

CASC: Rethinking Residual Connections in Deep Language Models

Student name: Haoxuan Qi, Jiatong Zhang, Chenyang He

School: Beijing No. 2 Middle School

Province: Beijing

Country/Region: China

Advisor name: Shan Gao

Advisor affiliation: Beijing No. 2 Middle School

Title: CASC: Rethinking Residual Connections in Deep Language Models

CASC: Rethinking Residual Connections in Deep Language Models

Haoxuan Qi^{1*} Jiatong Zhang^{1*} Chenyang He¹

^{*}Co-first authors

¹Beijing No. 2 Middle School

Abstract

We present a systematic comparison of residual connection paradigms for Transformers at 30M parameters. We compare four families: standard additive Pre-LN (GPT, serving as the purely additive baseline), subtractive cascading (CASC) with Post-RMSNorm and with Pre-LN, and additive with learnable scale (Additive-S). The three further families (two subtractive CASC variants and one additive control with learnable scale) are each evaluated across five group-to-layer configurations (16×1 , 8×2 , 4×4 , 2×8 , 1×16), giving 16 models in total together with the GPT baseline. Inside every group the standard additive residual is retained, and one subtractive outer update is applied at each group boundary; writing g for the number of subtractive outer updates, the mixing frequency is $g/16$, with $g = 0$ corresponding to the purely additive GPT baseline. All models are evaluated on the same 1,465 retained validation sequences with paired per-sequence tests. Moderate mixing improves validation performance: Pre- 1×16 at $g/16 = 1/16$ reaches val PPL 20.99 against 21.50 for GPT (+2.44%), and in the Post-RMSNorm family the best directly evaluated configuration is Post- 2×8 ($g/16 = 12.5\%$) at PPL 21.07, with a quadratic fit placing that family’s vertex near 23%. At each of the five nonzero matched configurations the subtractive update beats the identically structured additive one (5/5, mean +2.71%; all paired per-sequence tests $p < 10^{-9}$). All CASC variants incur negligible training overhead: the best configuration Pre- 1×16 runs at 26.5 ms/iter, only 0.4% slower than GPT (26.4 ms/iter). These results show that appropriately mixing subtractive residual updates with standard additive residuals improves validation performance at this 30M scale; all models share a single training seed, so the reported p -values cover evaluation-sampling error.

Keywords: residual connections; subtractive residual; additive–subtractive mixture; matched control; Transformer

Contents

1	Introduction	3
2	Method	3
2.1	Stacked Baseline	3
2.2	CASC: Subtractive Cascading Architecture	3
2.3	Post-Norm Design	4
2.3.1	Normalization design for subtractive residuals	4
2.3.2	Empirical Comparison	4
2.3.3	Feature Space Conditioning	4
3	Experiments	6
3.1	Setup	6
3.2	Evaluation Protocol	6
3.3	Main Results	6
3.4	Mixing Frequency and Performance	7
3.5	Matched Control: Subtractive vs Additive	9

3.6	Stripping Scale Analysis	10
3.7	Group Contribution	10
3.8	Serialized Group Analogy	10
3.9	Mixing Frequency Comparison	10
3.9.1	Jacobian Analysis	11
3.10	Marginal Utility of Successive Layers	11
3.11	Post-RMSNorm vs Pre-LN vs Additive-S Comparison	12
4	Conclusion	12
A	Model Architecture	13
B	Full Evaluation Data	13
	Declaration on the Use of AI Tools	16

1 Introduction

Transformer architectures have led to a series of breakthroughs in natural language processing. The standard residual connection is purely additive:

$$\mathbf{x} \leftarrow \mathbf{x} + \text{Layer}(\text{norm}(\mathbf{x})) \quad (\text{additive, Pre-Norm})$$

This design accumulates information across all layers. However, recent theoretical work has shown that additive residual networks can suffer from *representational collapse* [1]: token representations become increasingly aligned along a few dominant directions, limiting model capacity. Recent work instead redesigns how information is aggregated along depth, including depth-weighted averaging [8], multiway dynamic dense connections [9], learnable multi-stream residuals [10]. The condition number (ratio of largest to smallest singular value of the feature covariance) measures this anisotropy, and its reduction across layers indicates how effectively the architecture counteracts representational collapse [3, 22]. The Pre-LN variant has been specifically identified as causing representation collapse while Post-LN suffers from gradient vanishing [2, 18].

In this paper, we introduce CASC, a framework that periodically inserts *subtractive* residual operations between groups of additive layers. A subtractive cascaded group computes:

$$\mathbf{H} \leftarrow \text{RMSNorm}(\mathbf{H} - s \cdot \text{Block}(\mathbf{H})), \quad s = \sigma(\theta)$$

where $s \in (0, 1)$ controls how much of the Block’s output is stripped away. By placing subtractive groups between groups of standard additive layers, the network can both accumulate and discard information, actively preventing the dominance of any single representational direction. This is analogous to extending the internal multi-head mechanism to an external, serialized form: additive layers add views, subtractive groups redistribute capacity away from over-represented directions.

We compare four residual families at 30M parameters:

- **GPT**: 16 layers additive Pre-LN.
- **Post-RMSNorm**: CASC with Post-Norm subtractive.
- **Pre-LN**: CASC with Pre-Norm subtractive.
- **Additive-S**: Additive with learnable s .

Each family is evaluated with five group-to-layer splits preserving 16 total blocks: 16×1 , 8×2 , 4×4 , 2×8 , and 1×16 . All models: $\text{dim}=384$, $\text{n_head}=6$, $\text{block_size}=512$, LayerNorm, GELU, learned position embeddings.

2 Method

2.1 Stacked Baseline

Standard Transformer stacks N layers with additive residuals:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \text{TransformerBlock}(\mathbf{x}_i)$$

2.2 CASC: Subtractive Cascading Architecture

CASC groups blocks into cascaded groups with subtractive updates:

$$\mathbf{Z}_i = \text{Block}(\mathbf{H}_i)$$

$$\mathbf{H}_{i+1} = \text{RMSNorm}(\mathbf{H}_i - s_i \cdot \mathbf{Z}_i), \quad s_i = \sigma(\theta_i)$$

Here, Block denotes any Transformer variant (e.g., Pre-LN, Post-LN, ReZero, etc.), including standard attention+FFN. CASC is architecture-agnostic: the same subtractive cascading framework can be applied to any existing Transformer design by replacing stacked layers with cascaded groups. A recent line of work instead aggregates preceding-layer or block-level representations by attention [4, 5, 6, 7, 23, 24, 25, 26]; we keep the additive residual inside each group unchanged and apply one fixed-form subtractive update at each group boundary, with the learnable scale $s = \sigma(\theta)$ in the same spirit as ReZero and LayerScale [16, 17].

