Negative Attention is What Pre-trained Transformers Are Missing

Anonymous ACL submission

Abstract

Pre-trained language models over-attend to BOS tokens and high-frequency function words, diluting attention from the semantically informative tokens that downstream NLP tasks rely on. We trace this *attention noise* to a structural limitation of softmax: every attention weight is confined to $[0, 1]$, so a fine-tuned model has no architectural way to cancel attention on a distractor without compensating elsewhere. We identify *negative attention capacity*—the ability to assign *signed* effective weights—as the primitive that prior noise-suppression methods implicitly target, and propose Differential Attention Adaption (DAA): a single learnable matrix $W_{\mathrm{DAA}} \in \mathbb{R}^{d_k \times d_k}$ inserted into the query-key logit space, producing a secondary softmax map that is subtracted from the original attention. DAA is a minimal realization of this primitive: (i) cross-entropy gradients route the secondary branch onto distractor tokens without auxiliary supervision; (ii) DAA contains existing input-differentiation variants (DiffK / DiffQ) as a canonical minimum; (iii) any post-softmax linear correction is blocked by an information-theoretic obstruction. With $< 1\%$ extra parameters, DAA shows its largest gains where attention noise is the bottleneck: $+4.6$ accuracy on needle-in-a-haystack retrieval over GPT-2 Base, the highest LAMBADA last-word prediction accuracy among the methods we test, and a 56% reduction in attention entropy that concentrates on the BOS / function-word distractors. On Llama-3.2-1B and Llama-3.1-8B, DAA preserves common-sense reasoning while showing the same gain pattern. Pre-trained language models do not need bigger attention—they need a negative one.

1 Introduction

Attention can select, but it cannot suppress. Pre-trained language models systematically allocate disproportionate attention to BOS tokens and high-frequency function words—a phenomenon documented as "attention sinks" (Xiao et al., 2024; Gu et al., 2025) and linked to "massive activations" in the residual stream (Sun et al., 2024). The root cause is structural: every attention head is a softmax that assigns each key a non-negative weight summing to one, so the layer is good at *choosing* the most relevant token but has no architectural mechanism for actively *suppressing* a distractor. To reduce attention on the BOS sink or on a function word that has crowded out a semantically informative token, the model must redistribute mass to some other position—a zero-sum shuffle inside the probability simplex $\Delta^N = \{p \in \mathbb{R}^N_{\geq 0} : \sum_i p_i = 1\}$. We argue that this is the structural origin of *attention noise*: probability mass on distractor tokens that the architecture is, by construction, unable to remove. Concrete instance: on the LAMBADA last-word prediction task (Paperno et al., 2016), a fine-tuned GPT-2 layer leaves 2.01 nats of attention entropy diffused across context glue, and the base model identifies the correct continuation only 47.3% of the time (Liu et al., 2023; Lu et al., 2021). The information is there; what is missing is the architectural capacity to cancel.

What existing approaches miss. The Differential Transformer (Ye et al., 2025) computes two parallel softmax maps and subtracts one from the other, baking a subtractive branch into the architecture; this design is realized only as a pre-training-time choice, leaving the catalog of existing pre-trained checkpoints untouched. DEX (Kong et al., 2025) applies a learnable linear correction to the value-aggregated output, $O' = AV(I - \lambda(t)W_{\mathrm{DEX}})$; because this acts *after* softmax has produced a non-negative attention map, the Data Processing Inequality (Beaudry and Renner, 2011) prevents the correction from recovering negative effective attention—any signal lost to noise during the irreversible softmax mixing remains lost. LoRA

and standard fine-tuning update W_Q, W_K, W_V but leave the softmax non-negativity untouched, so they too can only reshuffle mass inside Δ^N. A complementary line of work *replaces* the softmax itself (Qiu et al., 2025; Zuhri et al., 2025), eliminating sinks at the activation-function level—but at the cost of pre-training or non-trivial re-training, which puts existing checkpoints out of reach. None of these mechanisms supplies the architecture with a subtractive primitive at the level where noise is generated *and* that is retrofittable to pre-trained models.

Our reframe: negative attention capacity as the missing primitive. We identify *negative attention capacity*—the architectural ability to assign *signed* (i.e. possibly negative) effective attention weights—as the missing primitive that all noise-suppression mechanisms implicitly attempt to inject. Differential Attention Adaption (DAA) is the minimal-parameter implementation of this primitive: a single learnable matrix $W_{\mathrm{DAA}} \in \mathbb{R}^{d_k \times d_k}$ inserted into the Q-K logit space, producing a second softmax map A_- that is subtracted from the primary map A_+. Less than 1% extra parameters; applicable to any pre-trained Transformer.

Contributions. Our contributions are fourfold. Conceptually, we identify the non-negativity of softmax attention as the structural origin of attention noise, and *negative attention capacity* as the corresponding missing primitive (§3.1). Methodologically, we propose DAA as a minimal-parameter implementation of negative capacity at the Q-K logit position, and prove (Theorem 3.1) that the K-side input-differentiation family (DiffK / DiffQ) is a strict over-parameterization of DAA; the empirical advantage of DAA can therefore be read as evidence of the inductive bias the smaller class encodes (§3.2). Theoretically, we give a gradient-decomposition argument (Theorem 3.6) showing that cross-entropy training drives the negative branch toward distractor tokens rather than the gold token (§3.4); we further show (Proposition 3.3) that no right-multiplication on the value-aggregated output can produce negative effective per-key attention, which rules out post-softmax linear corrections such as DEX as a source of genuine negative capacity (§3.3). Empirically, DAA shows its largest gains where attention noise is the bottleneck—needle-in-a-haystack retrieval ($+4.6$ over Base on GPT-2), LAMBADA accuracy, and attention entropy reduced from 2.01 to 0.88 nats—and preserves general reasoning performance on Llama-3.2-1B and Llama-3.1-8B benchmarks (§4).

Figure 1 shows what attention noise looks like on a schematic example and how DAA's negative branch cancels it; a side-by-side comparison of DAA against DIFF Transformer, DEX, DiffQ/K/V on noise-cancellation effectiveness, retrofittability, and parameter count appears in Appendix A (Table 3, Figure 4).

2 Background and Related Work

2.1 The Non-Negativity Constraint of Softmax Attention

Given an input sequence $X \in \mathbb{R}^{N \times d_{\mathrm{model}}}$, a single attention head computes $Q = XW_Q$, $K = XW_K$, $V = XW_V$, and outputs $O = AV$ with $A = \mathrm{softmax}(QK^\top/\sqrt{d_k})$ applied row-wise. By construction, each row of A lies in the probability simplex $\Delta^N = \{p \in \mathbb{R}^N_{\geq 0} : \sum_i p_i = 1\}$, i.e. $A^{ij} \geq 0$ for all i, j. The output $o_i = \sum_j A^{ij} v_j$ is therefore a non-negative convex combination of the value vectors. Two consequences follow that drive the rest of this paper. (i) The layer can *redistribute* attention but cannot *subtract*: reducing A^{ij} for any key j requires compensating increase on some other key. (ii) Any noise mixed into the attention scores by the bilinear $QK^\top$ interaction is funneled through the softmax non-linearity, after which the Data Processing Inequality (Beaudry and Renner, 2011) prevents any linear post-processing from recovering the signal lost during this irreversible mixing. We formalize these statements in §3.3 and use them to position DAA against prior work.

2.2 Differential Transformer

The Differential Transformer (Ye et al., 2025) equips a from-scratch model with negative capacity by generating two parallel attention maps and subtracting:

$$\begin{aligned} [Q_1; Q_2] = XW_Q, \quad [K_1; K_2] = XW_K, \\ V = XW_V, \\ O' = (A_1 - \lambda A_2)\, V, \end{aligned} \tag{1}$$

where $A_\ell = \mathrm{softmax}(Q_\ell K_\ell^\top/\sqrt{d_k})$ for $\ell \in \{1, 2\}$ and λ is a learnable scalar. This realizes the negative-capacity primitive of §3.1 as a pre-training-time architectural choice. The limitation is structural: existing pre-trained checkpoints cannot benefit from it without retraining.

