalita. Training-free neural architecture search for RNNs and transformers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2023.

Noam Shazeer. GLU Variants Improve Transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding, 2021. URL https://arxiv.org/abs/2104.09864.

Hidenori Tanaka, Daniel Kunin, Daniel L. K. Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Emre Telatar. Capacity of multi-antenna Gaussian channels. *European Transactions on Telecommunications*, 10(6):585–595, 1999.

Shikhar Tuli, Bhishma Dedhia, Shreshth Tuli, and Niraj K. Jha. FlexiBERT: Are current transformer architectures too homogeneous and rigid?, 2022. URL https://arxiv.org/abs/2205.11656.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Shang Wang. W-PCA based gradient-free proxy for efficient search of lightweight language models. In *International Conference on Learning Representations*, 2025.

Zhaofeng Wu, Oliver Sieberling, Shawn Tan, Rameswar Panda, Yury Polyanskiy, and Yoon Kim. Variable-width transformers, 2026. URL https://arxiv.org/abs/2606.18246.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *International Conference on Learning Representations*, 2024.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*. PMLR, 2020.

Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor Programs V: Tuning large neural networks via zero-shot hyperparameter transfer. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

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*. Association for Computational Linguistics, 2019.

Biao Zhang and Rico Sennrich. Root Mean Square Layer Normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Chenyu Zhu, Ruoyu Zhao, and Zhichao Lu. Neural spectral capacity: Measuring and designing architectures from network specification alone. In *The Fortieth Annual Conference on Neural Information Processing Systems*, 2026.

Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. Hyper-connections. In *International Conference on Learning Representations*, 2025.

A NOTATION AND THE LOCAL CAPACITY IDENTITY

The main text uses ψ for local capacity, $\widehat{\psi}$ for its deterministic spectral evaluation, S_L^{block} for a block sum, and S for the architecture sum. Superscripts read and write distinguish the two reference-ensemble evaluations of ψ. Table 6 fixes the notation used throughout these appendices.

Table 6: Notation shared by the main text and mathematical appendices.

Symbol	Meaning
ℓ, b	Block index and branch index; attention has $b = 2\ell - 2$, FFN has $b = 2\ell - 1$.
E, L, H, M	Embedding width, number of blocks, number of attention heads, and complete residual-state width.
d_ℓ, f_ℓ	Active residual width and FFN intermediate width of block ℓ.
$\widetilde{d}_b, k_b$	Active residual width (shared by reading and writing) and latent width of branch b.
h_b, p_b	Actual residual state and specification-derived reference-power ledger before branch b.
Σ_x, Σ_z	Generic source and noise covariances in $\psi(W; \Sigma_x, \Sigma_z)$.
$\Sigma_{x,b}^{\mathrm{read}}, \Sigma_{x,b}^{\mathrm{write}}$	Read source covariance and propagated write source covariance.
$\Sigma_{z,b}^{\mathrm{write}}$	Reference noise covariance for write scoring.
W_b^{in}	Normalized upstream reference, including the gate when present.
$\Sigma_{x,b}^{\mathrm{in}}$	Normalized branch-input covariance (Appendices B–D); it has no read discount.
X_b, G_b	Gaussian factor and diagonal gate used to construct $W_b^{\mathrm{in}} = G_b X_b$ in the appendices.
$W_{\ell,j}$	Scored projection of type j in block ℓ.
Z_L	Shared normalization of the full-width reference precision at depth L.
$\psi, \widehat{\psi}, S_L^{\mathrm{block}}, S$	Local functional, deterministic local evaluation, block sum, and architecture sum.
$\mathbf{n}, n_r, t$	Residual-width count vector, count of width option r, and completed block count.
$g, \mathcal{A}_g, B_g$	Fixed network specification, its block-choice menu, and resource budget (parameter ceiling in our experiments).
c, s	Spent variable resource cost and accumulated prefix score.
$\lambda, V_{g,\lambda}, U$	Resource price, relaxed suffix value, and prefix upper bound.

A.1 GAUSSIAN IDENTITY, INCLUDING SINGULAR SOURCES

Fix $W \in \mathbb{R}^{m\times n}$, $\Sigma_x \succeq 0$, and $\Sigma_z \succ 0$. Let

$$x \sim \mathcal{N}(0, \Sigma_x), \qquad z \sim \mathcal{N}(0, \Sigma_z), \qquad x \perp z, \qquad y = Wx + z. \tag{A.1}$$

The output covariances are $\mathrm{Cov}(y \mid W) = W\Sigma_x W^\top + \Sigma_z$ and $\mathrm{Cov}(y \mid x, W) = \Sigma_z$. Subtracting their Gaussian differential entropies gives (Cover & Thomas, 2006; Telatar, 1999)

$$\begin{aligned} 2I(x; y \mid W) &= \ln\det(W\Sigma_x W^\top + \Sigma_z) - \ln\det(\Sigma_z) \\ &= \ln\det(I_m + \Sigma_z^{-1/2} W \Sigma_x W^\top \Sigma_z^{-1/2}) \\ &= \ln\det(I_n + \Sigma_x^{1/2} W^\top \Sigma_z^{-1} W \Sigma_x^{1/2}). \end{aligned} \tag{A.2}$$

The last equality is Sylvester's determinant identity. The argument remains valid for singular Σ_x because both output covariances are positive definite. The quantity is twice the mutual information for the prescribed Gaussian source.

For a random reference projection, we average this pointwise expression over its ensemble. For a write projection, the source covariance is itself generated by W_b^{in}. The identity is then applied conditional on these factors before taking their expectation.

B THE RESIDUAL REFERENCE AND ITS INDUCED METRICS

B.1 RESIDUAL WINDOWS AND A PROCESS REALIZING THE LEDGER

Let $A_u \in \mathbb{R}^{u\times M}$ select the first u residual coordinates. A branch with active residual width $\widetilde{d}_b$ has the actual residual form

$$h_{b+1} = h_b + A_{\widetilde{d}_b}^\top F_b\Big(\mathrm{Norm}(A_{\widetilde{d}_b} h_b)\Big). \tag{B.1}$$

This parameter-free selection and insertion preserves coordinates outside the write window, as also used in VWT (Wu et al., 2026). Both branches of block ℓ read and write its first d_ℓ coordinates.

To construct an architecture-only reference, introduce an auxiliary process $\widetilde{h}_b$:

$$\begin{aligned} \widetilde{h}_0 &\sim \mathcal{N}\big(0, \mathrm{diag}_{i\le M}\, \mathbf{1}\{i \le E\}\big)\,, \\ \widetilde{h}_{b+1} &= \widetilde{h}_b + A_{\widetilde{d}_b}^{\top}\varepsilon_b, \qquad\qquad \varepsilon_b \sim \mathcal{N}(0, I_{\widetilde{d}_b}). \end{aligned} \tag{B.2}$$

The initial state and all increments are independent. Consequently,

$$\mathrm{Cov}(\widetilde{h}_b) = \mathrm{diag}(p_b), \qquad \mathrm{Cov}(\widetilde{h}_{b+1}) = \mathrm{diag}(p_b) + A_{\widetilde{d}_b}^{\top} A_{\widetilde{d}_b}, \tag{B.3}$$

which gives Equation (2).

An intermediate FFN width changes neither the support nor the unit variance assigned to a reference write. The ledger therefore depends on previous residual-write windows, while FFN width enters the projection capacities and parameter costs. Within a search specification, we zero-pad ledgers to the maximum width allowed by its menu.

B.2 Source normalization and inverse-background evaluation

On a read window of width $\widetilde{d}_b$, the expected squared norm of the auxiliary state is $\widetilde{d}_b \bar{p}_b$. For the appendix derivations, denote the normalized branch-input covariance by $\Sigma_{x,b}^{\mathrm{in}}$. Replacing the sample root-mean-square normalization factor by $\sqrt{\bar{p}_b}$ yields

$$\mathrm{Cov}\left(\frac{A_{\widetilde{d}_b}^{\top}\widetilde{h}_b}{\sqrt{\bar{p}_b}}\right) = \frac{\mathrm{diag}(p_{b,1:\widetilde{d}_b})}{\bar{p}_b} = \Sigma_{x,b}^{\mathrm{in}}, \qquad \bar{p}_b > 0. \tag{B.4}$$

This gives the mean-field normalized covariance. When $\bar{p}_b = 0$, all source coordinates have zero reference power and we define $\Sigma_{x,b}^{\mathrm{in}} = 0$.