2.3 Post-Norm Design

2.3.1 Normalization design for subtractive residuals

Standard Transformer uses pre-norm for additive residuals because addition approximately preserves scale: $\|\mathbf{x} + \Delta\| \approx \|\mathbf{x}\|$ when $\|\Delta\| \ll \|\mathbf{x}\|$, so the residual stream norm grows slowly as $O(\sqrt{L})$ and the norm only needs to constrain the sublayer input [18]. Subtraction fundamentally changes this:

$$\|\mathbf{H} - s\mathbf{Z}\|^2 = \|\mathbf{H}\|^2 - 2s\langle\mathbf{H}, \mathbf{Z}\rangle + s^2\|\mathbf{Z}\|^2 \neq \|\mathbf{H}\|^2$$

When $\langle\mathbf{H}, \mathbf{Z}\rangle > 0$, the subtractive residual reduces the norm, creating a systematic scale shrinkage across cascaded groups. This makes post-normalization necessary to restore the representation scale.

Four design choices exist for subtractive residuals:

(A) Normalize both terms:

$$\text{Norm}(\mathbf{H}) - s \cdot \text{Block}(\text{Norm}(\mathbf{H}))$$

Listed for completeness but not experimentally evaluated in this work.

(B) Normalize only Block input (Pre-Norm):

$$\mathbf{H} - s \cdot \text{Block}(\text{norm}(\mathbf{H}))$$

The Block input is normalized to RMS=1 while $\mathbf{H}$ has RMS= R , decoupling $\mathbf{Z}$ from $\mathbf{H}$. To maintain a replacement ratio p , we need $s = pR$, which grows linearly with R and can exceed 1.

(C) No normalization:

$$\mathbf{H} - s \cdot \text{Block}(\mathbf{H})$$

Listed for completeness but not experimentally evaluated in this work.

(D) Normalize after subtraction (Post-Norm):

$$\text{RMSNorm}(\mathbf{H} - s \cdot \text{Block}(\mathbf{H}))$$

Both $\mathbf{H}$ and $\mathbf{Z}$ originate from the same signal path, so their scales are naturally consistent, and the single post-normalization restores RMS=1. This is the design adopted in CASC [20, 30].

2.3.2 Empirical Comparison

Table 2 shows validation PPL at 50K steps for 30M models. Both Post-RMSNorm (D) and Pre-LN (B) are stable across all configurations [19]. Within Post-RMSNorm, longer additive segments between subtractive groups achieve lower PPL; the Pre-LN trend is U-shaped, lowest at the two extremes.

2.3.3 Feature Space Conditioning

To understand why Post-RMSNorm outperforms other schemes, we analyze the feature space condition number $\kappa = \sigma_{\max}/\sigma_{\min}$ of the input representation at each block across all 16 internal blocks. The condition number measures the isotropy of the feature space: low κ indicates features are evenly distributed across representational dimensions, while high κ indicates dominance by a few directions (representational collapse) [21, 31].

Table 1 shows the condition number evolution for all four residual families with the 2×8 configuration. All models start with similar $\kappa \approx 100$ – 115 at block 0 (the token embedding layer): GPT decreases to 80 (-27%); Post-RMSNorm decreases to 33 (-71%); Pre-LN increases to 134 ($+38\%$); Additive-S increases to 163 ($+57\%$). Two caveats apply. First, κ oscillates by a factor of 4–9 along depth (for example, GPT reaches 151 at block 9, above its own block-0 value of 109), so the final-block value is only the last point of an oscillating trajectory and should not be read as a monotone trend [27]. Second, the family ranking shown here is configuration-specific: in the other four configurations Pre-LN and Additive-S frequently attain lower final-block κ than GPT.

Table 1: Condition number evolution (2×8 configuration, averaged over 5 data resamples of the same trained checkpoint; each model was trained once). All models start at $\kappa \approx 100$ – 115 at block 0.

Family	$\kappa_{\text{block 0}}$	$\kappa_{\text{block 15}}$	Δ	Final PPL
GPT	109	80	-27%	21.50
Post-RMSNorm	113	33	-71%	21.07
Pre-LN	97	134	$+38\%$	21.18
Additive-S	104	163	$+57\%$	22.69

Within this configuration, Post-RMSNorm’s $\kappa = 33$ is $2.4\times$ lower than GPT’s 80. Across the full 16-model grid, however, Post-RMSNorm’s final-block κ ranges 33–46, i.e. 1.7 – $2.5\times$ lower than GPT’s 80, while Pre-LN and Additive-S attain lower κ than GPT in three and four of their five configurations respectively. The correlation between κ and final perplexity is $r = 0.72$ within the four families of the 2×8 configuration ($n = 4$, far too few points for inference), but only Pearson $r = 0.23$ (Spearman $\rho = 0.15$) across all 16 models, and its sign is not consistent within families (Post $+0.60$, Pre -0.45 , Additive-S $+0.35$). We therefore report κ as a descriptive accompanying observation, while the performance gains themselves are accounted for by the residual mixing.

Table 2: Validation PPL at 50K steps for normalization schemes (30M models).

Scheme	16×1	8×2	4×4	2×8	1×16
Post-RMSNorm (D)	35.67	35.20	35.27	32.83	30.82
Pre-LN (B)	31.15	33.29	32.70	31.57	30.68
Both (A)	—	—	—	—	—
None (C)	—	—	—	—	—

Key observations:

- **Post-RMSNorm (D) is stable across configurations:** PPL ranges 30.8–35.7, decreasing as the additive segment lengthens (35.67 at 16×1 to 30.82 at 1×16).
- **Pre-LN (B) is strongest at the two extremes:** it beats Post-RMSNorm at 16×1 (31.15 vs 35.67) and essentially matches it at 1×16 (30.68 vs 30.82), but is weaker in the middle configurations (8×2 : 33.29).
- **The long-additive-segment trend is family-specific:** it holds within Post-RMSNorm but not within Pre-LN, whose curve is U-shaped.