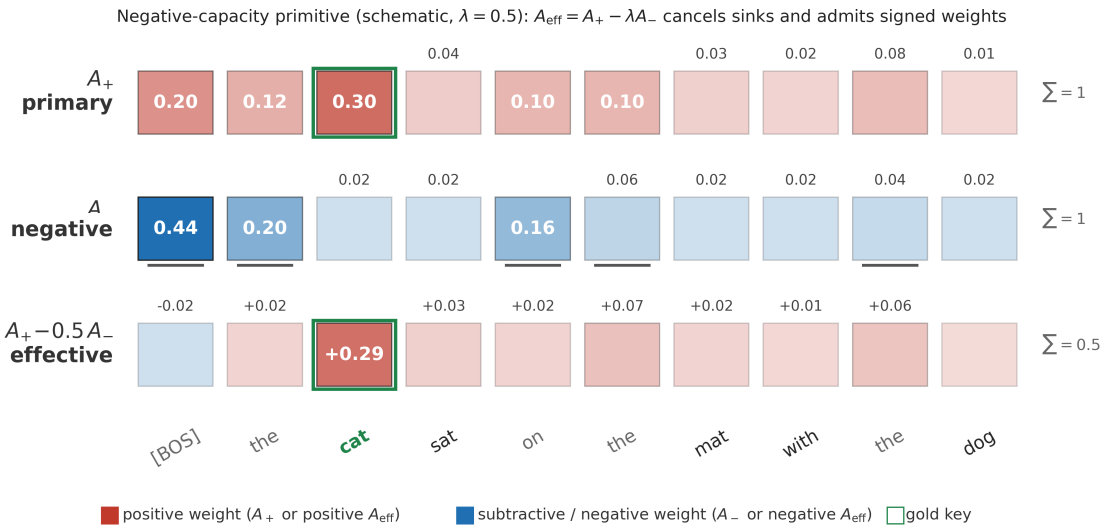

Figure 1: The negative-capacity primitive on a schematic prompt *"[BOS] the cat sat on the mat with the dog"*, with *cat* as the gold key (green box) and $\lambda = 0.5$. Each displayed value is the 2-decimal weight; every cell satisfies $A_{\text{eff}}[i] = A_+[i] - 0.5\,A_-[i]$ exactly. *Top* ($A_+ \in \Delta^N$, $\sum = 1$): the primary softmax map. Mass concentrates on the gold key, but a fine-tuned A_+ cannot avoid leaking probability onto the BOS token and the high-frequency function words (*the, on, the,* ...) underlined in the middle row—the structural attention-noise pattern documented by Xiao et al. (2024) and Gu et al. (2025). *Middle* ($A_- \in \Delta^N$, $\sum = 1$): the DAA negative-capacity branch (§3.1); cross-entropy gradients (Theorem 3.6) route A_- onto the sink positions without auxiliary supervision. *Bottom* ($A_+ - 0.5\,A_-$, $\sum = 0.5$): the effective attention. The subtraction cancels the sink-driven mass and, on [BOS], overshoots to a *negative* effective weight (-0.02)—an effective per-key coefficient that softmax attention alone cannot produce (Proposition 3.3). Because A_{eff} takes signed values it is no longer a probability distribution; the schematic is qualitative, and the empirical entropy reduction on real GPT-2 attention (2.01 → 0.88 nats) is reported separately in Figure 3 and §4.3.

2.3 DEX (Differential Extension)

DEX (Kong et al., 2025) avoids retraining by applying a learnable linear correction to the value-aggregated output:

$$\begin{aligned} A &= \mathrm{softmax}(QK^\top/\sqrt{d_k}), \\ O' &= AV\,(I - \lambda(t)\,W_{\text{DEX}}), \end{aligned} \tag{2}$$

where W_{DEX} is initialized to identity and $\lambda(t)$ anneals from a constant toward a learned value. Because the correction operates *after* softmax has produced a non-negative attention map, it cannot recover negative effective attention—the architectural primitive added by equation 1 is precisely the one DEX cannot supply. We make this information-theoretic obstruction precise as Proposition 3.3 in §3.3.

2.4 Positioning Against Related Lines of Work

Attention sinks and noise structure in pre-trained LMs. Xiao et al. (2024) documented that pre-trained Transformers accumulate disproportionate attention on a small number of "sink" tokens—typically BOS and high-frequency function words. Gu et al. (2025) traced when this sink behavior emerges during pre-training, and Sun et al. (2024) linked it to "massive activations" in the residual stream. The recurring finding is that these sinks are a *structural* property of softmax attention rather than a quirk of any specific checkpoint, and that any remediation must either avoid forming them at pre-training time or actively counteract them at fine-tuning time. DAA is a fine-tuning-time intervention for the latter regime: A_- concentrates on exactly the BOS / high-frequency positions Xiao et al. (2024); Gu et al. (2025) identified, and the subtraction $A_+ - \lambda A_-$ cancels the sink-driven mass at the source (§4.3).

Softmax-level alternatives. A complementary line of work attacks the same noise problem by *replacing* the softmax. Qiu et al. (2025) introduce a sigmoid gating that eliminates sinks; Zuhri et al. (2025) propose a rectified softmax ("Softpick") that drops the sum-to-one constraint. These approaches change the activation function and therefore require either pre-training from scratch or substantial re-training to integrate with existing checkpoints. DAA instead leaves the softmax untouched and

adds an architectural primitive on top of it, so any existing pre-trained model can be retrofitted with $<1\%$ extra parameters.

Differential attention. The Differential Transformer (Ye et al., 2025) bakes negative capacity into a pre-training-time architectural choice; DEX (Kong et al., 2025) adds a post-softmax linear correction at the output stream. DAA fills the gap between them: a fine-tuning-time intervention that, unlike DEX, lives *inside* the Q-K logit space and therefore can produce signed effective attention (Proposition 3.3).

Parameter-efficient fine-tuning (PEFT). LoRA (Hu et al., 2022), prefix tuning (Li and Liang, 2021), and adapters (Houlsby et al., 2019) all stay within the existing softmax-attention function class and leave $A \in \Delta^N$ non-negative. DAA is orthogonal at the expressivity level—adding a *signed*-attention primitive that no purely-positive parameterization of W_Q, W_K, W_V can realize—and could in principle be stacked on top of a LoRA adapter.

Position bias and long-context attention. Wu et al. (2025) and Liu et al. (2023) characterize a related "lost-in-the-middle" phenomenon, where attention degrades for middle-of-context tokens; Wu et al. (2025) unify this with attention sinks under a single graph-theoretic position-bias framework. DAA does not target position bias directly, but its noise-cancellation mechanism shares the same intuition—reshape attention logits without changing the model's signal pathway.

Missing-primitive precedents. Our framing of negative capacity as a *missing architectural primitive* has precedents: RoPE (Su et al., 2024) and ALiBi (Press et al., 2022) restore translation-equivariant positional information; FlashAttention (Dao et al., 2022) restores I/O-efficient softmax. All are expressivity or efficiency primitives absent from earlier architectures and added back at low cost (see Appendix D for an extended discussion).

3 Method

3.1 The Negative-Capacity Primitive

Let $A_+, A_- \in \Delta^N$ both be row-stochastic softmax outputs. The signed combination

$$A_{\text{eff}} \;:=\; A_+ - \lambda\, A_-, \qquad \lambda > 0, \tag{3}$$

takes values in $[-\lambda, 1] \subset \mathbb{R}$ and is the *minimal* construction equipping a softmax attention layer with signed effective attention. We call any layer-level realization of equation 3 a *negative-capacity injection*. The Differential Transformer equation 1 realizes it at pre-training time; in the fine-tuning regime the pre-trained W_Q, W_K, W_V are fixed, and the negative branch must be retrofitted via a small additional parameter W_-. We constrain A_+ and A_- to share the same Q and K projections, so the negative branch reuses the pre-trained token-embedding geometry.

A self-attention layer admits three positions for such a retrofit, each instantiated by a published adaptation family: the *output* position (DEX), where a learnable matrix multiplies AV *after* softmax—ruled out by Proposition 3.3 (§3.3); the *input* position (DiffQ / DiffK), where a $d_{\text{model}} \times d_{\text{model}}$ matrix is applied to X before W_Q/W_K—a strict over-parameterization of DAA by Theorem 3.1 (§3.2); and the *Q-K logit* position (DAA), where a $d_k \times d_k$ matrix is inserted *inside* the bilinear $QK^\top$ to produce a second logit matrix $QW_{\text{DAA}}K^\top$. Appendix A (Figure 4) gives a forward-pass diagram of the three positions.

3.2 K-Side Equivalence: DAA as Canonical Minimum

The lesson: DiffK and DAA reach the same family of attention logits; DAA reaches it with a strictly smaller, row-space-constrained set of parameters that matches the pre-trained W_K geometry.

Both DAA and the K-side input variant DiffK produce, for a single attention head, an $N \times N$ pre-softmax logit matrix of the form $QMX^\top$ for some $M \in \mathbb{R}^{d_k \times d_{\text{model}}}$; DAA reaches $M_{\text{DAA}} := W_{\text{DAA}}\, W_K^\top$ (from $QW_{\text{DAA}}K^\top$), while DiffK reaches $M_{\text{DiffK}} := W_K^\top W_{D_K}^\top$ (from $Q\,(XW_{D_K}W_K)^\top$). Theorem 3.1 below compares the two reachable sets.