For evaluation, we weight by the inverse of the sum of accumulated background and unit noise. Because all involved matrices are diagonal, the reference covariances in Equation (3) are equivalently

$$\begin{aligned} \Sigma_{x,b}^{\mathrm{read}} &= Z_L(\Sigma_{x,b}^{\mathrm{in}})^{1/2}\big(I_{\widetilde{d}_b} + \mathrm{diag}(p_{b,1:\widetilde{d}_b})\big)^{-1}(\Sigma_{x,b}^{\mathrm{in}})^{1/2}, \\ \Sigma_{z,b}^{\mathrm{write}} &= Z_L^{-1}\big(I_{\widetilde{d}_b} + \mathrm{diag}(p_{b,1:\widetilde{d}_b})\big). \end{aligned} \tag{B.5}$$

The branch propagates $\Sigma_{x,b}^{\mathrm{in}}$; the read covariance and write noise covariance apply inverse-background weighting during evaluation.

B.3 Depth normalization and the scalar specialization

For the full-width reference with $\widetilde{d}_b = E$ for every branch, every coordinate in the window has $p_{b,i} = b + 1$. The unnormalized receiver precisions are $(b+2)^{-1}$, so setting their mean to one gives

$$\begin{aligned} Z_L &= \frac{2L}{\sum_{b=0}^{2L-1}(b+2)^{-1}}, \qquad \Sigma_{x,b}^{\mathrm{in}} = I_E, \\ \Sigma_{x,b}^{\mathrm{read}} &= (\Sigma_{z,b}^{\mathrm{write}})^{-1} = \frac{Z_L}{b+2} I_E. \end{aligned} \tag{B.6}$$

Every candidate of depth L uses the same Z_L. In block ℓ, the attention and FFN precisions are $Z_L/(2\ell)$ and $Z_L/(2\ell+1)$, respectively, reflecting the intervening attention write.

The harmonic form follows from unit increments and inverse-background evaluation. For a fixed active width different from E, the ledger is likewise depth-determined and is obtained directly from Equation (2).

B.4 Source support and the mean propagated covariance

To expose the upstream ensemble in the appendix, write $W_b^{\mathrm{in}} = G_b X_b$, with normalized Gaussian factor $X_b \in \mathbb{R}^{k_b \times \widetilde{d}_b}$ and real diagonal gate G_b. Here k_b is the input width of the scored output projection. Attention uses $k_b = \widetilde{d}_b$ and $G_b = I$; the FFN uses $k_b = f_\ell$ and the gate specified in Appendix C.2. Thus Equation (4) is $\Sigma_{x,b}^{\mathrm{write}} = G_b X_b \Sigma_{x,b}^{\mathrm{in}} X_b^\top G_b$.

The diagonal factors in Equation (B.5) are positive on the source support, so $\Sigma_{x,b}^{\mathrm{read}}$ and $\Sigma_{x,b}^{\mathrm{in}}$ have the same rank. If $r = \operatorname{rank} \Sigma_{x,b}^{\mathrm{in}}$, then

$$\operatorname{rank}(W \Sigma_{x,b}^{\mathrm{read}} W^\top) \le r, \qquad \operatorname{rank}(G_b X_b \Sigma_{x,b}^{\mathrm{in}} X_b^\top G_b) \le \min(k_b, r). \tag{B.7}$$

Expanding the first read window beyond E therefore leaves its source rank at most E. A wider output can distribute features over a larger destination without increasing the current input rank. Subsequent reference writes update the ledger through Equation (2).

For nonzero $\Sigma_{x,b}^{\mathrm{in}}$, $\operatorname{tr} \Sigma_{x,b}^{\mathrm{in}} = \widetilde{d}_b$. Independent Gaussian entries of X_b with variance $1/\widetilde{d}_b$ satisfy

$$\begin{aligned} \mathbb{E}_{X_b}[X_b \Sigma_{x,b}^{\mathrm{in}} X_b^\top] &= \frac{\operatorname{tr} \Sigma_{x,b}^{\mathrm{in}}}{\widetilde{d}_b} I_{k_b} = I_{k_b}, \\ \mathbb{E}_{X_b, G_b}[\Sigma_{x,b}^{\mathrm{write}}] &= \mathbb{E}_{G_b}[G_b I_{k_b} G_b] = I_{k_b}, \end{aligned} \tag{B.8}$$

where the second line uses independence and unit second moment of every gate entry. When $\Sigma_{x,b}^{\mathrm{in}} = 0$, the propagated covariance is zero instead. ResNSC evaluates each propagated covariance inside the log-determinant before averaging.

C Projection-specific reference ensembles

This appendix specifies the reference ensembles for the common capacity functional. For local formulas we suppress the branch subscript on $\Sigma_{x,b}^{\mathrm{in}}, \Sigma_{x,b}^{\mathrm{read}}, \Sigma_{z,b}^{\mathrm{write}}, X_b, G_b$ and write the scored projection as W. In a write term, W denotes the output projection and X its normalized upstream reference.

C.1 Complete reads and inherited entry variances

For $W \in \mathbb{R}^{m \times d}$ with independent $\mathcal{N}(0, v)$ entries, the read-ensemble evaluation is

$$\psi_v^{\mathrm{read}}(\Sigma_x^{\mathrm{read}}; m) := \mathbb{E}_W \psi(W; \Sigma_x^{\mathrm{read}}, I_m) = \mathbb{E}_W \ln\det(I_m + W \Sigma_x^{\mathrm{read}} W^\top). \tag{C.1}$$

Its deterministic evaluation is denoted $\widehat{\psi}_v^{\mathrm{read}}(\Sigma_x^{\mathrm{read}}; m)$. Table 7 gives the reference variances for all seven projections.

Table 7: Projection and propagation conventions for a standard block, with $d = d_\ell$ and $f = f_\ell$.

Projection	Scored matrix shape	Entry variance	Upstream reference for a write
Q, K, V	$d \times d$	$2/(d + d/H)$	None
O	$d \times d$	$1/d$	$k = d$, $\mathrm{Var}(X_{ij}) = 1/d$, $G = I_d$
gate, up	$f \times d$	$2/(d + f)$	None
down	$d \times f$	$2/(d + f)$	$k = f$, $\mathrm{Var}(X_{ij}) = 1/d$, normalized gate

For attention, each complete Q, K, or V matrix vertically stacks H head matrices of shape $(d/H) \times d$. Each stacked projection retains the per-head entry variance $2/(d + d/H)$ and is evaluated as one complete observation. The number of heads and admissible widths must make d/H an integer in this standard attention setting.

Joint observation of a shared source. Let $x \sim \mathcal{N}(0, \Sigma_x^{\mathrm{read}})$ and let head h observe $y_h = W_h x + z_h$, with mutually independent unit Gaussian receiver noises, independent of x. Define $K_h = (\Sigma_x^{\mathrm{read}})^{1/2} W_h^\top W_h (\Sigma_x^{\mathrm{read}})^{1/2} \succeq 0$. Stacking the observations gives

$$\psi^{\mathrm{joint}} = \ln\det\left(I_d + \sum_{h=1}^{H} K_h\right), \qquad \psi^{\mathrm{separate}} = \sum_{h=1}^{H} \ln\det(I_d + K_h). \tag{C.2}$$

For positive semidefinite K_1, K_2,

$$\begin{aligned} \ln\det(I + K_1 + K_2) &= \ln\det(I + K_1) + \ln\det\Big(I + K_2^{1/2}(I + K_1)^{-1} K_2^{1/2}\Big) \\ &\le \ln\det(I + K_1) + \ln\det(I + K_2). \end{aligned} \tag{C.3}$$

The equality follows from determinant factorization and Sylvester's identity; the inequality uses $(I + K_1)^{-1} \preceq I$ and monotonicity of the log-determinant. Induction gives $\psi^{\mathrm{joint}} \le \psi^{\mathrm{separate}}$, pointwise and hence in expectation under matched ensembles. Exact recovery of an original NSC implementation uses its projection decomposition and variance conventions together with isotropic local metrics.