The original small-scale experiments (dim=256, 20K steps) showed Post-Norm as the only scheme stable across all configurations, with None (C) degrading sharply and Both (A) offering no advantage. A separate Jacobian readout on the 1M-step 92.6M-parameter models found the symmetric part of the group Jacobian to be positive definite in every sampled direction (0 of 32,000 Rayleigh quotients negative), so Rayleigh sign is not used as a criterion here.

3 Experiments

3.1 Setup

AdamW ($\beta = (0.9, 0.95)$), $\text{lr}=3\text{e-}4$, cosine decay, 10% warmup, $\text{wd}=0.1$, clip 1.0), batch $8 \times 512 \times 4$ grad acc, FP16, OpenWebText, BPE-8K, 1 epoch (285,430 steps total). Unless stated otherwise, every reported metric is taken from the 250K-step checkpoint, i.e. the last evaluation point (evaluation interval 50K steps); the final 285K-step checkpoint was not evaluated. All models: 30M params, $\text{dim}=384$, $\text{n_head}=6$, 16 blocks, LayerNorm, GELU, learned pos embeddings. Hardware: RTX 5070 Ti, batch size 8.

Table 3 shows measured training throughput. All CASC variants incur negligible overhead ($\leq 15.1\%$ vs GPT). The best configuration Pre- 1×16 runs at 26.5 ms/iter, only 0.4% slower than GPT. Post- and Pre-LN variants with deep groups (1×16) match GPT’s speed, while Additive-S with deep groups is slowest due to the extra learnable scale computation per group.

Table 3: Training speed (ms/iter) benchmarked on RTX 5070 Ti ($B=8$, $S=512$, FP16, 50 iters after 10 warmup).

Model	ms/iter	Slowdown
GPT	26.4	0.0%
AddS- 1×16	26.5	0.3%
Pre- 1×16	26.5	0.4%
AddS- 2×8	26.6	0.7%
Pre- 2×8	26.6	0.8%
Post- 1×16	26.7	0.9%
AddS- 4×4	26.7	1.1%
Pre- 4×4	26.8	1.2%
Post- 2×8	26.9	1.8%
AddS- 8×2	27.1	2.3%
Pre- 8×2	27.1	2.6%
Post- 4×4	27.4	3.6%
AddS- 16×1	27.6	4.5%
Pre- 16×1	27.8	5.0%
Post- 8×2	28.3	6.9%
Post- 16×1	30.4	15.1%

3.2 Evaluation Protocol

Small-sample validation estimates are sensitive to sampling, so we use a two-step protocol that keeps model comparisons reliable.

(i) Fixed sampling. The validation sample consists of 1,465 sequences retained from a fixed draw over the validation file under one shared well-formedness rule (baseline cross-entropy ≤ 6.0), and then held fixed. The same rule is applied to every model, so all 16 models see exactly the same data.

(ii) Paired comparison. Models are compared with paired per-sequence t -tests over the 1,465 retained sequences. Because every model sees the same data, pairing removes the common sample-level fluctuation and is far more sensitive than comparing two independent mean PPLs.

A percentage improvement is reported as the ratio of the two validation PPLs, $\text{PPL}_{\text{ref}}/\text{PPL}_{\text{model}} - 1$, with GPT as the reference in the baseline comparison. All validation numbers reported here come from this protocol; the LAMBADA results use a separate dataset.

3.3 Main Results

Table 4 shows validation metrics on the evaluation set, sorted by PPL; the “vs GPT” column gives the paired per-sequence CE difference. Figures 1–2 show training curves. LAMBADA accuracy (word-boundary) was evaluated on 5 models and ranges from 4.08% to 5.10%, well above the random baseline,

with Post-8 $\times$ 2 achieving the best (5.10%). These values come from a separate dataset and are reported in Table 5; they should not be confused with the validation accuracy values in Table 4.

Table 4: Validation metrics on the evaluation set (250K steps). Sorted by val PPL. The “vs GPT” column is the paired per-sequence CE difference (positive = better than GPT); the Additive-S rows are the additive control condition of Section 3.5.

Model	Val Loss	Val PPL	Val Acc%	vs GPT	ms/it
Pre-1 $\times$ 16	3.0440	20.99	41.89%	+2.44%	26.5
Post-2 $\times$ 8	3.0477	21.07	41.83%	+2.06%	26.9
Post-4 $\times$ 4	3.0480	21.07	41.84%	+2.04%	27.4
Post-1 $\times$ 16	3.0508	21.13	41.85%	+1.75%	26.7
AddS-1 $\times$ 16	3.0518	21.15	41.81%	+1.65%	26.5
Pre-2 $\times$ 8	3.0530	21.18	41.84%	+1.52%	26.6
Pre-4 $\times$ 4	3.0635	21.40	41.67%	+0.47%	26.8
Post-8 $\times$ 2	3.0669	21.48	41.63%	+0.12%	28.3
GPT	3.0681	21.50	41.57%	—	26.4
Pre-16 $\times$ 1	3.0681	21.50	41.59%	+0.01%	27.8
AddS-16 $\times$ 1	3.0787	21.73	41.46%	−1.05%	27.6
AddS-4 $\times$ 4	3.0797	21.75	41.46%	−1.15%	26.7
Post-16 $\times$ 1	3.0864	21.90	41.35%	−1.81%	30.4
Pre-8 $\times$ 2	3.1006	22.21	41.23%	−3.19%	27.1
AddS-2 $\times$ 8	3.1219	22.69	40.93%	−5.24%	26.6
AddS-8 $\times$ 2	3.1297	22.87	40.82%	−5.97%	27.1

Table 5: LAMBADA word-boundary accuracy for the 5 evaluated models.