Theorem 3.1 (K-side Equivalence and Strict Containment)**.** *Assume $W_K \in \mathbb{R}^{d_{\text{model}} \times d_k}$ has full column rank and $d_k < d_{\text{model}}$. Define $\mathcal{M}_{\text{DAA}} := \{W_{\text{DAA}}\, W_K^\top : W_{\text{DAA}} \in \mathbb{R}^{d_k \times d_k}\}$ and $\mathcal{M}_{\text{DiffK}} := \{W_K^\top W_{D_K}^\top : W_{D_K} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}\}$ as subsets of $\mathbb{R}^{d_k \times d_{\text{model}}}$, and let $\mathcal{F}_\bullet$ denote their images under the linear map $\Psi(M) = QMX^\top$. Then*

Clause 1 (parameter level, unconditional). *$\mathcal{M}_{\text{DAA}} \subsetneq \mathcal{M}_{\text{DiffK}}$, with codimension $d_k(d_{\text{model}} - d_k)$.*

Clause 2 (logit level, conditional). *If Ψ is injective on $\mathbb{R}^{d_k \times d_{\text{model}}}$ (e.g., when $\operatorname{rank}(W_Q) = d_k$, $N \geq d_{\text{model}}$, and X has full column rank d_{model}), then $\mathcal{F}_{\text{DAA}} \subsetneq \mathcal{F}_{\text{DiffK}}$.*

Inclusion $\mathcal{F}_{\text{DAA}} \subseteq \mathcal{F}_{\text{DiffK}}$ holds unconditionally; only strict $\mathcal{F}$-level containment needs the Clause 2 hypothesis.

Proof sketch. **(a) Inclusion.** For any W_{DAA}, set $W_{D_K}^\top := (W_K^\top)^+ W_{\text{DAA}} W_K^\top$ using the Moore–Penrose pseudoinverse; full row rank of $W_K^\top$ gives $W_K^\top (W_K^\top)^+ = I_{d_k}$, so $W_K^\top W_{D_K}^\top = W_{\text{DAA}} W_K^\top \in \mathcal{M}_{\text{DAA}}$. **(b) Strictness.** Dimension counting: $\dim \mathcal{M}_{\text{DAA}} = d_k^2$ vs. $\dim \mathcal{M}_{\text{DiffK}} = d_k\, d_{\text{model}}$. An explicit witness in the gap is $M^\star = e_1 (w^\perp)^\top$ for any $w^\perp \in \ker(W_K^\top) \setminus \{0\}$. Full proof, the Moore–Penrose construction details, the dimension/witness argument, and corrections of the hand-derived $(W_K^\top)^{-1}$ equivalence circulated by an earlier reviewer are deferred to Appendix B. □

Corollary 3.2 (DAA is a strict regularization of DiffK)**.** *Under the assumptions of Theorem 3.1, DAA implements DiffK* under a row-space constraint*: every row of the effective parameter M_{DAA} is restricted to $\operatorname{col}(W_K)$, the d_k-dimensional subspace already used by the pre-trained key projection. This is a head-shared, low-rank* regularization *that ties the differential branch to the geometry of the pre-trained model. The empirical superiority of DAA over DiffK in fine-tuning is therefore evidence that this inductive bias—not any structural difference between the two mechanisms—captures the correct hypothesis class for noise modeling at fine-tuning scale: DAA does not solve a problem DiffK cannot solve; it commits to a strictly smaller, better-conditioned hypothesis class in which the correct solution is easier to find from limited fine-tuning data.*

3.3 Pre-Softmax vs. Post-Softmax: A Non-Negativity Obstruction

We next rule out the output-position family. The softmax map $\sigma : \mathbb{R}^N \to \Delta^N$ is not injective—it identifies any pair of logit vectors s, s' satisfying $s' = s + c\,\mathbf{1}$ for $c \in \mathbb{R}$, since $\sigma(s) = \sigma(s + c\,\mathbf{1})$—and its image Δ^N is a measure-zero subset of $\mathbb{R}^N$. By the Data Processing Inequality applied to the deterministic Markov chain (hypothetical signed precursor) $\to$ (post-softmax map $A \in \Delta^N$) $\to$ (linear post-processing of AV) (Beaudry and Renner, 2011), any information about a hypothetical pre-softmax negative-attention structure that is not preserved by σ is unrecoverable from any deterministic post-processing of AV. The following Proposition gives the concrete operational consequence: post-softmax right-multiplication has effective per-key coefficients that remain in $[0, 1]$.

Proposition 3.3 (Post-softmax linear correction preserves non-negativity of effective attention)**.** *Let $A \in \Delta^N$ be a row-stochastic attention map produced by softmax, and let $V \in \mathbb{R}^{N \times d_v}$ be the value matrix. Any operator of the form*

$$T(O) \;=\; O\,M \qquad \text{with } M \in \mathbb{R}^{d_v \times d_v},$$

applied to $O = AV$ produces an output $T(AV) = (AV)M$ whose effective per-key attention coefficients $\widetilde{A}^{ij}$ (defined by $T(AV)_i = \sum_j \widetilde{A}^{ij}\,(v_j M)$) are equal to A^{ij}, hence non-negative. In particular, no choice of M can produce an effective $\widetilde{A}^{ij} < 0$.

Proof. Expanding the definition,

$$T(AV)_i \;=\; \Big(\sum_j A^{ij}\, v_j\Big) M \;=\; \sum_j A^{ij}\,(v_j M),$$

so the coefficient on the transformed value vector $v_j M$ is exactly $A^{ij} \in [0, 1]$. A negative effective coefficient would require $A^{ij} < 0$, contradicting $A \in \Delta^N$. □

Corollary 3.4 (DEX cannot express negative effective attention)**.** *DEX (eq. equation 2) is the instance $T(O) = O\,(I - \lambda(t)\,W_{\text{DEX}})$ of Proposition 3.3; its effective per-key attention coefficients on the value vectors are therefore equal to $A^{ij} \geq 0$ for every $\lambda(t)$ and W_{DEX}. The differential mechanism of equation 1, by contrast, combines two softmax outputs* before *value aggregation, and the resulting signed effective coefficients lie outside the image of the post-softmax-linear map and are therefore architecturally inaccessible to DEX. DAA (§3.5) inherits this pre-softmax mixing while reusing the pre-trained Q, K matrices.*

3.4 The Gradient Incentive

The lesson: Cross-entropy training automatically routes the negative branch onto distractor tokens; no auxiliary loss, no manual supervision, no regularizer is required. The mechanism is a sign opposition in the gradient itself.

Setup. At a single attention head and query position i with annealed coefficient $\lambda > 0$, the DAA

forward pass is

$$o_i \;=\; \sum_{j=1}^{N} \big(A_+^{ij} - \lambda\, A_-^{ij}\big)\, v_j \;\in\; \mathbb{R}^{d_v}. \qquad (4)$$

We declare a key position j to be a *signal token* (resp. *distractor token*) for query i if attending more to j would decrease (resp. increase) the loss; the labels are read off the gradient $\nabla_{o_i} L$ directly, with no oracle supervision:

Definition 3.5 (Signal and distractor tokens)**.** Key position j is a *signal token for query* i when $\langle -\nabla_{o_i} L,\, v_j\rangle > 0$, and a *distractor token* when $\langle -\nabla_{o_i} L,\, v_j\rangle < 0$. The borderline case is excluded by generic position.

Theorem 3.6 (Gradient Incentive for Distractor Subtraction)**.** *Let L be any differentiable scalar loss computed from o_i in equation 17, and let $\lambda > 0$ be fixed. Then for every query position i and key position j,*

$$\frac{\partial L}{\partial A_+^{ij}} = \langle \nabla_{o_i} L,\, v_j\rangle, \qquad \frac{\partial L}{\partial A_-^{ij}} = -\lambda\, \langle \nabla_{o_i} L,\, v_j\rangle, \qquad (5)$$

so $\partial L/\partial A_-^{ij}$ and $\partial L/\partial A_+^{ij}$ have opposite signs *whenever $\langle \nabla_{o_i} L, v_j\rangle \neq 0$. Consequently, an unconstrained gradient-descent step in the ambient attention-entry coordinates increases A_+^{ij} on signal tokens and increases A_-^{ij} on distractor tokens. The two branches receive* complementary *ambient incentives from the chain rule, with no auxiliary loss.*

Proof sketch. For each pair (i, j), $\partial o_i/\partial A_+^{ij} = v_j$ and $\partial o_i/\partial A_-^{ij} = -\lambda v_j$ from the linear forward equation 17. Composing with $\nabla_{o_i} L$ via the chain rule gives equation 5. Propagation through the row-wise softmax is governed by the PSD Jacobian $J(p) = \mathrm{diag}(p) - pp^\top$ and preserves the global $-\lambda < 0$ scalar opposition but not per-coordinate antiparallelism; full statement and softmax-Jacobian discussion are deferred to Appendix C. □

Proposition 3.7 (Reduction to standard fine-tuning at $\lambda = 0$)**.** *When $\lambda = 0$, $o_i = \sum_j A_+^{ij} v_j$, the negative branch disappears from the forward pass, and $\partial L/\partial A_-^{ij} \equiv 0$. DAA is then functionally identical to standard fine-tuning of the attention layer; the gap between the "+FT" and "+DAA" rows of our experimental tables is therefore exactly the empirical contribution of the negative-capacity branch (§**??**).*

Two empirical predictions. Theorem 3.6 predicts (P1) DAA and standard fine-tuning have nearly indistinguishable training-loss curves near initialization—both place the same incentive on A_+—while (P2) downstream performance on noise-bottleneck tasks diverges because the negative branch A_- silently absorbs distractor mass without affecting cross-entropy on signal tokens. Empirically we observe exactly this pattern (Figure 2; Table 1). Detailed scoping and the parameter-level coupling discussion appear in Appendix C.