C.2 Normalized propagation and the gated FFN reference

A write projection $W \in \mathbb{R}^{d\times k}$ receives the source $\Sigma_x^{\mathrm{write}} = GX\Sigma_x^{\mathrm{in}} X^\top G$. Its normalized upstream factor has shape $X \in \mathbb{R}^{k\times d}$ and independent entries with variance $v_{\mathrm{in}} = 1/d$. The scored matrix has independent entries with variance $v_{\mathrm{out}} = 2/(d+k)$. The factors X, G, W are independent in this reference.

For attention, $k = d$ and $G = I_d$ define a normalized value-feature-to-output reference. For the gated FFN, $k = f$ and the independent diagonal entries of G follow the normalized SiLU law

$$\eta = \frac{\mathrm{SiLU}(\sqrt{q}\, z)}{\sqrt{\mathbb{E}_{z\sim\mathcal{N}(0,1)}[\mathrm{SiLU}(\sqrt{q}\, z)^2]}}, \quad z \sim \mathcal{N}(0,1), \quad q = \frac{2d}{d+f}, \quad \mu_{d,f} = \mathcal{L}(\eta^2). \tag{C.4}$$

The mean-field preactivation variance is $q = \frac{2}{d+f}\operatorname{tr}\Sigma_x^{\mathrm{in}} = 2d/(d+f)$ for a nonzero source. Normalization gives $\mathbb{E}\eta^2 = 1$. The independent-gate approximation retains this marginal gate variation, and the real diagonal G preserves positive semidefiniteness of $GX\Sigma_x^{\mathrm{in}} X^\top G$.

The expected write capacity is

$$\begin{aligned} &\psi^{\mathrm{write}}(\Sigma_x^{\mathrm{in}}, \Sigma_z^{\mathrm{write}}; k, v_{\mathrm{in}}, v_{\mathrm{out}}, \mu) \\ &\quad := \mathbb{E}_{W,X,G}\psi(W; GX\Sigma_x^{\mathrm{in}} X^\top G, \Sigma_z^{\mathrm{write}}) \\ &\quad = \mathbb{E}_{W,X,G} \ln\det\Big(I_d + (\Sigma_z^{\mathrm{write}})^{-1/2} WGX\Sigma_x^{\mathrm{in}} X^\top G W^\top (\Sigma_z^{\mathrm{write}})^{-1/2}\Big). \end{aligned} \tag{C.5}$$

Here μ is the law of a squared gate entry, and the input and output residual widths are both d, while the intermediate width is k. Its deterministic evaluation is $\widehat{\psi}^{\mathrm{write}}$. For a zero source covariance both versions are zero. The Gaussian-channel identity applies conditional on X, G, W, preserving the propagated spectrum inside the log-determinant before averaging.

C.3 The complete block sum

At the attention branch of block ℓ, write $b = 2\ell - 2$, $d = d_\ell$, and $f = f_\ell$. The next ledger is $p_{b+1} = p_b + \mathbf{1}_{[1,d]}$, where $\mathbf{1}_{[1,d]}$ is an M-dimensional indicator of the active prefix. The covariance and metrics for the FFN are recomputed from p_{b+1} using the same read width d. Let $v_A = 2/(d + d/H)$ and $v_F = 2/(d+f)$. Then the block shorthand in Equation (5) expands as

$$\begin{aligned} S_L^{\mathrm{block}}(p_b; d, f) = {} & 3\widehat{\psi}_{v_A}^{\mathrm{read}}(\Sigma_{x,b}^{\mathrm{read}}; d) \\ & + \widehat{\psi}^{\mathrm{write}}(\Sigma_{x,b}^{\mathrm{in}}, \Sigma_{z,b}^{\mathrm{write}}; d, 1/d, 1/d, \delta_1) \\ & + 2\widehat{\psi}_{v_F}^{\mathrm{read}}(\Sigma_{x,b+1}^{\mathrm{read}}; f) \\ & + \widehat{\psi}^{\mathrm{write}}(\Sigma_{x,b+1}^{\mathrm{in}}, \Sigma_{z,b+1}^{\mathrm{write}}; f, 1/d, v_F, \mu_{d,f}). \end{aligned} \tag{C.6}$$

The coefficients count the three attention reads and two FFN reads; the remaining terms score O and down. After the FFN write, $p_{b+2} = p_b + 2\mathbf{1}_{[1,d]}$.

The dependence on L enters through Z_L; E, H, the gate choice, and variance conventions are fixed by the specification.

D Deterministic evaluation of local capacity

The expectations in Equations (C.1) and (C.5) are evaluated using deterministic equivalents for Gaussian covariance and product ensembles (Hoydis et al., 2011). These approximate the expected log-determinants at finite dimensions. Their deterministic values, with the declared gate quadrature, define the numerical objective optimized by ResNSC-DP.

The large-system interpretation scales the participating dimensions proportionally while retaining their aspect ratios and using normalized variances and controlled covariance spectra. The scalar systems below take the reference-metric eigenvalues as inputs.

D.1 Single-projection evaluation

Let $\tau_1, \dots, \tau_d$ be the eigenvalues of $\Sigma_x^{\mathrm{read}} \succeq 0$ and let $W \in \mathbb{R}^{m\times d}$ have entry variance $v > 0$. Solve

$$1 - \xi = \sum_{i=1}^{d} \frac{v\xi\tau_i}{1 + mv\xi\tau_i}, \qquad 0 < \xi \le 1. \tag{D.1}$$

The left side strictly decreases from one to zero, while the right side is continuous and nondecreasing. There is therefore a unique root; if all τ_i vanish, it is $\xi = 1$. The deterministic read evaluation is

$$\widehat{\psi}_v^{\mathrm{read}}(\Sigma_x^{\mathrm{read}}; m) = \sum_{i=1}^{d} \ln(1 + mv\xi\tau_i) - m\ln\xi - m(1-\xi). \tag{D.2}$$

For $\Sigma_x^{\mathrm{read}} = \tau I_d$, this reduces to the Marchenko–Pastur Shannon transform used by NSC with effective entry variance $v\tau$ (Marčenko & Pastur, 1967; Zhu et al., 2026). For nonuniform histories, the entire eigenvalue list contributes. We solve Equation (D.1) by bisection; a zero metric gives exactly zero capacity.

D.2 Propagated write evaluation

Let $\chi_1, \dots, \chi_d$ be the eigenvalues of $\Sigma_x^{\mathrm{in}} \succeq 0$ and $\rho_1, \dots, \rho_d$ the precision eigenvalues of $(\Sigma_z^{\mathrm{write}})^{-1} \succ 0$. Let μ be the squared-gate law. For a nonzero source, the coupled fixed-point system is

$$\begin{aligned} \gamma &= \int \frac{u}{1 + \alpha\beta u}\, d\mu(u), \\ \alpha &= v_{\mathrm{in}} \sum_{i=1}^{d} \frac{\chi_i}{1 + kv_{\mathrm{in}}\beta\gamma\chi_i}, \qquad \beta = v_{\mathrm{out}} \sum_{j=1}^{d} \frac{\rho_j}{1 + kv_{\mathrm{out}}\alpha\gamma\rho_j}. \end{aligned} \tag{D.3}$$

The associated deterministic write evaluation is

$$\begin{aligned} &\widehat{\psi}^{\mathrm{write}}(\Sigma_x^{\mathrm{in}}, \Sigma_z^{\mathrm{write}}; k, v_{\mathrm{in}}, v_{\mathrm{out}}, \mu) \\ &\quad = \sum_{i=1}^{d} \ln(1 + kv_{\mathrm{in}}\beta\gamma\chi_i) + \sum_{j=1}^{d} \ln(1 + kv_{\mathrm{out}}\alpha\gamma\rho_j) \\ &\qquad + k\int \ln(1 + \alpha\beta u)\, d\mu(u) - 2k\alpha\beta\gamma. \end{aligned} \tag{D.4}$$

These equations specialize the single-user double-scattering system in Hoydis et al. (2011, Theorems 4–5). In that paper's notation, set $N = n_1 = d$, $N_1 = k$, and the signal-to-noise ratio to one; take $R_1 = kv_{\mathrm{out}}(\Sigma_z^{\mathrm{write}})^{-1}$, $T_1 = dv_{\mathrm{in}}\Sigma_x^{\mathrm{in}}$, $Q_1 = I_d$, and S_1 as the diagonal squared-gate matrix. Substituting $\bar{g}_1 = \alpha$, $\delta_1 = \beta$, and $g_1 = (k/d)\beta\gamma$, and expressing the normalized gate trace

as an integral against μ, gives Equation (D.3). Multiplying the normalized log-determinant in their Theorem 5 by d gives Equation (D.4), including the term $-2k\alpha\beta\gamma$.