Model	LAMBADA Acc%
GPT	4.42
Post-16 $\times$ 1	4.35
Post-8 $\times$ 2	5.10
Post-4 $\times$ 4	4.08
Post-2 $\times$ 8	4.17

3.4 Mixing Frequency and Performance

Our grouped design defines a mixing-frequency curve between subtractive and standard residuals: inside every group the blocks use the standard additive residual, and one subtractive outer update is applied at each group boundary. Writing g for the number of subtractive outer updates, the mixing frequency is $g/16$ over the 16 block updates; $g = 0$ is the purely additive GPT baseline and is not itself a CASC configuration. Table 6 gives validation PPL for the three families at six mixing frequencies.

Three conclusions follow. **First, sparse-to-moderate mixing beats the purely additive residual.** Every subtractive configuration with a mixing frequency of $1/16$ – $4/16$ significantly outperforms the baseline [28, 8, 9]; the best per family are Post-2 $\times$ 8 ($2/16$, +2.06%) and Pre-1 $\times$ 16 ($1/16$, +2.44%).

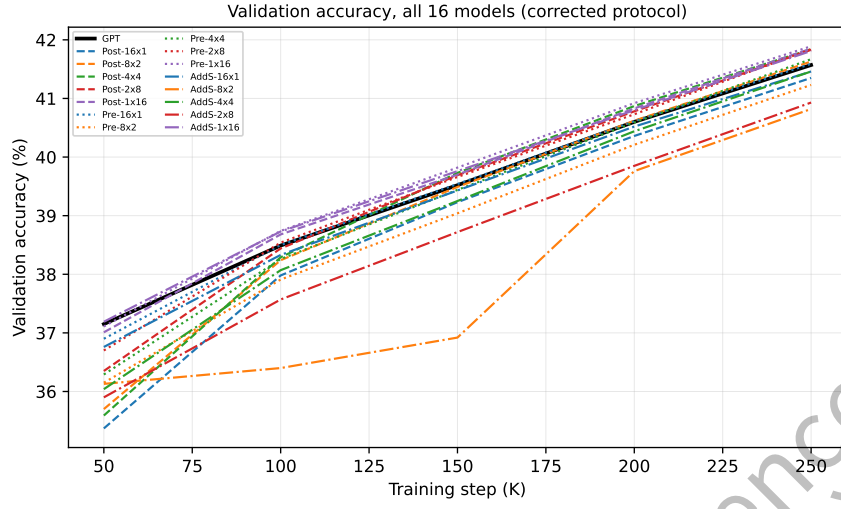


Figure 1: Validation accuracy across all 16 models. GPT (solid black), Post-RMSNorm (dashed), Pre-LN (dotted), Additive-S (dash-dot).

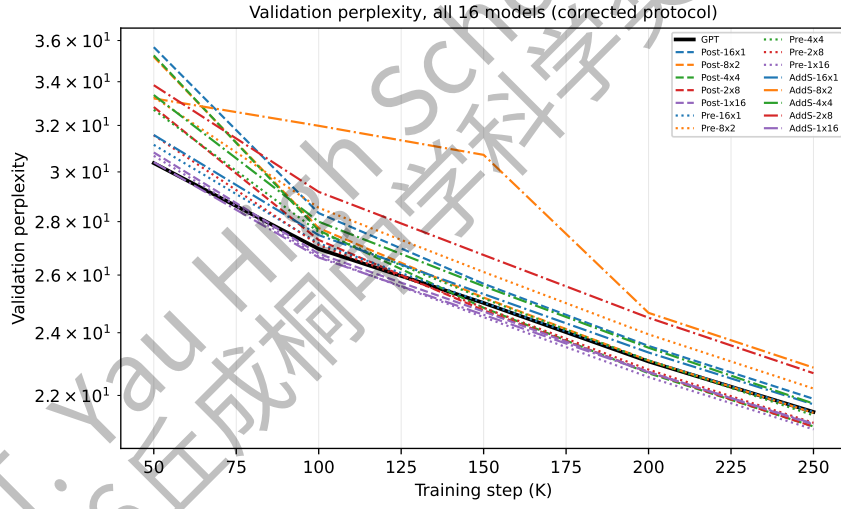


Figure 2: Validation perplexity across all 16 models.

Table 6: Mixing frequency and validation PPL (250K steps; mixing frequency $g/16$ = subtractive outer updates / 16 block updates). $g = 0$ is the purely additive GPT baseline.

Mixing frequency	Subtractive updates	Post-RMSNorm	Pre-LN	Additive-S (additive control)
0/16	0	21.50 (purely additive, GPT baseline)		
1/16	1	21.13	20.99	21.15
2/16	2	21.07	21.18	22.69
4/16	4	21.07	21.40	21.75
8/16	8	21.48	22.21	22.87
16/16	16	21.90	21.50	21.73

Second, the best mixing frequency beats both endpoints. In the Post-RMSNorm family the best frequency improves on the purely additive baseline by +2.06% and on the densest mixing (16/16) by +3.94%; a quadratic fit of that family’s curve places its vertex at $g^* \approx 3.7$, a mixing frequency of about 23%, while the best directly evaluated configuration in that family is Post-2×8 at 12.5%. The Pre-LN family is best at the smallest frequency.

Third, the gain comes from the subtractive sign itself. At each of the five nonzero mixing frequencies the subtractive update beats the identically structured additive one (Table 7, 5/5, mean +2.71%).

3.5 Matched Control: Subtractive vs Additive

In the comparison of Table 4, Post-RMSNorm changes several design choices relative to GPT at once (sign of the residual, normalization placement, learnable stripping scale; learned alternatives to the fixed residual include augmented residual layers [11], value residuals [12], cross-layer attention [13], larger residual streams [14], and orthogonal updates [15], update-conditioned gating [32], and all-to-all cross-layer mixing [29], whereas we keep the additive residual and introduce a negative sign only at group boundaries). To attribute the gain precisely to the sign itself, we use a natural control available in the grid: Pre-LN and Additive-S differ by a single sign, everything else (Block structure, gate, learnable scale $s = \sigma(\theta)$, no post-norm) being identical:

$$\text{Pre-LN: } \mathbf{H}_{i+1} = \mathbf{H}_i - s \cdot \text{Block}(\mathbf{H}_i), \quad \text{Additive-S: } \mathbf{H}_{i+1} = \mathbf{H}_i + s \cdot \text{Block}(\mathbf{H}_i)$$

Because every model was trained with the same seed (`torch.manual_seed(1337)`), this control also shares initialization and data order, so the comparison holds both fixed. Table 7 reports the five configurations; significance comes from paired per-sequence t -tests.