3.5 Differential Attention Adaption: Formal Definition

DAA introduces one learnable matrix $W_{\mathrm{DAA}} \in \mathbb{R}^{d_k \times d_k}$ per head, initialized at identity, and replaces $O = AV$ with

$$\begin{aligned} A_+ &= \mathrm{softmax}(QK^\top/\sqrt{d_k}), \\ A_- &= \mathrm{softmax}(QW_{\mathrm{DAA}}K^\top/\sqrt{d_k}), \\ O' &= \big(A_+ - \lambda(t)\, A_-\big)\, V. \end{aligned} \qquad (6)$$

The mixing coefficient $\lambda(t) = (1-\alpha)(t/T)\,\lambda_{\mathrm{init}} + \alpha\,\lambda_{\mathrm{learn}}$ with $\alpha = \min(1, t/T)$ is the annealing schedule of Kong et al. (2025). At $t = 0$ both branches coincide so DAA reduces to standard FT (Proposition 3.7); as $\lambda(t)$ anneals up, the gradient of Theorem 3.6 pushes W_{DAA} off identity toward distractor patterns. The added cost is d_k^2 per head, head-shared, giving d_{model}^2/h extra parameters per layer ($<1\%$ of base; Appendix G). Algorithm 1 in Appendix F states the forward pass procedurally.

4 Experiments and Analysis

We evaluate DAA against three families of baselines: standard fine-tuning (FT), the post-softmax linear correction DEX (Kong et al., 2025), and the input-differentiation variants DiffQ / DiffK / DiffV formalized in Appendix I. Rather than claim that DAA is best on average, we make a sharper claim: its largest gains appear where attention noise is the bottleneck, and elsewhere it preserves general capability. This pattern is what Theorem 3.6 and Proposition 3.3 predict.

Setup. Three pre-trained models—GPT-2 (117M) (Radford et al., 2019), Llama-3.2-1B, and Llama-3.1-8B (Dubey et al., 2024)—are fine-tuned with each adaptation method. GPT-2 uses a subset of OpenWebText (Peterson et al., 2019); Llama models use a subset of the Tulu-3 SFT mixture (OLMo et al., 2024). Only the

attention-layer parameters are updated; all other weights are frozen. Detailed hyperparameters are in Appendix K.

4.1 Noise-Bottleneck Tasks: Where DAA Shines

We first isolate tasks whose performance is known to be limited by attention noise rather than by representation quality: needle-in-a-haystack retrieval (NR) and LAMBADA last-word prediction. On these, the differential between methods is dominated by their ability to suppress distractor attention.

Figure 2: Training loss curves on the OpenWebText fine-tuning subset. As predicted by Theorem 3.6, DAA and standard fine-tuning produce nearly indistinguishable training-loss trajectories: the positive branch A_+ in DAA receives the same gradient incentive as the lone attention map in standard FT (both pursue signal tokens), while the negative branch A_- silently absorbs distractor mass without altering the cross-entropy target. Convergence-rate similarity is therefore evidence *for* the theorem, not against the method—the downstream performance gap (Tables 1–2) measures what the loss curve cannot see.

Table 1 reports perplexity on three corpora (OpenWebText, WikiText-2 (Kwiatkowski et al., 2019), LAMBADA (Grave et al., 2016)) and accuracy on LAMBADA and needle retrieval. On the noise-sensitive benchmarks DAA leads by a clear margin: needle-retrieval accuracy reaches 89.5 (+4.6 over Base, +2.1 over DEX, +5.1 over DiffQ), and LAMBADA accuracy reaches 48.9, the highest among the methods tested. Standard fine-tuning improves general-text perplexity but does not move noise-sensitive accuracy (NR 85.3 vs. Base 84.9), consistent with the view that minimizing next-token cross-entropy alone cannot reduce the attention-noise floor: lowering that floor requires negative capacity, which standard FT does not have.

The training-loss trajectory (Figure 2) supports

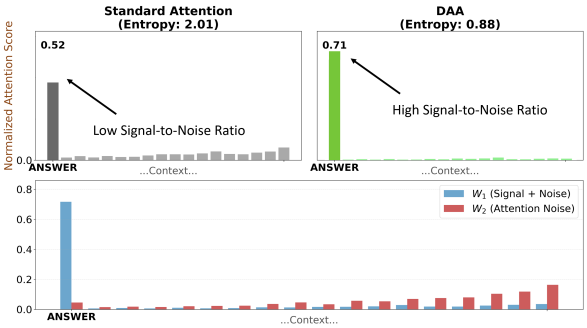

Figure 3: Decomposition of DAA's effective attention into A_+ and A_- on a representative GPT-2 layer (layer 2, head 3). *Top*: standard FT is diffuse ($H \approx 2.01$ nats); DAA's effective attention concentrates sharply (≈ 0.88 nats), a 56% reduction. *Bottom*: A_+ (W_1) peaks on the gold token; A_- (W_2) concentrates on the attention sinks of Xiao et al. (2024). The subtraction cancels the sinks at the source, as the primitive (§3.1) and Theorem 3.6 predict.

Theorem 3.6: DAA and standard FT track each other closely on cross-entropy, because the positive branch A_+ in DAA receives the same signal-token incentive as the lone attention map in standard FT, while A_- absorbs distractor mass without altering the loss. The downstream NR/LAMBADA gap is therefore the part of the picture the loss curve does not capture.

4.2 General Reasoning: Preservation, Not Regression

A natural objection is that suppressing attention to some tokens may degrade tasks where those tokens carry signal. We test this on eight reasoning benchmarks (Clark et al., 2018; Wang et al., 2019; Mihaylov et al., 2018; Bisk et al., 2020; Sakaguchi et al., 2021) for Llama-3.2-1B and Llama-3.1-8B (Table 2). Standard FT regresses on average (−1.77 on 1B, +0.18 on 8B), consistent both with catastrophic forgetting (Kirkpatrick et al., 2017) and with the framing in §2.1: minimizing cross-entropy on a small corpus can only redistribute attention inside the simplex, which can hurt general capability. The input-differentiation variants also regress on 8B (−0.58 DiffQ, −0.48 DiffK); Theorem 3.1 only places them in a strictly larger hypothesis class, but this is consistent with higher variance under matched data budgets (multi-seed runs would be needed for a formal claim). DAA improves the average and has no category collapse: +1.06 on 1B, +0.74 on 8B. On 8B the DAA-vs-DEX gap is small (+0.74 vs. +0.57; see Limitation 4 in §5); we therefore read DAA on this scale as *matching*

Table 1: Comparison of models before and after fine-tuning with different attention architectures. FT represents the standard fine-tuning method. PPL stands for Perplexity, lower is better. ACC stands for Accuracy, higher is better. Bold values indicate the best performance in that column. WKT2 represents the WikiText2 corpus, NR represents the needle retrieval.

Model	**OWT (PPL)**	**OWT(new) (PPL)**	**WKT2 (PPL)**	**LAMBADA (PPL)**	**LAMBADA (ACC)**	**NR (ACC)**
GPT-2 (117M)						
Base	33.3	32.4	24.7	42.98	47.3	84.9
+ FT	29.0	**28.1**	25.6	43.1	48.1	85.3
+ DEX	**28.9**	28.2	25.2	42.95	48.2	87.4
+ DiffQ (Ours)	29.2	28.4	**24.4**	43.55	48.0	84.4
+ DiffK (Ours)	29.3	28.5	**24.4**	43.55	48.1	88.3
+ DAA (Ours)	**28.9**	**28.1**	24.7	**42.25**	**48.9**	**89.5**

Table 2: Llama-3.2-1B and Llama-3.1-8B on 8 commonsense reasoning benchmarks: Base, standard FT, DEX, DAA, and the input-differentiation variants DiffQ/DiffK (DiffV underperforms at every scale, see Appendix I, and is omitted). Green = gain over Base, Gray = drop; Δ is the average gain over Base.