We use the positive solution specified by this system. The cited complex-Gaussian result supplies the first-order spectral approximation used here for the real Gaussian reference ensembles. The numerical gate law below has finite support, consistent with the bounded-spectrum large-system setting. A zero source covariance gives zero capacity directly.

For attention, $\mu = \delta_1$. For the gated FFN, we use 32-point Gauss–Hermite quadrature for the standard-normal variable in Equation (C.4). Given nodes z_s and normalized weights ω_s, define

$$u_s = \frac{\mathrm{SiLU}(\sqrt{q}\, z_s)^2}{\sum_{v=1}^{32} \omega_v\, \mathrm{SiLU}(\sqrt{q}\, z_v)^2}, \qquad \mu_{d,f}^{(32)} = \sum_{s=1}^{32} \omega_s \delta_{u_s}. \tag{D.5}$$

Then $\sum_s \omega_s u_s = 1$. All candidates use this discrete law in the numerical block score.

The coupled equations are evaluated by damped fixed-point iteration, initialized at $\alpha = v_{\mathrm{in}} \operatorname{tr} \Sigma_x^{\mathrm{in}}$ and $\beta = v_{\mathrm{out}} \operatorname{tr}[(\Sigma_z^{\mathrm{write}})^{-1}]$. Repeated eigenvalues are grouped with their multiplicities in every sum in Equations (D.1)–(D.4).

Numerical settings and finite-matrix validation. Single-projection evaluation uses 48 bisection steps. Product evaluation uses damping $1/2$, stopping tolerance 2×10^{-12} on the relative fixed-point update, and a maximum of 1,000 iterations. All computations use float64 and the 32-point gate rule above.

Table 8 compares these evaluations with explicit independent Gaussian reference matrices at $d \in \{64, 128\}$. The isotropic history is $p_i = 5$; the nonuniform history assigns 5 and 1 to equal coordinate groups; the zero-source case assigns 5 and 0. We use $L = 8$, construct Σ_x^{in} by Equation (B.4) and $\Sigma_x^{\mathrm{read}}, \Sigma_z^{\mathrm{write}}$ by Equation (3), and set $k = d$ for these three cases. The gated case uses the nonuniform history and $k = 2d$, drawing independent gates from the normalized 32-point law in Equation (D.5). Each case uses 1,200 independent draws, upstream variance $1/d$, write variance $2/(d + k)$, and read variance $2/(d + k)$. The maximum relative differences round to 0.337% for reads and 0.686% for writes in these cases.

Table 8: Finite-matrix validation: absolute relative difference between the deterministic value and the mean of 1,200 independent reference draws (%).

Context	d	Read difference (%)	Write difference (%)
Isotropic	64	0.176	0.444
Nonuniform history	64	0.222	0.397
Zero source directions	64	0.337	0.647
SiLU-gated	64	0.165	0.562
Isotropic	128	0.135	0.203
Nonuniform history	128	0.135	0.260
Zero source directions	128	0.132	0.337
SiLU-gated	128	0.047	0.686

E HISTORY-AWARE SEARCH, BOUNDS, AND SPECIALIZATION

This appendix supplies the cost convention, state argument, resource-price bounds, and complete search procedure. All results concern the finite product menus in Equation (6), with two equal-window writes per block.

E.1 SEARCH DOMAIN AND PARAMETER COSTS

A specification g fixes L_g, E_g, H_g, a residual-width menu, admissible (d, f) pairs in $\mathcal{A}_g$, and reference conventions. Each block chooses a pair from that menu. The depth and chosen pair determine the

block cost $c_{g,\ell}(d, f)$; the specification determines the fixed cost $c_{\text{fixed}}(g)$. Define the variable budget

$$\mathcal{B}_g = B_g - c_{\text{fixed}}(g). \tag{E.1}$$

For bias-free standard attention and gated FFNs, two width-d block norms, and a final norm of width equal to the last active width, the parameter cost is

$$c_{g,\ell}(d, f) = 4d^2 + 3df + 2d + \mathbf{1}\{\ell = L_g\}d. \tag{E.2}$$

The terminal term counts the final width-d_L RMSNorm, and $c_{\text{fixed}} = 2VE$ counts untied embedding and readout matrices. Appendix G gives the instantiated parameter counts.

For prefixes within g, $\mathbf{n}$ counts the residual-width choices, $t = \sum_r n_r$, and $p_g(\mathbf{n})$ is Equation (7) with that specification's embedding support and width menu. A local choice (d, f) has score $S^{\text{block}}_{L_g}(p_g(\mathbf{n}); d, f)$ with all other reference arguments fixed by g.

E.2 Order dependence and sufficient history

For two choices (d_1, f_1) and (d_2, f_2) evaluated after a ledger p, the score difference on exchanging them is

$$\begin{aligned} &S^{\text{block}}_L(p; d_1, f_1) + S^{\text{block}}_L(p + 2\mathbf{1}_{[1,d_1]}; d_2, f_2) \\ &\quad - S^{\text{block}}_L(p; d_2, f_2) - S^{\text{block}}_L(p + 2\mathbf{1}_{[1,d_2]}; d_1, f_1). \end{aligned} \tag{E.3}$$

Both orders end at $p + 2\mathbf{1}_{[1,d_1]} + 2\mathbf{1}_{[1,d_2]}$. Their intermediate contexts differ, allowing different accumulated scores.

Proposition 1 (Residual-history sufficiency and dominance)**.** *Fix g and the two-write reference. Prefixes with the same count vector $\mathbf{n}$ have identical additional scores for every common suffix. If their remaining budgets agree, their feasible continuation sets agree. If their spent costs and accumulated scores satisfy $c_1 \le c_2$ and $s_1 \ge s_2$, respectively, then the first prefix dominates the second.*

Proof. Equation (7) reconstructs the ledger from $\mathbf{n}$, and $t = \sum_r n_r$ fixes the next block index. For a common next choice (d, f), both prefixes therefore use identical reference contexts, contribute the same value $S^{\text{block}}_{L_g}(p_g(\mathbf{n}); d, f)$, and enter the same next count state. Induction over the suffix length gives equality of all additional scores. Feasible continuations depend only on the fixed specification, block index, choices, and remaining additive budget. Any suffix affordable after cost c_2 is affordable after $c_1 \le c_2$. Adding its common score to $s_1 \ge s_2$ proves dominance. □

The result applies to the stated product menus and additive budget. Residual widths determine the unit-write increments; c and s retain the costs and earned scores of all choices, including FFN widths.

E.3 Resource-price suffix bounds

A prefix label is $\pi = (g, \mathbf{n}, c, s)$. Let $r(d)$ be the index of width d in the menu, and let $\mathbf{e}_r$ be the corresponding unit count vector. For a nonnegative resource price λ, define

$$\begin{aligned} V_{g,\lambda}(\mathbf{n}) = \max_{(d,f)\in\mathcal{A}_g} \big\{ &S^{\text{block}}_{L_g}(p_g(\mathbf{n}); d, f) - \lambda c_{g,t+1}(d, f) \\ &+ V_{g,\lambda}(\mathbf{n} + \mathbf{e}_{r(d)}) \big\}, \qquad V_{g,\lambda}(\mathbf{n}) = 0 \text{ if } t = L_g. \end{aligned} \tag{E.4}$$

This backward recurrence retains history transitions but relaxes the hard remaining budget into a linear cost penalty. For a nonempty finite price set $\Lambda_g \subset [0, \infty)$, define

$$U(\pi) = s + \min_{\lambda\in\Lambda_g} \left\{ V_{g,\lambda}(\mathbf{n}) + \lambda(\mathcal{B}_g - c) \right\}. \tag{E.5}$$

Proposition 2 (Admissible completion bound)**.** *When the suffix potentials in Equation* (E.4) *are evaluated exactly, $U(\pi)$ bounds the score of every budget-feasible complete architecture extending π.*

Proof. Take any feasible suffix with cost $c_{\text{suf}} \le \mathcal{B}_g - c$ and additional score s_{suf}. The relaxed recurrence maximizes over all such suffix choices and possibly budget-infeasible ones. Hence, for each $\lambda \ge 0$,

$$s_{\text{suf}} - \lambda c_{\text{suf}} \le V_{g,\lambda}(\mathbf{n}), \qquad s + s_{\text{suf}} \le s + V_{g,\lambda}(\mathbf{n}) + \lambda(\mathcal{B}_g - c). \tag{E.6}$$

Algorithm 2 ResNSC-DP: detailed search procedure. Here $r(d)$ indexes residual width d in its menu, and $\mathbf{e}_r$ is the corresponding unit count vector.