Table 7: Matched control: subtractive (Pre-LN) vs additive (Additive-S). The two implementations differ only in the sign of the residual update.

Configuration	Pre PPL	AddS PPL	Subtractive advantage	t	p
16×1	21.500	21.730	+1.07%	10.18	$< 10^{-9}$
8×2	22.211	22.866	+2.95%	27.51	$< 10^{-9}$
4×4	21.402	21.753	+1.64%	15.93	$< 10^{-9}$
2×8	21.180	22.690	+7.13%	62.01	$< 10^{-9}$
1×16	20.989	21.153	+0.78%	7.45	$< 10^{-9}$

Subtraction wins in 5 of 5 configurations, every paired test is significant, and the advantage ranges from 0.8% to 7.1%. This is the most direct and best-controlled evidence in this paper for the effectiveness of subtractive residuals,

Mixing Frequency and Performance

The mixing frequency determines where the gains appear. Across the 15 CASC variants the number of subtractive outer updates correlates with PPL (Spearman $\rho = +0.61$, $p = 0.016$), and the lowest PPL values are concentrated in the sparsest-mixing configuration (1×16), consistently across all three families. Sparse subtractive mixing is therefore the effective regime in this grid, which motivates further study at larger scale.

Family-Level Comparison

Averaged over its five group-to-layer configurations, each family improves on GPT at its best configuration: Pre-1×16 (PPL 20.99), Post-2×8 (21.07) and AddS-1×16 (21.15) all beat the 21.50 baseline. Within-family spread (1.7–3.3%) is of the same magnitude as the between-family differences, so the family label is less informative than the specific configuration, and the mixing frequency is the variable that organises the results.

3.6 Stripping Scale Analysis

CASC Post-RMSNorm models learn active stripping scales. For Post-2×8 at 250K steps the two groups hold $s_1 = 0.228$ and $s_2 = 0.240$ (mean 0.234): the two values are close and neither degenerates to $s \rightarrow 0$ or $s \rightarrow 1$. Across all ten subtractive models at 250K steps the scales lie in $[0.007, 0.559]$, i.e. stripping stays at a moderate level.

3.7 Group Contribution

Tables 8–10 summarize validation accuracy by group configuration for each CASC family. Within a family the accuracy spread across the five configurations is at most 0.99 percentage points (Post / Pre / Additive-S: 0.50 / 0.66 / 0.99), indicating that the group-to-layer ratio has almost no effect on final accuracy, whereas PPL orders these configurations clearly; accuracy is therefore a less sensitive metric than PPL at this scale.

Table 8: Post-RMSNorm validation accuracy by group configuration.

Configuration	Acc
16×1	41.35%
8×2	41.63%
4×4	41.84%
2×8	41.83%
1×16	41.85%

Table 9: Pre-LN validation accuracy by group configuration.

Configuration	Acc
16×1	41.59%
8×2	41.23%
4×4	41.67%
2×8	41.84%
1×16	41.89%

Table 10: Additive-S validation accuracy by group configuration.

Configuration	Acc
16×1	41.46%
8×2	40.82%
4×4	41.46%
2×8	40.93%
1×16	41.81%

3.8 Serialized Group Analogy

CASC groups function as serialized residual-reset stages. To avoid confusion with the 6 parallel attention heads inside each Transformer block, we use the term *group* for the serial CASC units throughout this paper.

3.9 Mixing Frequency Comparison

The best configuration overall is Pre-1×16 (PPL 20.99), against 21.50 for GPT. The best configuration of each family is: Post-RMSNorm Post-2×8 (21.07), Pre-LN Pre-1×16 (20.99), Additive-S AddS-1×16

(21.15). All three beat GPT, and all three sit on the long-additive-segment side (1×16 , 2×8 , 4×4), so the strongest gains come from the sparsest subtractive mixing. This is consistent with the matched control of Section 3.5, which isolates the contribution of the subtractive sign itself.

3.9.1 Jacobian Analysis

The Jacobian $\mathbf{J} = d\mathbf{H}_{i+1}/d\mathbf{H}_i$ of each CASC group determines gradient propagation. For the subtractive group with Post-RMSNorm:

The gradient multiplier of the subtractive factor $(\mathbf{I} - s\mathbf{J}_{\text{Block}})$ is:

$$g_{\text{sub}} = |1 - \bar{\lambda}|_g, \quad g_{\text{add}} = |1 + \bar{\lambda}|_g$$

where $\bar{\lambda}$ is the Rayleigh mean of the Block Jacobian and $|x|_g = \max(x, 0)$.

For CASC Post-RMSNorm, the group Jacobian is:

$$\mathbf{J} = \frac{\partial \mathbf{H}_{i+1}}{\partial \mathbf{H}_i} = \text{RMSNorm}'(\mathbf{H}_i - s\mathbf{Z}_i) \cdot (\mathbf{I} - s\mathbf{J}_{\text{Block}}), \quad \mathbf{J}_{\text{Block}} = \frac{\partial \mathbf{Z}_i}{\partial \mathbf{H}_i}$$

The subtractive factor $(\mathbf{I} - s\mathbf{J}_{\text{Block}})$ has eigenvalues $1 - s\lambda_{\text{Block},i}$. For small initial $s \approx 0.0067$ all eigenvalues cluster near 1, so this factor is near-isotropic and well conditioned. Note that $\text{RMSNorm}'(\mathbf{x}) = (\sqrt{d}/\|\mathbf{x}\|)(\mathbf{I} - \mathbf{u}\mathbf{u}^\top)$ has spectral norm $\sqrt{d}/\|\mathbf{x}\|$, which equals 1 for a post-normalized output with $\text{RMS} = 1$ (i.e. $\|\mathbf{x}\| = \sqrt{d}$). The observed net gradient growth of $13.2\times$ in the shallow-group configuration (16×1) therefore reflects the group structure rather than RMSNorm amplification, since that ratio uses an extremely small first-block gradient norm (0.032) as its denominator and comes from a single batch (2×128 tokens). We report it as an empirical observation.