Model	**Arc-E**	**Arc-C**	**BoolQ**	**Hellaswag**	**OBQA**	**WIC**	**Winogrande**	**WSC**	**AVG**	Δ
Llama-3.2-1B										
Base	62.08	36.3	61.8	63.6	34.6	48.6	56.49	36.5	51.31	-
+ FT	65.08	33.56	57	64.13	30.8	50.1	56.2	39.42	49.54	-1.77
+ DEX	65.86	34.92	60.24	64.16	31.6	48.43	56.75	53.65	51.93	+0.62
+ DiffQ (Ours)	65.26	35.25	58.97	64.06	31.6	48.8	54.2	55.29	51.68	+0.37
+ DiffK (Ours)	66.31	35.59	62.51	64.11	29.4	48.8	49.3	55.72	50.1	+0.37
+ DAA (Ours)	65.96	33.22	62.78	64.29	33.4	49.9	56.51	52.88	**52.37**	**+1.06**
Llama-3.1-8B										
Base	78.9	52.6	74.9	78	42.1	51.9	73.1	58.6	63.76	-
+ FT	76.01	52.2	74.39	80.5	43.2	52.8	73.46	58.94	63.94	+0.18
+ DEX	77.2	52.66	78.4	78.6	42.3	52.5	73.9	59.1	64.33	+0.57
+ DiffQ (Ours)	78.6	51.39	73.21	79.3	43.8	52	70.62	56.54	63.18	-0.58
+ DiffK (Ours)	75.49	51.5	74.83	78.6	44.4	52.2	70.54	58.65	63.28	-0.48
+ DAA (Ours)	76.9	52.71	79.04	77.4	42.8	53.3	74.25	59.62	**64.50**	**+0.74**

DEX's general reasoning, with the discriminating evidence coming from the noise-bottleneck tasks of §4.1.

4.3 Where Does the Negative Branch Attend?

Figure 3 decomposes a DAA attention computation into its two softmax components A_+ and A_- on a representative GPT-2 layer (layer 2, head 3). The top row shows that the standard-FT attention distribution is diffuse (entropy ≈ 2.01 nats), while DAA's effective attention concentrates sharply (entropy ≈ 0.88 nats)—a 56% reduction. The bottom row breaks DAA into its two softmax maps: A_+ peaks on the gold context token, while A_- concentrates on the BOS sink and high-frequency function tokens—the attention sinks of Xiao et al. (2024). This is the distractor pattern Theorem 3.6 predicts the gradient will steer A_- toward, and the subtraction $A_+ - \lambda A_-$ cancels the sink-driven noise that vanilla softmax attention cannot remove (§3.1).

At $\lambda = 0$ DAA collapses to standard FT (Proposition 3.7), so the "+FT" vs "+DAA" gap in Tables 1 and 2 isolates the contribution of the negative branch under matched optimization: $+4.2$ NR, $+0.8$ LAMBADA, $+2.83/+0.56$ on 1B/8B reasoning. Qualitative failure modes are in Appendix E.

5 Conclusion

We identified *negative attention capacity*—the ability to assign signed effective weights—as the primitive that pre-trained softmax attention is missing, and installed it via DAA: one $d_k \times d_k$ matrix at the Q-K logit position, $< 1\%$ extra parameters, retrofittable to any pre-trained Transformer. Gains concentrate on noise-bottleneck tasks and vanish elsewhere, the pattern Theorems 3.1–3.6 and Proposition 3.3 predict. *Pre-trained language models do not need bigger attention; they need a negative one.*

Limitations

Five limitations bound the scope of our claims and motivate future work.

- *Inference overhead.* The negative-capacity branch computes a second softmax map $A_- = \mathrm{softmax}(QW_{\mathrm{DAA}}K^\top/\sqrt{d_k})$, adding a quadratic term to the attention FLOPs (Appendix H): the dominant $N^2 d_{\mathrm{model}}$ coefficient grows from 2 to 3, while the linear projection term remains $(4 + 1/h)\, N d_{\mathrm{model}}^2$. We see this as the inherent cost of adding negative capacity rather than a flaw of the specific parameterization, but it does limit applicability to latency-sensitive long-context inference. Sparse-attention approximations of A_- are a natural extension.

- *Scale ceiling.* Our evaluation reaches Llama-3.1-8B. Behavior on the $\geq$ 70B regime—where the relationship between attention noise and downstream errors may shift—remains open.

- *Long-context benchmarks beyond NIAH.* The needle-retrieval result on GPT-2 is suggestive of long-context benefits, but a full evaluation on RULER (Hsieh et al., 2024), LongBench (Bai et al., 2024), or multi-needle settings is outside the scope of this paper.

- *PEFT baselines.* Our empirical comparisons focus on the differential-attention family (FT, DEX, DiffQ / DiffK / DiffV) because these methods target the same underlying mechanism (attention noise at the Q-K logit position). A head-to-head comparison with LoRA (Hu et al., 2022), prefix tuning (Li and Liang, 2021), and adapter-based PEFT (Houlsby et al., 2019) would help quantify how DAA's signed-attention primitive complements parameter-space PEFT; we leave this comparison and a DAA $\circ$ LoRA composition study to future work.

- *Single-seed reporting.* All accuracies and perplexities come from single-seed runs. The consistency of gains across three model scales (GPT-2 / 1B / 8B) is indirect robustness evidence, but formal significance testing requires multi-seed runs that we leave to future work.

Ethical Considerations

This work proposes an architectural modification to attention layers in pre-trained language models. It uses publicly available pre-trained checkpoints (GPT-2, Llama-3.2-1B, Llama-3.1-8B) and publicly released fine-tuning corpora (OpenWebText, Tulu-3 SFT mixture); no new data are collected. Negative-capacity injection reduces attention to certain tokens by design, which could in principle interact with mitigation strategies for harmful content; we expect this interaction to be neutral or mildly beneficial (sharper attention on signal tokens) but have not measured it directly, and we flag this as worth study. We see no immediate dual-use, privacy, or fairness concerns specific to the method beyond those already inherent to fine-tuning large language models.

Use of Large Language Models

In preparing this manuscript, the authors used large language models (LLMs) to assist with proofreading, grammatical correction, and stylistic refinement of the English prose. All research questions, theoretical claims, proofs, experimental designs, code, and analyses were conceived, implemented, and verified by the human authors. No LLM was used to generate experimental results, theorem proofs, citation choices, or any factual content of the paper. LLM-assisted edits were reviewed by the authors before inclusion.

References

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, and 1 others. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Normand J Beaudry and Renato Renner. 2011. An intuitive proof of the data processing inequality. *arXiv preprint arXiv:1107.0740*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Tri Dao, Daniel Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Edouard Grave, Armand Joulin, and Nicolas Usunier. 2016. Improving neural language models with a continuous cache. In *International Conference on Learning Representations*.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Wang, and Min Lin. 2025. When attention sink emerges in language models: An empirical view. In *International Conference on Learning Representations (ICLR)*. ArXiv:2410.10781.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning (ICML)*.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. 2024. RULER: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew E. Peters, Pradeep Dasigi, Jemix Gu, David Atkinson, Aleksandra Piktus, Shmuel Amar, Oyvind Tafjord, Sam Skjonsberg, Haritz Puerto, Iz Beltagy, Hanna Hajishirzi, Noah A. Smith, Yejin Choi, and Luke Zettlemoyer. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *Preprint*, arXiv:2311.10702.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Chaerin Kong, Jiho Jang, and Nojun Kwak. 2025. Understanding differential transformer unchains pretrained self-attentions. *arXiv preprint arXiv:2505.16333*.

T. Kwiatkowski, J. Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, D. Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. In *Transactions of the Association for Computational Linguistics*.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*. AAAI Press.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, F. Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *ArXiv*, abs/2104.08786.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 21 others. 2024. 2 olmo 2 furious. *ArXiv*, abs/2501.00656.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. 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)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc,

E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Nicolas Penedo and 1 others. 2022. Safe and sound: The safetensors format. https://github.com/huggingface/safetensors.

Joshua Peterson, Stephan Meylan, and David Bourgin. 2019. Open clone of openai's unreleased webtext dataset scraper.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations (ICLR)*.

Zihan Qiu, Zekun Wang, Bo Zheng, and 1 others. 2025. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *arXiv preprint arXiv:2505.06708*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. 2024. Massive activations in large language models. In *Conference on Language Modeling (COLM)*. ArXiv:2402.17762.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Xinyi Wu, Yifei Wang, and Stefanie Jegelka. 2025. On the emergence of position bias in transformers. *arXiv preprint arXiv:2502.01951*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. 2025. Differential transformer. In *The Thirteenth International Conference on Learning Representations*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *Preprint*, arXiv:1905.07830.

Zayd M. K. Zuhri, Erland Hilman Fuadi, and Alham Fikri Aji. 2025. Softpick: No attention sink, no massive activations with rectified softmax. *arXiv preprint arXiv:2504.20966*.

A Architectural Comparison

Figure 4 sketches the three positions in a self-attention layer at which a negative-capacity branch can be installed—input (DIFF Transformer), output (DEX), and Q-K logit (DAA)—and Table 3 summarizes the resulting trade-off between noise-cancellation effectiveness, retrofittability to pre-trained checkpoints, and parameter cost.

B Full Proof of Theorem 3.1 (K-side Equivalence)

This appendix contains the full proof of Theorem 3.1, including the Moore–Penrose pseudoinverse construction for inclusion (Part a), the dimension counting and explicit witness for strict containment (Part b), the conditional $\mathcal{F}$-level argument (Part c), the milder sufficient condition for strict $\mathcal{F}$-containment (Remark B.1), the explicit correction of an earlier hand-derived equivalence claim that used a two-sided inverse where only a pseudoinverse exists (Remark B.2), and a discussion of the mildness of the full-column-rank assumption on W_K (Remark B.3).