Require: Specifications $\mathcal{G}$, block menus $\mathcal{A}_g$, budgets B_g, tolerance $\epsilon \geq 0$
Ensure: Architecture $(\widehat{g}, \widehat{a})$ and bounds (LB, UB), or infeasibility

$(\mathcal{Q}, \widehat{g}, \widehat{a}, \mathrm{LB}) \leftarrow$ INITIALIZE$(\mathcal{G}, \{\mathcal{A}_g, B_g\})$
if $\mathrm{LB} = -\infty$ **then return** infeasible
while $\mathcal{Q} \neq \varnothing$ **do**
  $\mathcal{Q} \leftarrow$ PRUNE$(\mathcal{Q})$; $\mathrm{UB} \leftarrow \max\{\mathrm{LB}, \max_{\pi \in \mathcal{Q}} U(\pi)\}$
  if $\mathrm{UB} - \mathrm{LB} \leq \epsilon$ or LIMITREACHED **then break**
  $\pi = (g, \mathbf{n}, c, s) \leftarrow$ POPMAX$(\mathcal{Q})$; $t \leftarrow \sum_r n_r$
  for $(d, f) \in \mathcal{A}_g$ **do**
    $\mathbf{n}' \leftarrow \mathbf{n} + \mathbf{e}_{r(d)}, \quad c' \leftarrow c + c_{g,t+1}(d, f)$
    $s' \leftarrow s + S^{\mathrm{block}}_{L_g}(p_g(\mathbf{n}); d, f), \quad \pi' \leftarrow (g, \mathbf{n}', c', s')$
    if $\neg$FEASIBLE(π') or DOMINATED(π') or $U(\pi') \leq \mathrm{LB}$ **then continue**
    if $t + 1 = L_g$ **then** $(\widehat{g}, \widehat{a}, \mathrm{LB}) \leftarrow$ UPDATEBEST(π')
    else INSERT$(\mathcal{Q}, \pi')$
  end for
end while
$\mathcal{Q} \leftarrow$ PRUNE$(\mathcal{Q})$; $\mathrm{UB} \leftarrow \max\{\mathrm{LB}, \max_{\pi \in \mathcal{Q}} U(\pi)\}$
return $(\widehat{g}, \widehat{a}), \mathrm{LB}, \mathrm{UB}$ ▷ $\max \varnothing = -\infty$

Each price gives an upper bound on the same feasible completion; taking their minimum proves the claim. A conservatively overestimated potential also preserves the inequality. □

The state graph is acyclic because every transition increases t by one. Potentials can therefore be computed backward. States known to lie on no budget-feasible root-to-terminal path may be removed, provided the remaining graph still contains every feasible completion relevant to the bound.

E.4 RESNSC-DP AND THE GLOBAL OBJECTIVE GAP

ResNSC-DP maintains a maximum-priority queue $\mathcal{Q}$ of prefixes, nondominated frontiers indexed by $(g, \mathbf{n})$, and an incumbent complete architecture $(\widehat{g}, \widehat{a})$. Its score is $\mathrm{LB} = S_{\widehat{g}}(\widehat{a})$. The initialization and search operations of Algorithm 2 are specified below.

Initialization. INITIALIZE prepares spectral values and resource-price potentials. Minimum-cost paths supply an incumbent for every feasible specification; price-solution paths and feasible one-block replacements improve it before feasible roots enter the queue.

Shared search operations. FEASIBLE compares spent cost plus minimum completion cost with $\mathcal{B}_g$. DOMINATED and INSERT maintain the frontier at $(g, \mathbf{n})$ according to Proposition 1, marking displaced labels stale. PRUNE removes stale or incumbent-bounded entries, and UPDATEBEST retains the best feasible complete path. Time and expansion limits are checked between complete prefix expansions.

Theorem 1 (Finite-menu objective certificate)**.** *Suppose Equation* (6) *has a nonempty feasible set and the suffix bounds are admissible. With exact arithmetic for the fixed objective, Algorithm* 2 *maintains the following bounds at initialization and after each complete prefix expansion:*

$$\mathrm{LB} = S_{\widehat{g}}(\widehat{a}) \leq S^* \leq \max\left\{\mathrm{LB}, \max_{\pi \in \mathcal{Q}\ valid} U(\pi)\right\} = \mathrm{UB}. \tag{E.7}$$

It returns an ϵ-optimal architecture if $\mathrm{UB} - \mathrm{LB} \leq \epsilon$*, and an optimal architecture if the valid queue is exhausted. At an earlier stop, the same bounds quantify the remaining objective gap.*

Proof. The feasible incumbent supplies a lower bound. Initially every feasible architecture has a prefix at one of the feasible roots. On expanding a valid prefix, all possible next choices are considered. Budget pruning removes only paths that cannot have a feasible completion. By Proposition 1, a dominated prefix can be replaced by one with no greater cost, at least as much earned score, and identical continuation values. By Proposition 2, a prefix with upper bound at most the incumbent cannot improve that incumbent.

After each complete expansion, every potentially improving feasible completion is therefore represented by a valid live prefix, or by a dominating prefix with at least as good a completion; completed paths have already updated the incumbent. The maximum live bound together with the incumbent bounds the optimum. The finite search tree is exhausted if no valid improving prefix remains, at which point $\mathrm{UB} = \mathrm{LB}$. Otherwise the gap bounds the incumbent's suboptimality, proving the stopping claim. □

Numerical objective and reporting. We evaluate the spectral objective in float64, with a small upward margin on computed completion bounds and absolute stopping tolerance 10^{-6}. We report the absolute gap $\mathrm{UB} - \mathrm{LB}$ and relative gap $(\mathrm{UB} - \mathrm{LB})/\max(1, |\mathrm{LB}|)$; the search terminates when the valid queue is exhausted or the numerical gap reaches this tolerance. Theorem 1 assumes exact arithmetic and admissible bounds.

E.5 Fixed-residual-width dynamic programming

When $d_\ell = d$ for every layer, the ledger before each block is determined by depth independently of the FFN choices. In the full-width case $d = E$, $p_{2\ell-2,i} = 2\ell - 1$ on the active window, and the two scalar precisions follow Appendix B.3. The block value remains $S_L^{\mathrm{block}}(p_{2\ell-2}; d, f)$.

Let $D_\ell(B')$ be the highest score of the first ℓ blocks spending at most the integer variable budget B'. Fix one specification g and write $L = L_g$. Then

$$D_\ell(B') = \max_{\substack{f:(d,f)\in\mathcal{A}_g \\ c_{g,\ell}(d,f)\le B'}} \left\{ D_{\ell-1}(B' - c_{g,\ell}(d,f)) + S_L^{\mathrm{block}}(p_{2\ell-2}; d, f) \right\}, \qquad D_0(B') = 0 \quad (B' \ge 0), \qquad D_\ell(B') = -\infty \quad (B' < 0). \tag{E.8}$$

The maximum over an empty feasible set is $-\infty$. This multiple-choice knapsack recurrence is exact for the evaluated score and stated costs. The ledger remains depth-determined, preserving residual-context weighting and propagated write sources while making each layer's value independent of earlier FFN choices.

E.6 Joint-width search results

Joint-width search. On the complete 121M and 374M menus specified in Appendix G, ResNSC-DP returns the reported allocations with zero final numerical objective gap (Table 9). Both searches exhaust the valid queue.

Table 9: Final objectives and numerical gaps for joint residual/FFN-width search. Both searches exhaust the valid queue.

Instance	Objective $S(\widehat{a})$	Rel. gap
121M joint	59,330.78	0
374M joint	133,930.49	0

F Training and evaluation statistics

Data and training budget. Pretraining uses FineWeb-Edu (Penedo et al., 2024), a 32,000-token Llama-2 tokenizer, and packed sequences of length 4,096. The fixed corpus subsets contain approximately 8.34B available tokens at 121M and 37.45B at 374M. The smaller subset uses the first ten of the larger subset's 45 source shards. Each subset is split by document index into 99.5% training and 0.5% validation. All architectures at a given scale use the same subset, split, and token budget.