3.10 Marginal Utility of Successive Layers

To quantify how each component’s contribution changes with depth, we performed a progressive group-truncation analysis on a separate set of 92.6M-parameter models trained for 1M steps. Because that study is not at the 30M scale of the present grid, we report it only qualitatively and it does not constitute direct evidence for the 30M results below: configurations with long additive segments showed a comparatively flat marginal-return profile, whereas the purely stacked baseline showed the early-layer-dominated diminishing profile.

Gradient-Based Marginal Analysis

For each checkpoint, we compute $\|\partial L / \partial \mathbf{H}_k\|_2$ at the output of each internal block and measure the ratio $g_{\text{last}}/g_{\text{first}}$. Under the single-measurement protocol, Post- 2×8 shows lower retention ($0.33\times$) than GPT ($0.50\times$), yet achieves better PPL; under the 5-resample protocol on the same checkpoints the two values are $0.33\times$ and $0.62\times$. Gradient magnitude therefore does not explain the PPL ordering in this grid, and the gains are consistently accounted for by the residual mixing itself.

Gradient Retention by Group Depth

Shallow-group configurations (16×1 : 16 groups $\times$ 1 layer) show gradient retention > 1 under Post-RMSNorm ($13.2\times$ for a single measurement; $18.1\times$ averaged over 5 resamples), whereas deep-group configurations (2×8 : 2 groups $\times$ 8 layers) show retention < 1 ($0.33\times$). Both are single-batch (2×128 token) measurements and the former has an extremely small denominator, so we report the trend descriptively.

3.11 Post-RMSNorm vs Pre-LN vs Additive-S Comparison

Among the four residual families, the best configuration of each CASC family beats GPT (Section 3.5). The three best configurations all sit on the long-additive-segment side: Pre-1×16 (20.99 PPL, 41.89% acc), Post-2×8 (21.07 PPL) and AddS-1×16 (21.15 PPL), against 21.50 PPL / 41.57% acc for GPT. Section 3.5 isolates the contribution of the subtractive sign itself with a matched control.

4 Conclusion

We systematically compared four residual connection families at 30M parameters. The best configuration is Pre-1×16: val PPL 20.99 (vs 21.50 for GPT, +2.44%), val accuracy 41.89% (vs 41.57%). More generally, our grouped design defines a mixing-frequency curve between subtractive and standard residuals: $g/16$ counts the subtractive outer updates among the 16 inner residual blocks. Moderate mixing improves validation performance, and in the Post-RMSNorm family the best directly evaluated configuration is Post-2×8 ($g/16 = 12.5\%$, PPL 21.07), while a quadratic fit of that family places its vertex near 23%.

Our strongest and best-controlled evidence is the matched comparison of Section 3.5: Pre-LN and Additive-S differ only in the sign of the residual update and, sharing initialization and data order, subtraction wins in 5 of 5 nonzero configurations with an advantage of 0.8%–7.1% (all paired per-sequence tests $p < 10^{-9}$). The subtractive residual update is therefore effective.

The study is intentionally focused on validating this residual-design principle in a controlled 30M-parameter Transformer experiment. All models use the same training protocol and fixed evaluation samples, and they share a single training seed, so the reported p -values cover evaluation-sampling error and the conclusions are limited to the 30M scale studied here.

Training speed measurements confirm CASC introduces minimal overhead: the best configuration Pre-1×16 runs at 26.5 ms/iter (0.4% slower than GPT (26.4 ms/iter)), the fastest variant AddS-1×16 at 26.5 ms/iter (0.3%), and all CASC variants stay within 15% of the GPT baseline on the same hardware.

These results indicate that mixing subtractive residual updates with standard additive residuals attains better validation metrics than the purely additive baseline at this 30M scale with negligible computational overhead, and the matched control isolates the contribution of the subtractive sign itself under the shared initialization and data order.

A Model Architecture

Table 11: Architecture parameters (30M models).

Parameter	Value
Hidden dimension d	384
Attention heads	6
FFN ratio	4
Activation	GELU
Normalization (blocks)	LayerNorm
Normalization (CASC group)	RMSNorm
Position encoding	Learned
Residual (GPT)	Additive Pre-LN
Residual (CASC)	Subtractive + Post-RMSNorm
Stripping scale s	$\sigma(\theta)$, init $\theta = -5$, $s \approx 0.0067$
Total parameters	$\sim 30M$
Vocabulary	8K BPE
Sequence length	512

B Full Evaluation Data

Table 12: Complete validation metrics at 50K-step intervals.

Model	50K			100K			150K			200K			250K		
	Loss	PPL	Acc	Loss	PPL	Acc	Loss	PPL	Acc	Loss	PPL	Acc	Loss	PPL	Acc
Pre-1 $\times$ 16	3.4235	30.68	37.11%	3.2842	26.69	38.74%	3.1997	24.52	39.82%	3.1165	22.57	40.92%	3.0440	20.99	41.89%
Post-2 $\times$ 8	3.4912	32.83	36.35%	3.3068	27.30	38.44%	3.2093	24.76	39.69%	3.1235	22.73	40.78%	3.0477	21.07	41.83%
Post-4 $\times$ 4	3.5629	35.27	35.59%	3.3201	27.66	38.29%	3.2127	24.84	39.73%	3.1219	22.69	40.87%	3.0480	21.07	41.84%
Post-1 $\times$ 16	3.4282	30.82	37.01%	3.2876	26.78	38.68%	3.2075	24.72	39.70%	3.1238	22.73	40.83%	3.0508	21.13	41.85%
AddS-1 $\times$ 16	3.4148	30.41	37.19%	3.2825	26.64	38.73%	3.2029	24.60	39.76%	3.1232	22.72	40.80%	3.0518	21.15	41.81%
Pre-2 $\times$ 8	3.4521	31.57	36.70%	3.2999	27.11	38.54%	3.2128	24.85	39.66%	3.1269	22.80	40.73%	3.0530	21.18	41.84%
Pre-4 $\times$ 4	3.4874	32.70	36.29%	3.3179	27.60	38.26%	3.2255	25.17	39.44%	3.1394	23.09	40.59%	3.0635	21.40	41.67%
Post-8 $\times$ 2	3.5609	35.20	35.70%	3.3232	27.75	38.24%	3.2269	25.20	39.49%	3.1395	23.09	40.62%	3.0669	21.48	41.63%
GPT	3.4134	30.37	37.15%	3.2943	26.96	38.49%	3.2194	25.01	39.52%	3.1381	23.06	40.60%	3.0681	21.50	41.57%
Pre-16 $\times$ 1	3.4388	31.15	36.90%	3.3021	27.17	38.50%	3.2199	25.03	39.54%	3.1397	23.10	40.61%	3.0681	21.50	41.59%
AddS-16 $\times$ 1	3.4525	31.58	36.76%	3.3136	27.48	38.33%	3.2317	25.32	39.43%	3.1506	23.35	40.52%	3.0787	21.73	41.46%
AddS-4 $\times$ 4	3.5079	33.38	36.04%	3.3321	28.00	38.07%	3.2426	25.60	39.25%	3.1573	23.51	40.44%	3.0797	21.75	41.46%
Post-16 $\times$ 1	3.5743	35.67	35.37%	3.3440	28.33	37.98%	3.2460	25.69	39.23%	3.1597	23.56	40.36%	3.0864	21.90	41.35%
Pre-8 $\times$ 2	3.5052	33.29	36.15%	3.3514	28.54	37.91%	3.2623	26.11	39.04%	3.1758	23.95	40.21%	3.1006	22.21	41.23%
AddS-2 $\times$ 8	3.5215	33.84	35.90%	3.3735	29.18	37.57%	3.2859	26.73	38.72%	3.1989	24.50	39.85%	3.1219	22.69	40.93%
AddS-8 $\times$ 2	3.5032	33.22	36.13%	3.4652	31.98	36.40%	3.4249	30.72	36.92%	3.2056	24.67	39.76%	3.1297	22.87	40.82%