Logit families. Both DAA and the input-side DiffK variant of differential attention produce, for a single attention head, an $N \times N$ *pre-softmax logit matrix* of the form $QMX^\top$ for some $M \in \mathbb{R}^{d_k \times d_{\text{model}}}$. The difference lies in *which* matrices M each parameterization can reach. Indeed, DAA produces

$$\begin{aligned} Q\, W_{\text{DAA}}\, K^\top &= Q\, W_{\text{DAA}}\, (XW_K)^\top \\ &= Q\, \underbrace{W_{\text{DAA}} W_K^\top}_{=:\, M_{\text{DAA}}}\, X^\top, \end{aligned}$$

while DiffK produces

$$\begin{aligned} Q\, (XW_{D_K} W_K)^\top &= Q\, W_K^\top W_{D_K}^\top X^\top \\ &= Q\, \underbrace{W_K^\top W_{D_K}^\top}_{=:\, M_{\text{DiffK}}}\, X^\top. \end{aligned}$$

We therefore compare the families of reachable M-matrices directly:

$$\begin{aligned} \mathcal{M}_{\text{DAA}} &:= \left\{ W_{\text{DAA}} W_K^\top \,:\, W_{\text{DAA}} \in \mathbb{R}^{d_k \times d_k} \right\} \\ &\subseteq \mathbb{R}^{d_k \times d_{\text{model}}}, \qquad (7) \\ \mathcal{M}_{\text{DiffK}} &:= \left\{ W_K^\top W_{D_K}^\top \,:\, W_{D_K} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}} \right\} \\ &\subseteq \mathbb{R}^{d_k \times d_{\text{model}}}. \qquad (8) \end{aligned}$$

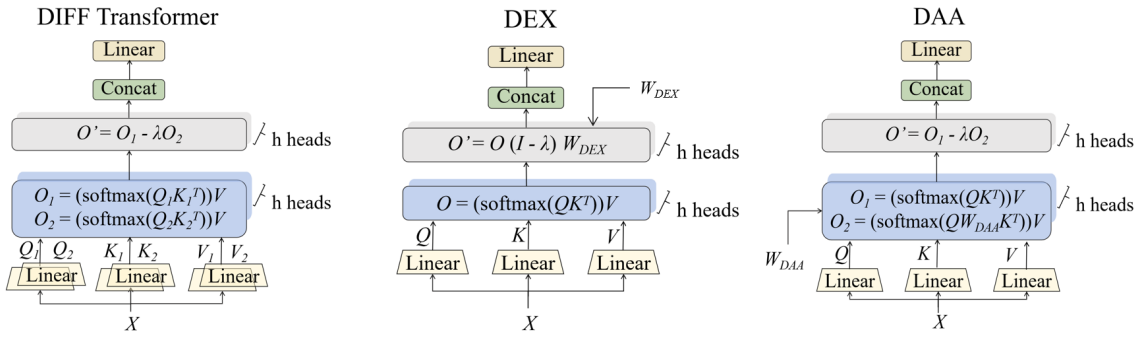

Figure 4: Where to inject a negative-capacity branch into self-attention. *Left*, DIFF Transformer (Ye et al., 2025): at the *input* position, two parallel Q, K, V streams are computed from X and their attention maps subtracted ($O' = O_1 - \lambda O_2$); a pre-training-time architectural choice. *Middle*, DEX (Kong et al., 2025): at the *output* position, a learnable correction W_{DEX} multiplies $O = (\mathrm{softmax}\, QK^\top)V$; ruled out as a source of signed effective attention by Proposition 3.3. *Right*, DAA (this work): at the *Q-K logit* position, a $d_k \times d_k$ matrix W_{DAA} is inserted *inside* the bilinear product to produce a second softmax map $A_- = \mathrm{softmax}(QW_{\mathrm{DAA}}K^\top)$, which is subtracted from A_+.

Table 3: Comparison of differential attention architectures, where d_{model} is the model's hidden-state dimension and h is the number of heads.

Architecture	**Reduce Attention Noise**	**Introduced to Existing Transformer Models**	**# Parameters of Each Attention Layer (h heads)**
DIFF Transformer	✓	X	$7d_{\text{model}}^2$
DEX	X	✓	$\boldsymbol{(4 + \frac{1}{h})d_{\text{model}}^2}$
DiffK (Ours)	✓	✓	$5d_{\text{model}}^2$
DiffQ (Ours)	✓	✓	$5d_{\text{model}}^2$
DiffV (Ours)	X	✓	$5d_{\text{model}}^2$
DAA (Ours)	✓	✓	$\boldsymbol{(4 + \frac{1}{h})d_{\text{model}}^2}$

For any fixed X, W_Q, W_K, the corresponding logit families $\mathcal{F}_{\mathrm{DAA}} = \{QMX^\top : M \in \mathcal{M}_{\mathrm{DAA}}\}$ and $\mathcal{F}_{\mathrm{DiffK}} = \{QMX^\top : M \in \mathcal{M}_{\mathrm{DiffK}}\}$ are the images of $\mathcal{M}_{\mathrm{DAA}}$ and $\mathcal{M}_{\mathrm{DiffK}}$ under the common linear map $\Psi\colon M \mapsto QMX^\top$. Inclusion at the $\mathcal{M}$-level therefore transfers *verbatim* to inclusion at the $\mathcal{F}$-level. Strict containment, by contrast, transfers only when Ψ does not collapse the extra directions in $\mathcal{M}_{\mathrm{DiffK}} \setminus \mathcal{M}_{\mathrm{DAA}}$ to zero; this requires a separate hypothesis, which we make explicit in Clause 2 below. The degenerate case $Q = 0$ (e.g., $W_Q = 0$ or $X = 0$), which forces $\mathcal{F}_{\mathrm{DAA}} = \mathcal{F}_{\mathrm{DiffK}} = \{0\}$, illustrates why such a hypothesis is necessary.

Proof. The proof has three parts. Part **(a)** establishes inclusion $\mathcal{M}_{\mathrm{DAA}} \subseteq \mathcal{M}_{\mathrm{DiffK}}$ (and hence $\mathcal{F}_{\mathrm{DAA}} \subseteq \mathcal{F}_{\mathrm{DiffK}}$) via an explicit pseudoinverse construction. Part **(b)** establishes Clause 1 (strict $\mathcal{M}$-level containment) via dimension counting and an explicit witness in $\mathcal{M}_{\mathrm{DiffK}} \setminus \mathcal{M}_{\mathrm{DAA}}$. Part **(c)** transfers strictness to the $\mathcal{F}$-level under the injectivity hypothesis, establishing Clause 2.

Throughout, let $B := W_K^\top \in \mathbb{R}^{d_k \times d_{\text{model}}}$. By assumption $\mathrm{rank}(B) = d_k$, so B has full *row* rank. The Moore–Penrose pseudoinverse $B^+ \in \mathbb{R}^{d_{\text{model}} \times d_k}$ of such a B satisfies

$$B\,B^+ \;=\; I_{d_k}, \tag{9}$$

because $BB^\top \in \mathbb{R}^{d_k \times d_k}$ is invertible (its $\mathrm{rank}(BB^\top) = \mathrm{rank}(B) = d_k$) and $B^+ = B^\top(BB^\top)^{-1}$ is a right inverse of B. We will use only property equation 9 below.

Part (a): $\mathcal{M}_{\mathrm{DAA}} \subseteq \mathcal{M}_{\mathrm{DiffK}}$**.** Let $W_{\mathrm{DAA}} \in \mathbb{R}^{d_k \times d_k}$ be arbitrary, and set $M := W_{\mathrm{DAA}} W_K^\top = W_{\mathrm{DAA}} B \in \mathcal{M}_{\mathrm{DAA}}$. Define

$$\begin{aligned} W_{D_K}^\top &:= B^+\, W_{\mathrm{DAA}}\, B \\ &= (W_K^\top)^+\, W_{\mathrm{DAA}}\, W_K^\top \\ &\in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}. \end{aligned} \tag{10}$$

(Equivalently, $W_{D_K} := B^\top W_{\mathrm{DAA}}^\top (B^+)^\top$.) Then

$$\begin{aligned} W_K^\top W_{D_K}^\top &= B\, W_{D_K}^\top = B\left(B^+ W_{\mathrm{DAA}}\, B\right) \\ &= \left(BB^+\right) W_{\mathrm{DAA}}\, B \\ &\overset{\text{equation 9}}{=} I_{d_k}\, W_{\mathrm{DAA}}\, B \\ &= W_{\mathrm{DAA}}\, W_K^\top = M. \end{aligned}$$

Hence $M \in \mathcal{M}_{\mathrm{DiffK}}$. Since W_{DAA} was arbitrary, $\mathcal{M}_{\mathrm{DAA}} \subseteq \mathcal{M}_{\mathrm{DiffK}}$.