The global batch contains 128 sequences, or 524,288 tokens. We set the main-phase budget to approximately 20 tokens per parameter using the uniform model's instantiated parameter count, following the budget convention motivated by compute-optimal training (Hoffmann et al., 2022). The main phase contains 4,616 updates at 121M and 14,267 at 374M, corresponding to approximately 2.42B and 7.48B tokens. A final linear-decay phase adds 462 and 1,427 updates. Total training therefore uses 2.662B and 8.228B tokens, or approximately 22 tokens per parameter.

Optimization. All models use AdamW (Loshchilov & Hutter, 2019) with base learning rate 0.01, $(\beta_1, \beta_2) = (0.9, 0.95)$, weight decay 0.1, and numerical stabilizer 10^{-10}. Initialization and parameter-group scaling follow the shared μP recipe (Yang et al., 2021): initialization scale 0.1, width multiplier 8, embedding multiplier 12, and residual multiplier 0.22. Attention logits are scaled by the inverse head dimension. These are the training conventions; the reference variance rules defining the spectral score are specified separately in Section 3.3.

The schedule uses linear warmup for the first 8% of the main phase, followed by a power-law learning-rate schedule with coefficient 512 and exponent -0.566 as a function of consumed tokens. The final decay linearly reduces the ending main-phase learning rate to zero over approximately 10% additional updates. Training uses eight A100-40GB GPUs, BF16, FSDP2[1], and FlashAttention-2 (Dao, 2024).

Training seeds and aggregation. At 121M, every arm in Table 2 has three independent training runs (seeds 42, 43, 44), each starting from a fresh initialization and using its seed for data order; means and sample standard deviations are taken over these runs. At 374M and for the scoring ablations, each arm has one full run with seed 42 and two continuations of its final-decay phase (1,427 and 462 updates, respectively) that restore the seed-42 main checkpoint and regenerate the data order with seeds 43 and 44; the reported standard deviations measure annealing variation. Perplexity is exponentiated from each run's loss before averaging; Avg8 is computed within each run before aggregation. Table 10 lists the losses and Table 11 the 374M per-task accuracies.

Table 10: Held-out FineWeb-Edu loss (nats) at 121M and 374M. 121M arms: three independent runs (seeds 42–44). 374M arms: one run (seed 42) plus two final-decay continuations, whose standard deviation reflects annealing variation only.

Method	121M loss	374M loss
ResNSC	2.96496 (±0.00067)	2.61025 (±0.00017)
uniform	3.00813 (±0.00109)	2.65818 (±0.00011)
VWT	2.96922 (±0.00286)	2.61367 (±0.00016)
TLM	3.01478 (±0.00108)	2.66964 (±0.00006)
NSC	2.99598 (±0.00134)	2.62640 (±0.00019)
Reverse	3.00905 (±0.00233)	2.67282 (±0.00005)

Table 11: 374M zero-shot accuracies (%) after the final decay: eight tasks and their mean (Avg8), one run plus two final-decay continuations (mean and sample standard deviation). HellaSwag and ARC-c use normalized accuracy. Bold: best per column; underline: second best.

Method	LMB ↑	PIQA ↑	HSwag ↑	WGrnd ↑	ARC-e ↑	ARC-c ↑	SIQA ↑	BoolQ ↑	Avg8 ↑
ResNSC (ours)	**29.26** ± 0.16	64.89 ± 0.33	35.49 ± 0.20	50.41 ± 0.59	**54.53** ± 0.42	25.06 ± 0.26	37.17 ± 0.16	59.97 ± 0.96	**44.60** ± 0.07
uniform	26.69 ± 0.26	64.56 ± 0.22	34.16 ± 0.08	**52.57** ± 0.21	52.46 ± 0.40	26.11 ± 0.34	**37.22** ± 0.18	58.76 ± 0.36	44.07 ± 0.19
VWT	27.03 ± 0.32	**65.38** ± 0.35	**35.94** ± 0.30	50.51 ± 0.96	53.68 ± 0.76	**26.39** ± 0.47	36.51 ± 0.34	58.66 ± 0.55	44.26 ± 0.11
TLM	26.24 ± 0.16	63.13 ± 0.36	33.87 ± 0.06	50.75 ± 0.36	52.57 ± 0.51	25.63 ± 0.34	36.18 ± 0.22	**61.46** ± 0.23	43.73 ± 0.05
NSC	27.74 ± 0.36	64.42 ± 0.14	35.27 ± 0.14	50.33 ± 0.79	53.04 ± 0.21	25.74 ± 0.27	36.51 ± 0.38	61.07 ± 0.27	44.26 ± 0.09
Reverse	25.76 ± 0.21	64.16 ± 0.45	33.62 ± 0.06	52.07 ± 0.84	53.31 ± 0.48	25.26 ± 0.09	35.55 ± 0.06	53.12 ± 1.72	42.86 ± 0.18

Evaluation protocol. FineWeb-Edu cross-entropy is measured on the same 2,048 held-out sequences across architectures and runs, each contributing 4,096 next-token targets. We use `lm-eval` 0.4.13 (Gao et al., 2023) with BF16 models, the shared Llama-2 tokenizer, a maximum input length of 4,096, and zero-shot task templates. Each task uses its complete evaluation split (Table 12).

At both scales, we report WikiText-2 word perplexity and LAMBADA final-word perplexity. WikiText-2 uses rolling likelihood with word-count normalization; LAMBADA scores the final

[1] https://docs.pytorch.org/docs/2.8/distributed.fsdp.fully_shard.html

word conditioned on its preceding text. At 374M, the eight accuracy tasks are LAMBADA, PIQA, HellaSwag, WinoGrande, ARC-e, ARC-c, SocialIQA, and BoolQ. HellaSwag and ARC-c use normalized accuracy; the other tasks use accuracy. Avg8 is their arithmetic mean within each run.

Table 12: Pretraining-model evaluation datasets, splits, and metrics. All tasks use zero-shot templates and the complete indicated split.

Task ID	Dataset	Split	Reported metric
`wikitext`	WikiText-2 (Merity et al., 2017)	test	word PPL
`lambada_openai`	LAMBADA (Paperno et al., 2016)	test	PPL, accuracy
`piqa`	PIQA (Bisk et al., 2020)	validation	accuracy
`hellaswag`	HellaSwag (Zellers et al., 2019)	validation	normalized accuracy
`winogrande`	WinoGrande (XL) (Sakaguchi et al., 2020)	validation	accuracy
`arc_easy`	ARC-Easy (Clark et al., 2018)	test	accuracy
`arc_challenge`	ARC-Challenge (Clark et al., 2018)	test	normalized accuracy
`social_iqa`	SocialIQA (Sap et al., 2019)	validation	accuracy
`boolq`	BoolQ (Clark et al., 2019)	validation	accuracy

F.1 Scoring ablation protocol

Each scoring variant in Table 3 selects an architecture from the same 121M width menus under the same parameter ceiling, then uses the training and evaluation protocols above. We test three simplifications of residual history. *Without residual context* uses isotropic normalized source powers and identity read and receiver metrics, while retaining within-branch propagation. *Depth only* sets every active coordinate's reference power to $p_{b,i} = b+1$, where b indexes residual branches, replacing width history with branch position. *Mean history only* uses the average reference power over the current active coordinates when evaluating each branch, retaining mean power while removing coordinate-wise differences.

To test propagated features, *without propagated spectrum* replaces the write-source covariance $\Sigma^{\mathrm{write}}_{x,b}$ with the appropriately sized identity for both attention output and FFN down projections, preserving expected reference power while removing the propagated spectrum. Read scoring and the residual-dependent receiver covariance remain unchanged.

G Layer-wise architecture configurations

Architecture and allocation rules. Each Transformer block (Vaswani et al., 2017) contains pre-RMSNorm (Zhang & Sennrich, 2019) causal self-attention followed by a pre-RMSNorm SwiGLU FFN (Shazeer, 2020). Attention uses rotary position embeddings (Su et al., 2021) and 16 query and 16 key/value heads, with head dimension $d_\ell/16$. The FFN has gate and up projections from d_ℓ to f_ℓ and a down projection back to d_ℓ. Linear projections are bias-free, and input embeddings and vocabulary readout are untied. Inactive residual coordinates retain their values along the identity path without additional parameters. The embedding/readout width is fixed at $E = 640$ for 121M and $E = 1024$ for 374M. Final RMSNorm acts on the last active width d_L, before the state is restored or cropped to readout width E.