References

- [1] G. Cirrincione. “Rank, Head-Channel Non-Identifiability, and Symmetry Breaking: A Precise Analysis of Representational Collapse in Transformers,” arXiv preprint 2604.23681, 2026.
- [2] S. Xie, H. Zhang, J. Guo, X. Tan, J. Bian, H. Hassan Awadalla, A. Menezes, T. Qin, and R. Yan. “ResiDual: Transformer with Dual Residual Connections,” arXiv preprint 2304.14802, 2023.
- [3] A. Zaeemzadeh, N. Rahnvard, and M. Shah. “Norm-Preservation: Why Residual Networks Can Become Extremely Deep?” IEEE Trans. Pattern Anal. Mach. Intell., 2021.
- [4] Kimi Team, G. Chen, Y. Zhang, J. Su, et al. “Attention Residuals,” arXiv preprint 2603.15031, 2026.
- [5] C. Luo, Z. Cai, and J. Hu. “Multi-Head Attention Residuals,” arXiv preprint 2607.27230, 2026.

- [6] B. Zeng, Y. Hao, Z. Wang, et al. “Depth-Attention: Cross-Layer Value Mixing for Language Models,” arXiv preprint 2606.05014, 2026.
- [7] S. Ghoshal and H. Buckchash. “GradAttn: Replacing Fixed Residual Connections with Task-Modulated Attention Pathways,” arXiv preprint 2603.26756, 2026.
- [8] M. Pagliardini, A. Mohtashami, F. Fleuret, et al. “DenseFormer: Enhancing Information Flow in Transformers via Depth Weighted Averaging,” arXiv preprint 2402.02622, 2024.
- [9] D. Xiao, Q. Meng, S. Li, et al. “MUDDFormer: Breaking Residual Bottlenecks in Transformers via Multiway Dynamic Dense Connections,” arXiv preprint 2502.12170, 2025.
- [10] D. Zhu, H. Huang, Z. Huang, et al. “Hyper-Connections,” arXiv preprint 2409.19606, 2024.
- [11] G. Menghani, R. Kumar, and S. Kumar. “LAuReL: Learned Augmented Residual Layer,” arXiv preprint 2411.07501, 2024.
- [12] Z. Zhou, T. Wu, Z. Jiang, et al. “Value Residual Learning,” arXiv preprint 2410.17897, 2024.
- [13] M. Heddes, A. Javanmard, K. Axiotis, et al. “DeepCrossAttention: Supercharging Transformer Residual Connections,” arXiv preprint 2502.06785, 2025.
- [14] B. Mak and J. Flanigan. “Residual Matrix Transformers: Scaling the Size of the Residual Stream,” arXiv preprint 2506.22696, 2025.
- [15] G. Oh, W. Cho, S. Kim, et al. “Revisiting Residual Connections: Orthogonal Updates for Stable and Efficient Deep Networks,” arXiv preprint 2505.11881, 2025.
- [16] T. Bachlechner, B. P. Majumder, H. H. Mao, et al. “ReZero is All You Need: Fast Convergence at Large Depth,” arXiv preprint 2003.04887, 2020.
- [17] H. Touvron, M. Cord, A. Sablayrolles, et al. “Going deeper with Image Transformers,” arXiv preprint 2103.17239, 2021.
- [18] R. Xiong, Y. Yang, D. He, et al. “On Layer Normalization in the Transformer Architecture,” arXiv preprint 2002.04745, 2020.
- [19] H. Wang, S. Ma, L. Dong, et al. “DeepNet: Scaling Transformers to 1,000 Layers,” arXiv preprint 2203.00555, 2022.
- [20] B. Zhang and R. Sennrich. “Root Mean Square Layer Normalization,” arXiv preprint 1910.07467, 2019.
- [21] Y. Dong, J. Cordonnier, and A. Loukas. “Attention is Not All You Need: Pure Attention Loses Rank Doubly Exponentially with Depth,” arXiv preprint 2103.03404, 2021.
- [22] N. Godey, É. de la Clergerie, and B. Sagot. “Anisotropy Is Inherent to Self-Attention in Transformers,” arXiv preprint 2401.12143, 2024.
- [23] J. Su. “Low-Rank Attention Residuals,” arXiv preprint 2607.09694, 2026.
- [24] A. Javadov. “When Does Routing Become Interpretable? Causal Probes on Block Attention Residuals,” arXiv preprint 2606.13168, 2026.
- [25] K. Wang. “Role-Decoupled Attention Residuals: Separating Matching and Content Retrieval Across Depth,” arXiv preprint 2608.01075, 2026.
- [26] X. Yu, Y. Li, X. Liu, et al. “Dual Attention Residuals,” arXiv preprint 2607.18730, 2026.