Translating to logits, for any W_{DAA} the choice equation 10 gives $Q\, W_{\mathrm{DAA}}\, K^\top = Q\,(X W_{D_K} W_K)^\top$, so $\mathcal{F}_{\mathrm{DAA}} \subseteq \mathcal{F}_{\mathrm{DiffK}}$.

Part (b): Strict containment. We argue both by dimension counting and by exhibiting an explicit witness.

(b.1) Geometric characterizations. We first identify each family as a linear subspace of $\mathbb{R}^{d_k \times d_{\mathrm{model}}}$.

- The map $\Phi_{\mathrm{DAA}} \colon \mathbb{R}^{d_k \times d_k} \to \mathbb{R}^{d_k \times d_{\mathrm{model}}}$, $\Phi_{\mathrm{DAA}}(W) = WB$, is linear with image

$$\mathcal{M}_{\mathrm{DAA}} = \big\{ M \in \mathbb{R}^{d_k \times d_{\mathrm{model}}} : \text{every row of } M \text{ lies in } \mathrm{row}(B) \big\},$$

  because the r-th row of WB equals $W_{r,:}\, B$, a linear combination of the rows of B. Equivalently, $\mathcal{M}_{\mathrm{DAA}} = \mathbb{R}^{d_k} \otimes \mathrm{row}(B)$, where $\mathrm{row}(B) = \mathrm{col}(W_K) \subseteq \mathbb{R}^{d_{\mathrm{model}}}$ is the d_k-dimensional column space of W_K. Since Φ_{DAA} is injective (if $WB = 0$ then $WBB^\top = 0$, and $BB^\top$ is invertible, hence $W = 0$),

$$\dim \mathcal{M}_{\mathrm{DAA}} \;=\; d_k \cdot d_k \;=\; d_k^2. \tag{11}$$

- The map $\Phi_{\mathrm{DiffK}} \colon \mathbb{R}^{d_{\mathrm{model}} \times d_{\mathrm{model}}} \to \mathbb{R}^{d_k \times d_{\mathrm{model}}}$, $\Phi_{\mathrm{DiffK}}(W) = BW^\top$, is linear, and we now show $\mathcal{M}_{\mathrm{DiffK}} = \mathbb{R}^{d_k \times d_{\mathrm{model}}}$ in full. Indeed, given any $M \in \mathbb{R}^{d_k \times d_{\mathrm{model}}}$, set $W_{D_K}^\top := B^+ M \in \mathbb{R}^{d_{\mathrm{model}} \times d_{\mathrm{model}}}$; then $BW_{D_K}^\top = (BB^+)M = I_{d_k} M = M$ by equation 9. Therefore

$$\begin{aligned} \dim \mathcal{M}_{\mathrm{DiffK}} &= \dim \mathbb{R}^{d_k \times d_{\mathrm{model}}} \\ &= d_k \cdot d_{\mathrm{model}}. \end{aligned} \tag{12}$$

(b.2) Dimension gap. Combining equation 11–equation 12 with the assumption $d_k < d_{\mathrm{model}}$,

$$\begin{aligned} &\dim \mathcal{M}_{\mathrm{DiffK}} - \dim \mathcal{M}_{\mathrm{DAA}} \\ &\quad = d_k(d_{\mathrm{model}} - d_k) \;>\; 0, \end{aligned} \tag{13}$$

so $\mathcal{M}_{\mathrm{DAA}}$ is a *proper* linear subspace of $\mathcal{M}_{\mathrm{DiffK}}$.

(b.3) Explicit witness. We exhibit a specific matrix $M^\star \in \mathcal{M}_{\mathrm{DiffK}} \setminus \mathcal{M}_{\mathrm{DAA}}$. Since $d_k < d_{\mathrm{model}}$, the column space $\mathrm{col}(W_K)$ is a proper d_k-dimensional subspace of $\mathbb{R}^{d_{\mathrm{model}}}$, hence its orthogonal complement is nontrivial. Pick any

$$w^\perp \in \mathbb{R}^{d_{\mathrm{model}}} \quad \text{with } w^\perp \neq 0 \text{ and } W_K^\top w^\perp = 0, \tag{14}$$

which exists because $\dim \ker(W_K^\top) = d_{\mathrm{model}} - d_k \geq 1$ by rank–nullity. Define $M^\star \in \mathbb{R}^{d_k \times d_{\mathrm{model}}}$ to be the matrix whose first row equals $(w^\perp)^\top$ and whose remaining rows are zero, i.e.

$$M^\star \;:=\; e_1\,(w^\perp)^\top,$$

where $e_1 \in \mathbb{R}^{d_k}$ is the first standard basis vector.

(i) $M^\star \in \mathcal{M}_{\mathrm{DiffK}}$. By part (a) of the geometric characterization, $\mathcal{M}_{\mathrm{DiffK}}$ is all of $\mathbb{R}^{d_k \times d_{\mathrm{model}}}$; explicitly, take $W_{D_K}^\top := B^+ M^\star$, which gives $W_K^\top W_{D_K}^\top = BB^+ M^\star = M^\star$.

(ii) $M^\star \notin \mathcal{M}_{\mathrm{DAA}}$. Suppose, for contradiction, that $M^\star = W\, W_K^\top$ for some $W \in \mathbb{R}^{d_k \times d_k}$. Taking the first row of both sides (a row in $\mathbb{R}^{1 \times d_{\mathrm{model}}}$),

$$(w^\perp)^\top \;=\; M^\star_{1,:} \;=\; W_{1,:}\, W_K^\top,$$

where $W_{1,:} \in \mathbb{R}^{1 \times d_k}$ is the first row of W. Transposing both sides (to work with column vectors in $\mathbb{R}^{d_{\mathrm{model}}}$),

$$w^\perp \;=\; \left(W_{1,:}\, W_K^\top\right)^\top \;=\; W_K\, W_{1,:}^\top,$$

so $w^\perp$ is a linear combination of the columns of W_K, i.e. $w^\perp \in \mathrm{col}(W_K)$. But equation 14 forces $w^\perp \perp \mathrm{col}(W_K)$ (indeed, $W_K^\top w^\perp = 0$ means $w^\perp$ is orthogonal to every column of W_K). Combined with $w^\perp \neq 0$, this is a contradiction.

Therefore $\mathcal{M}_{\mathrm{DAA}} \subsetneq \mathcal{M}_{\mathrm{DiffK}}$, which establishes Clause 1.

Part (c): Conditional $\mathcal{F}$-level strictness (Clause 2). Since $\mathcal{F}_\bullet = \Psi(\mathcal{M}_\bullet)$ for the common linear map $\Psi \colon M \mapsto QMX^\top$, inclusion at the $\mathcal{M}$-level transfers verbatim: $\mathcal{F}_{\mathrm{DAA}} \subseteq \mathcal{F}_{\mathrm{DiffK}}$ unconditionally.

For strict containment we use the explicit witness $M^\star$. Suppose Ψ is injective on $\mathbb{R}^{d_k \times d_{\mathrm{model}}}$ (so in particular on $\mathcal{M}_{\mathrm{DiffK}} \supseteq \mathcal{M}_{\mathrm{DAA}}$). Then $\Psi(M^\star) \neq 0$, and if $\Psi(M^\star) = \Psi(M')$ for some $M' \in \mathcal{M}_{\mathrm{DAA}}$ we would get $M^\star = M'$ by injectivity, contradicting $M^\star \notin \mathcal{M}_{\mathrm{DAA}}$. Hence $\Psi(M^\star) \in \mathcal{F}_{\mathrm{DiffK}} \setminus \mathcal{F}_{\mathrm{DAA}}$.

It remains to check that the sufficient condition equation **??** implies injectivity of Ψ. The map $\Psi(M) = QMX^\top$ is injective on $\mathbb{R}^{d_k \times d_{\text{model}}}$ iff Q has linearly independent columns and $X^\top$ has linearly independent rows — i.e. $\text{rank}(Q) = d_k$ and $\text{rank}(X) = d_{\text{model}}$. Under equation **??**, $\text{rank}(X) = d_{\text{model}}$ (since $N \geq d_{\text{model}}$ and X has full column rank), and $Q = XW_Q$ has $\text{rank}(Q) = \text{rank}(W_Q) = d_k$, since X has full column rank d_{model} and is therefore left-invertible on $\text{col}(W_Q)$. So equation **??** suffices. □

Remark B.1 (Milder sufficient condition for $\mathcal{F}$-strictness). Injectivity of Ψ on the *whole* ambient space $\mathbb{R}^{d_k \times d_{\text{model}}}$ is sufficient but not necessary for $\mathcal{F}_{\text{DAA}} \subsetneq \mathcal{F}_{\text{DiffK}}$. A sharper criterion follows from the rank identity:

$$\dim \mathcal{F}_{\text{DiffK}} = \text{rank}(Q) \cdot \text{rank}(X),$$
$$\dim \mathcal{F}_{\text{DAA}} = \text{rank}(Q) \cdot \text{rank}(XW_K).$$

Hence $\mathcal{F}_{\text{DAA}} \subsetneq \mathcal{F}_{\text{DiffK}}$ holds whenever

$$\text{rank}(Q) > 0 \quad \text{and} \quad \text{rank}(X) > \text{rank}(XW_K), \tag{15}$$

which is strictly weaker than equation **??** and, in particular, does not require $N \geq d_{\text{model}}$ or $\text{rank}(X) = d_{\text{model}}$.