We compare six allocation rules. *Uniform* uses $d_\ell = E$ and $f_\ell = 2.75E$. *VWT* uses the default bottleneck-location and width-ratio settings $(0.75, 0.3)$, with $f_\ell = 4d_\ell$. *TLM* holds residual width fixed and applies a cosine FFN taper from 1.5 to 0.5 times the uniform FFN width (Bayat et al., 2026). *ResNSC* jointly selects active residual and FFN widths. *NSC* maximizes the original NSC score over the same residual-width and FFN-width menu under the same parameter ceiling. *Reverse* reverses the ResNSC sequence of (d_ℓ, f_ℓ) pairs, preserving the projection-shape inventory used by the score.

Search menus. At each scale, the active-width menu is $d_\ell/E \in \{0.25, 0.50, \ldots, 3.00\}$. The FFN-ratio menu is $\{1.375, 2.0625, 2.75, 3.4375, 4.125, 4.8125\}$ at 121M and $\{2.0625, 2.75, 3.4375, 4.125, 4.8125\}$ at 374M. FFN widths are rounded to multiples of 32. We search each scale independently under the uniform model's parameter ceiling, counting

embeddings, readout, and normalization as well as projections. Each layer chooses its own active width and FFN ratio; the feasible set imposes no monotonicity constraint or prescribed taper.

Tables 14 and 15 give each model's exact (d_ℓ, f_ℓ) sequence, ordered from input to output. These are active computation widths; the persistent residual state follows Equation (B.1).

Parameter counts. Table 13 reports total parameters, including embeddings, readout, and normalization, using $2VE + \sum_\ell (4d_\ell^2 + 3d_\ell f_\ell + 2d_\ell) + d_L$ with $V = 32{,}000$. The last-active-width RMSNorm accounts for the ResNSC/Reverse differences of 640 parameters at 121M and 1,280 at 374M. The 121M VWT configuration exceeds the Uniform search ceiling by 12,960 parameters.

Table 13: Instantiated parameter counts for the pretraining configurations, including embeddings, readout, and normalization. Column headings are the scale labels used in the paper.

Method	121M	374M
ResNSC	121,188,640	373,799,936
Uniform	121,262,720	373,867,520
VWT	121,275,680	373,859,200
TLM	121,262,720	373,867,520
NSC	120,898,560	373,847,040
Reverse	121,189,280	373,801,216

Table 14: Exact 121M per-layer widths in input-to-output order: residual width d_ℓ and FFN intermediate width f_ℓ.

Layer	ResNSC		uniform		VWT		TLM		NSC		Reverse	
	d	f	d	f	d	f	d	f	d	f	d	f
1	1120	1536	640	1760	864	3456	640	2624	640	1312	480	672
2	1280	1760	640	1760	768	3072	640	2624	800	1088	480	672
3	1280	1760	640	1760	704	2816	640	2560	800	1088	480	672
4	960	1312	640	1760	608	2432	640	2464	800	1088	480	672
5	960	1312	640	1760	544	2176	640	2336	800	1088	480	672
6	800	1088	640	1760	512	2048	640	2208	800	1088	480	672
7	640	896	640	1760	448	1792	640	2016	800	1088	480	672
8	640	896	640	1760	416	1664	640	1856	800	1088	640	896
9	640	896	640	1760	352	1408	640	1664	800	1088	640	896
10	480	672	640	1760	320	1280	640	1504	800	1088	640	896
11	480	672	640	1760	288	1152	640	1312	800	1088	800	1088
12	480	672	640	1760	256	1024	640	1184	800	1088	960	1312
13	480	672	640	1760	352	1408	640	1056	800	1088	960	1312
14	480	672	640	1760	480	1920	640	960	800	1088	1280	1760
15	480	672	640	1760	640	2560	640	896	800	1088	1280	1760
16	480	672	640	1760	864	3456	640	896	640	896	1120	1536

Table 15: Exact 374M per-layer widths in input-to-output order: residual width d_ℓ and FFN intermediate width f_ℓ.

Layer	ResNSC		uniform		VWT		TLM		NSC		Reverse	
	d	f	d	f	d	f	d	f	d	f	d	f
1	1792	3712	1024	2816	1408	5632	1024	4224	1280	3520	512	1056
2	2048	4224	1024	2816	1312	5248	1024	4224	1280	2656	512	1056
3	2048	4224	1024	2816	1216	4864	1024	4160	1280	2656	512	1056
4	2048	4224	1024	2816	1152	4608	1024	4096	1280	2656	512	1056
5	1536	3168	1024	2816	1056	4224	1024	4032	1280	2656	512	1056
6	1280	2656	1024	2816	992	3968	1024	3904	1280	2656	512	1056
7	1280	2656	1024	2816	928	3712	1024	3776	1280	2656	512	1056
8	1024	2112	1024	2816	864	3456	1024	3616	1280	2656	512	1056
9	1024	2112	1024	2816	800	3200	1024	3456	1024	2112	768	1600
10	1024	2112	1024	2816	736	2944	1024	3296	1024	2112	768	1600
11	768	1600	1024	2816	704	2816	1024	3104	1024	2112	768	1600
12	768	1600	1024	2816	640	2560	1024	2912	1024	2112	768	1600
13	768	1600	1024	2816	608	2432	1024	2720	1024	2112	768	1600
14	768	1600	1024	2816	544	2176	1024	2528	1024	2112	768	1600
15	768	1600	1024	2816	512	2048	1024	2336	1024	2112	1024	2112
16	768	1600	1024	2816	480	1920	1024	2176	1024	2112	1024	2112
17	512	1056	1024	2816	448	1792	1024	2016	1024	2112	1024	2112
18	512	1056	1024	2816	416	1664	1024	1856	1024	2112	1280	2656
19	512	1056	1024	2816	512	2048	1024	1728	1024	2112	1280	2656
20	512	1056	1024	2816	640	2560	1024	1600	1024	2112	1536	3168
21	512	1056	1024	2816	768	3072	1024	1536	1024	2112	2048	4224
22	512	1056	1024	2816	928	3712	1024	1472	1024	2112	2048	4224
23	512	1056	1024	2816	1152	4608	1024	1408	1024	2112	2048	4224
24	512	1056	1024	2816	1408	5632	1024	1408	1024	2112	1792	3712

H LoNAS-LLaMA protocol and budget results

For the supernet in Section 4.2, each block chooses an FFN width from $\{5504, 6880, 8256, 9632, 11008\}$ and a Q/K/V LoRA rank from $\{28, 32\}$ (Hu et al., 2022); O and FFN adapters retain rank 32. Both methods select rank 32 throughout. Table 16 reports eight matched parameter budgets and the eight benchmark accuracies averaged into Avg8, including OpenBookQA (Mihaylov et al., 2018); Table 17 gives the configurations. Reverse reverses FFN-width assignments across the original pretrained layers. Labels A–G match Figure 3. The archived proxy searches use 50 individuals over 20 generations (1,000 evaluations) under a 5.7B ceiling (Zhu et al., 2026).

LoRA scoring. We add the contextual scores of the three Q/K/V adapter paths to the seven-projection score. Each path has $A \in \mathbb{R}^{r\times d}$ and $B \in \mathbb{R}^{d\times r}$, with entry variance $2/(d+r)$ and unit noise. Their source covariances are $\Sigma^{\mathrm{read}}_{x,b}$ for A and $A^{\mathrm{ref}}\Sigma^{\mathrm{read}}_{x,b}(A^{\mathrm{ref}})^\top$ for B, where independent A^{ref} has entry variance $1/d$. These terms use the spectral evaluations in Appendix D, leaving the residual ledger unchanged. At fixed residual width they are independent of FFN width; rank adds no dense deployment cost after merging. Rank 32 scores higher at every layer on this menu.

Search cost and FLOPs. The 0.28 s search time is the median of five fresh-process runs on one CPU thread, covering spectral scoring and joint FFN/rank selection at all eight budgets. NSC and proxy timings follow Zhu et al. (2026). Figure 3 counts dense forward FLOPs at sequence length 256, with two FLOPs per multiply-add, including projections, attention products, FFNs, and vocabulary readout. Matched ResNSC/NSC configurations and the highest-budget Reverse control have identical FLOPs.