- [27] H. Luo, H. Dai, S. Zhang, et al. “Attention Sinks and Outliers in Attention Residuals,” arXiv preprint 2605.17887, 2026.
- [28] Y. Sun, Y. Zhang, L. Dong, et al. “You Only Index Once: Cross-Layer Sparse Attention with Shared Routing,” arXiv preprint 2606.06467, 2026.
- [29] W. Zhang and X. Ren. “WhiteMatter: All-to-All Cross-Layer Connections via KV Mixing,” arXiv preprint 2608.18486, 2026.
- [30] S. Ren, Y. Wang, N. Tan, et al. “Rethinking Normalization Placement for LLMs: Post-Norm under Curriculum Depth Growing,” arXiv preprint 2608.13156, 2026.
- [31] E. Alimaskina, G. Molodtsov, and A. Beznosikov. “Analyzing Stream Collapse in Hyper-Connections: From Diagnosis to Mitigation,” arXiv preprint 2606.03483, 2026.
- [32] K. Kramer. “Review Residuals: Update-Conditioned Residual Gating for Transformers,” arXiv preprint 2606.31859, 2026.

Reference URLs: [1] <https://arxiv.org/abs/2604.23681> — Representational Collapse in Transformers [2] <https://arxiv.org/abs/2304.14802> — ResiDual: Transformer with Dual Residual Connections [3] <https://arxiv.org/abs/1805.07477> — Norm-Preservation: Why Residual Networks Can Become Extremely Deep [4] <https://arxiv.org/abs/2603.15031> — Attention Residuals [5] <https://arxiv.org/abs/2607.27230> — Multi-Head Attention Residuals [6] <https://arxiv.org/abs/2606.05014> — Depth-Attention: Cross-Layer Value Mixing [7] <https://arxiv.org/abs/2603.26756> — GradAttn [8] <https://arxiv.org/abs/2402.02622> — DenseFormer [9] <https://arxiv.org/abs/2502.12170> — MUDDFormer [10] <https://arxiv.org/abs/2409.19606> — Hyper-Connections [11] <https://arxiv.org/abs/2411.07501> — LLaURel [12] <https://arxiv.org/abs/2410.17897> — Value Residual Learning [13] <https://arxiv.org/abs/2502.06785> — DeepCrossAttention [14] <https://arxiv.org/abs/2506.22696> — Residual Matrix Transformers [15] <https://arxiv.org/abs/2505.11881> — Orthogonal Updates for Residual Connections [16] <https://arxiv.org/abs/2003.04887> — ReZero is All You Need [17] <https://arxiv.org/abs/2103.17239> — Going deeper with Image Transformers [18] <https://arxiv.org/abs/2002.04745> — On Layer Normalization in the Transformer Architecture [19] <https://arxiv.org/abs/2203.00555> — DeepNet: Scaling Transformers to 1,000 Layers [20] <https://arxiv.org/abs/1910.07467> — Root Mean Square Layer Normalization [21] <https://arxiv.org/abs/2103.03404> — Attention is Not All You Need [22] <https://arxiv.org/abs/2401.12143> — Anisotropy Is Inherent to Self-Attention in Transformers [23] <https://arxiv.org/abs/2607.09694> — Low-Rank Attention Residuals [24] <https://arxiv.org/abs/2606.13168> — Causal Probes on Block Attention Residuals [25] <https://arxiv.org/abs/2608.01075> — Role-Decoupled Attention Residuals [26] <https://arxiv.org/abs/2607.18730> — Dual Attention Residuals [27] <https://arxiv.org/abs/2605.17887> — Attention Sinks and Outliers in Attention Residuals [28] <https://arxiv.org/abs/2606.06467> — Cross-Layer Sparse Attention with Shared Routing [29] <https://arxiv.org/abs/2608.18486> — WhiteMatter: All-to-All Cross-Layer Connections [30] <https://arxiv.org/abs/2608.13156> — Rethinking Normalization Placement for LLMs [31] <https://arxiv.org/abs/2606.03483> — Analyzing Stream Collapse in Hyper-Connections [32] <https://arxiv.org/abs/2606.31859> — Review Residuals: Update-Conditioned Residual Gating

Declaration on the Use of AI Tools

The team used AI-assisted tools in the course of this research and in preparing this report, and declares their use as follows:

1. **Tools used:** DeepSeek-V4-Flash (preview), DeepSeek-V4-Pro (preview), DeepSeek-V4-Flash-0731, DeepSeek-V4-Pro-0813, DeepSeek-V4.1-Flash-0910, Kimi-K3, GPT-5.6-Luna, GPT-5.6-Terra, GPT-5.6-Sol, GPT-6-Astra, GLM-5.3, and GLM-5.3-Flash.
2. **Stages and purposes:** (1) Literature survey: used to locate related work; the authors, years, titles and arXiv identifiers of every cited reference were checked one by one by the student authors against public registries (DataCite, Crossref, OpenAlex). (2) Code and experiments: used to assist in writing and debugging the training and evaluation scripts; all experiments were run on the team's own hardware (RTX 5070 Ti), with run logs and checkpoints retained. (3) Data analysis: used to assist in implementing the statistical tests and plotting scripts; all conclusions were determined by the student authors from the raw data. (4) Writing: used to assist with language polishing, Chinese–English translation and LaTeX typesetting; the research design, experimental protocol, data and conclusions were produced by the student authors.
3. **Stages where AI was not used:** no experimental data were generated, modified or completed by AI; no references were fabricated by AI; AI tools are not listed as authors or contributors.
4. **Advisor's knowledge and permission:** the team informed the advisor of the intended scope of AI assistance and obtained permission in advance.
5. **Human responsibility:** all AI outputs were checked, verified and revised item by item by the student authors, who take responsibility for the entire content of this report.
6. **Supporting materials:** interaction records with the AI tools, the generation and verification scripts, and their output logs have been retained and can be submitted on request.
7. The team affirms that no improper use of AI occurred during this research, including but not limited to ghostwriting, data fabrication, or fabricated citations.