Table 16: Complete LoNAS results with the shared FFN menu $\{5504, 6880, 8256, 9632, 11008\}$. ResNSC and NSC have identical parameter counts at each budget. Labels match Figure 3; "best" denotes the highest-budget point. Δ is the Avg_8 difference from the corresponding NSC model (percentage points).

Method	Params	BoolQ	PIQA	SIQA	HSwag	WGrnd	ARC-e	ARC-c	OBQA	Avg_8	Δ
NSC best	5.690 B	64.01	72.85	68.94	51.30	64.72	72.43	58.70	70.80	65.47	—
NSC A	5.572 B	64.01	72.42	68.17	51.26	64.09	72.52	58.53	70.60	65.20	—
NSC B	5.538 B	63.73	72.42	68.27	51.05	63.46	72.18	58.79	71.20	65.14	—
NSC C	5.487 B	63.61	72.25	67.50	50.69	63.38	72.18	58.87	72.00	65.06	—
NSC D	5.453 B	63.73	72.03	66.73	49.95	63.85	71.34	58.19	70.80	64.58	—
NSC E	5.420 B	63.15	71.49	67.71	49.73	63.69	71.30	57.08	71.00	64.39	—
NSC F	5.369 B	62.51	69.37	66.63	46.54	62.51	68.06	54.86	67.80	62.29	—
NSC G	5.098 B	59.27	69.48	64.64	44.73	62.27	66.04	52.82	66.60	60.73	—
ResNSC best	5.690 B	61.71	76.71	70.98	58.39	67.01	78.45	61.69	76.00	68.87	+3.40
ResNSC A	5.572 B	62.69	76.44	70.83	57.29	66.77	78.03	61.26	76.20	68.69	+3.49
ResNSC B	5.538 B	63.73	76.55	70.83	57.08	66.30	77.99	61.35	75.80	68.70	+3.56
ResNSC C	5.487 B	63.55	76.50	70.78	57.08	65.27	77.69	61.09	74.80	68.34	+3.28
ResNSC D	5.453 B	63.55	76.12	70.37	57.36	65.19	77.82	60.15	74.20	68.09	+3.51
ResNSC E	5.420 B	62.87	76.01	70.78	56.54	65.43	77.23	60.32	73.20	67.80	+3.41
ResNSC F	5.369 B	64.37	76.01	70.37	55.82	65.04	76.98	59.22	74.20	67.75	+5.46
ResNSC G	5.098 B	63.61	73.01	67.76	50.97	63.22	73.19	57.17	72.00	65.12	+4.39
Reverse best	5.690 B	61.25	67.46	63.10	42.64	61.17	65.24	51.45	63.60	59.49	-5.98

Table 17: Exact LoNAS FFN configurations for the budgets in Table 16. Each width column lists the layers using that FFN intermediate width; layers are numbered 1–32 from input to output, and a dash means none. All configurations have residual width 4,096; both search methods select LoRA rank 32 in every layer.

			Layer indices by FFN width				
Method	Budget	Params (B)	11008	9632	8256	6880	5504
ResNSC	best	5.690	1–9	10–13	14–18	19–26	27–32
ResNSC	A	5.572	1–8	9–12	13–16	17–23	24–32
ResNSC	B	5.538	1–8	9–11	12–16	17–22	23–32
ResNSC	C	5.487	1–7	8–11	12–15	16–21	22–32
ResNSC	D	5.453	1–7	8–10	11–14	15–21	22–32
ResNSC	E	5.420	1–6	7–10	11–14	15–20	21–32
ResNSC	F	5.369	1–6	7–9	10–13	14–19	20–32
ResNSC	G	5.098	1–3	4–6	7–9	10–13	14–32
NSC	best	5.690	—	1–2	3–32	—	—
NSC	A	5.572	—	—	1–15, 21–32	16–20	—
NSC	B	5.538	—	—	1–13, 21–32	14–20	—
NSC	C	5.487	—	—	1–11, 21–31	12–20, 32	—
NSC	D	5.453	—	—	1–9, 21–31	10–20, 32	—
NSC	E	5.420	—	—	1, 3–8, 21–31	2, 9–20, 32	—
NSC	F	5.369	—	—	6–9, 21–31	1–5, 10–20, 32	—
NSC	G	5.098	—	—	—	1–31	32
Reverse	best	5.690	24–32	20–23	15–19	7–14	1–6

I ARCHITECTURE-RANKING BENCHMARKS AND PROTOCOL

Evaluation sets and targets. We reuse the existing architecture IDs and performance labels in four benchmark sets: 500 FlexiBERT entries with widths read from each architecture configuration, 200 GPT-2 entries, and 1,001 entries each in AutoFormer-Tiny and AutoFormer-Small. Correlations use GLUE for FlexiBERT, negative validation perplexity for GPT-2 (test perplexity where the validation field is absent), and clean ImageNet accuracy for AutoFormer. All scores are computed analytically without candidate training or forward passes.

Architecture and variance conventions. FlexiBERT reads widths from each architecture configuration and retains the archived inventory: merged QKV for self-attention, two square matrices for the LT operator, one square matrix for DSC, and the archived paired-matrix decomposition for multiple FFNs. This multiple-FFN inventory follows the benchmark scoring representation; the source network implements the FFNs sequentially. GPT-2 uses the archived entry standard deviation 0.02. The remaining inventories retain the NSC shape-based reference variances. AutoFormer uses head dimension 64 and internal attention width $64H$, including a rectangular output projection. Both scores use the same architectural specifications; their Q/K/V evaluation conventions are distinguished below.

Residual and FFN adaptation. Each candidate has constant residual width across its layers, giving normalized branch-input covariance $\Sigma^{\text{in}}_{x,b} = I$ and scalar depth-dependent read/write metrics. The same residual reference is applied across these archived sets, including their post-normalized models. Ordinary FFNs have one up/read projection and one propagated down/write projection, with Gaussian hidden reference and unit gate. The propagation model uses this Gaussian closure in place of a GELU covariance kernel. For GPT-2 and AutoFormer, NSC retains its headwise decomposition, while ResNSC evaluates complete Q/K/V projections with the inherited per-head entry variance and uses a normalized propagated value-to-output channel.

Table 18: Detailed ranking correlations. Parameter-count correlations retain the archived counting convention.

	NSC		ResNSC		Params		
Benchmark	τ_b	ρ	τ_b	ρ	τ_b	ρ	$\Delta\rho$
FlexiBERT	0.6946	0.8843	0.6921	0.8833	0.4846	0.6763	-0.0009
GPT-2	0.8488	0.9681	0.8543	0.9698	0.6386	0.7943	+0.0018
AutoFormer-Tiny	0.6119	0.8106	0.5717	0.7786	0.4691	0.6347	-0.0320
AutoFormer-Small	0.6580	0.8519	0.7192	0.9070	0.7885	0.9414	+0.0551

Ranking within a 10% parameter window. We retain unordered architecture pairs satisfying $|P_i - P_j|/\max(P_i, P_j) < 0.10$, using the same archived parameter counts, performance labels, and frozen NSC/ResNSC scores as above. For each score, let C and D count concordant and discordant pairs after excluding ties in either the score or the performance label. Pairwise accuracy is $C/(C+D)$ and $\tau_{\text{PW}} = (C-D)/(C+D)$. The pair count in Table 19 is measured before tie exclusion; τ_{PW} uses this restricted pair set rather than the global τ_b statistic.

Table 19: Parameter-controlled pairwise ranking (↑). Only pairs with less than 10% relative parameter difference are compared. Accuracy is reported in percent; Δ is ResNSC minus NSC accuracy in percentage points. Bold marks the higher value within each metric and row.

		PW accuracy (%)		τ_{PW}		
Benchmark	Pairs	NSC	ResNSC	NSC	ResNSC	Δ (pp)
FlexiBERT	26,769	**75.15**	74.98	**0.5031**	0.4997	-0.17
GPT-2	1,721	77.28	**78.27**	0.5456	**0.5654**	+0.99
AutoFormer-Tiny	372,469	**75.32**	72.51	**0.5063**	0.4503	-2.80
AutoFormer-Small	130,265	**67.42**	66.16	**0.3483**	0.3232	-1.25

Using only pairs for which the performance label and both scores are non-tied gives the same direction of change on all four sets. Under this common-pair protocol, the ResNSC accuracy differences are -0.09, $+0.99$, -2.72, and -1.25 percentage points in table order.