Remark B.2 (Correcting the hand-derived equivalence). A naive algebraic derivation might equate $W_{\text{DAA}} W_K^\top = W_K^\top W_{D_K}^\top$ and "solve" for $W_{\text{DAA}} = W_K^\top W_{D_K}^\top (W_K^\top)^{-1}$, asserting that DAA and DiffK are equivalent. This step is ill-defined: under the standard pre-training regime $d_k < d_{\text{model}}$, the matrix $W_K^\top \in \mathbb{R}^{d_k \times d_{\text{model}}}$ is *fat* and has no two-sided inverse.

To see exactly where the equivalence fails, write $B := W_K^\top$ and substitute the pseudoinverse B^+ wherever the naive derivation uses B^{-1}. The identity $BB^+ = I_{d_k}$ holds (Eq. equation 9), but the *other* side does not: instead of $B^+B = I_{d_{\text{model}}}$, one only gets

$$B^+B = P_{\text{row}(B)} \neq I_{d_{\text{model}}}, \tag{16}$$

the orthogonal projector onto the d_k-dimensional subspace $\text{row}(B) = \text{col}(W_K)$. Consequently, given a DiffK parameter $M = BW_{D_K}^\top \in \mathcal{M}_{\text{DiffK}}$, the candidate DAA reconstruction $W_{\text{DAA}} := MB^+$ yields

$$W_{\text{DAA}} B = MB^+B = M P_{\text{row}(B)},$$

which equals M *only* when every row of M already lies in $\text{row}(B) = \text{col}(W_K)$. For generic DiffK M this constraint fails on a $d_k(d_{\text{model}} - d_k)$-dimensional set of directions — precisely the codimension computed in Clause 1 of Theorem 3.1. The pseudoinverse rescues the *DAA-to-DiffK* direction (Part (a) of the proof), where one only needs the right-inverse identity $BB^+ = I_{d_k}$, but it cannot rescue the *DiffK-to-DAA* direction, where one would need $B^+B = I_{d_{\text{model}}}$.

The correct relationship, derived in Theorem 3.1, uses the right pseudoinverse $(W_K^\top)^+$ satisfying $W_K^\top (W_K^\top)^+ = I_{d_k}$, and yields the asymmetric construction equation 10: DiffK can simulate any DAA, but DAA cannot simulate every DiffK. Once the algebra is done correctly, the two families are not equivalent; they stand in a strict-subset relationship, which is exactly what permits the regularization interpretation in Corollary 3.2.

Remark B.3 (On the full-column-rank assumption on W_K). The assumption $\text{rank}(W_K) = d_k$ is mild and essentially always satisfied in practice. For any standard pre-trained Transformer: (i) initialization schemes (e.g. truncated normal, Xavier) produce matrices with full column rank almost surely (full-rankness is a generic property under any continuous initialization distribution, since the rank-deficient locus has Lebesgue measure zero in $\mathbb{R}^{d_{\text{model}} \times d_k}$); (ii) full-rankness is verifiable on trained checkpoints via singular-value inspection — the smallest singular value of W_K is bounded well away from zero for all standard pre-trained models used in this paper — rather than relying on any mathematical guarantee that gradient descent preserves the property along the optimization trajectory. The assumption $d_k < d_{\text{model}}$ is structural: it holds whenever the model has more than one head ($d_k = d_{\text{model}}/H$ with $H \geq 2$), which covers every Transformer architecture used in this paper. The boundary case $d_k = d_{\text{model}}$ would make $W_K^\top$ square and (under full rank) invertible; in that case the inclusion in Theorem 3.1 becomes an equality, and DAA and DiffK *do* coincide as logit families. Multi-head attention — the only regime that matters in practice — is precisely the regime in which the strict-subset relation $\mathcal{F}_{\text{DAA}} \subsetneq \mathcal{F}_{\text{DiffK}}$ holds.

C Full Proof of Theorem 3.6 (Gradient Incentive)

This appendix contains the full three-step chain-rule proof of Theorem 3.6, the softmax-Jacobian propagation analysis (Remark C.1) including a discussion of when per-coordinate sign opposi-

tion fails at the logit level, and the two-empirical-predictions remark (Remark C.2) covering the parameter-level couplings that scope the loss-curve similarity prediction.

Setup and definitions. We work at a single attention head, a single query position i, and a fixed training step where the annealed coefficient takes the value $\lambda > 0$ (the boundary case $\lambda = 0$ is treated as Proposition 3.7 below). The DAA forward pass is

$$o_i \;=\; \sum_{j=1}^{N} \big(A_+^{ij} - \lambda\, A_-^{ij}\big)\, v_j \;\in\; \mathbb{R}^{d_v}, \qquad (17)$$

with $A_+, A_- \in \mathbb{R}^{N\times N}$ row-stochastic and $v_j \in \mathbb{R}^{d_v}$ the per-key value vector. To articulate *what the optimizer pushes attention toward*, we declare a position j to be a *signal* or a *distractor* relative to the current loss landscape:

Definition 3.5 is the natural *quantitative* analogue of the informal use of "noise token" / "signal token" in the attention literature: a token is a distractor when attending to it *currently* hurts the loss. The definition is loss-aware and depends on the data point and the current model state; it is not a static property of the token. We do *not* require any oracle labelling of signal vs distractor nor any auxiliary loss; the labels are read off from ∇L directly.

Proof. **Level of claim.** We treat o_i as a function $F : \mathbb{R}^{N\times N} \times \mathbb{R}^{N\times N} \times \mathbb{R}^{N\times d_v} \to \mathbb{R}^{d_v}$ given by equation 17, with arguments (A_+, A_-, V) (where V is the matrix whose j-th row is v_j). The partial derivatives computed below are *ambient* partial derivatives in this product space, treating each attention entry as a free real variable. The row-stochasticity constraint $\sum_j A_\pm^{ij} = 1$ is not imposed at this stage; the relationship between these ambient partials and the constrained optimization on the simplex (or on the underlying parameters $W_Q, W_K, W_{\mathrm{DAA}}$) is discussed in Remark C.1. The proof itself is a direct application of the chain rule to equation 17, followed by reading off the sign of the resulting inner product against $\nabla_{o_i} L$.

Step 1: Partial derivatives of o_i. From equation 17, o_i is linear in both A_+^{ij} and A_-^{ij} when all other entries are held fixed (the row-stochasticity constraints are not imposed at this stage; see the remark on softmax backprop below). Differentiating directly,

$$\frac{\partial o_i}{\partial A_+^{ij}} \;=\; v_j, \qquad \frac{\partial o_i}{\partial A_-^{ij}} \;=\; -\lambda\, v_j \;\in\; \mathbb{R}^{d_v}. \qquad (18)$$

For any key position $j' \neq j$, the entries $A_+^{ij'}$ and $A_-^{ij'}$ do not appear in the term indexed by j in equation 17; conversely, $A_\pm^{ij}$ does not appear in any term indexed by $j' \neq j$. Hence the Jacobians equation 18 are independent of the other attention entries.

Step 2: Chain rule. For any differentiable L whose dependence on the DAA layer factors through o_i, the multivariable chain rule gives

$$\begin{aligned} \frac{\partial L}{\partial A_+^{ij}} &= \Big\langle \nabla_{o_i} L,\ \tfrac{\partial o_i}{\partial A_+^{ij}} \Big\rangle \\ &= \big\langle \nabla_{o_i} L,\ v_j \big\rangle, \qquad (19) \\ \frac{\partial L}{\partial A_-^{ij}} &= \Big\langle \nabla_{o_i} L,\ \tfrac{\partial o_i}{\partial A_-^{ij}} \Big\rangle \\ &= \big\langle \nabla_{o_i} L,\ -\lambda\, v_j \big\rangle \\ &= -\lambda \big\langle \nabla_{o_i} L,\ v_j \big\rangle. \qquad (20) \end{aligned}$$

This establishes equation 5, and dividing yields equation **??** under $\lambda > 0$.

Step 3: Interpretation as complementary incentives. A gradient-descent update with learning rate $\eta > 0$ moves the attention entries by

$$\begin{aligned} \Delta A_+^{ij} &= -\eta\, \frac{\partial L}{\partial A_+^{ij}} \\ &= -\eta \big\langle \nabla_{o_i} L,\ v_j \big\rangle \\ &= \eta \big\langle -\nabla_{o_i} L,\ v_j \big\rangle, \qquad (21) \\ \Delta A_-^{ij} &= -\eta\, \frac{\partial L}{\partial A_-^{ij}} \\ &= \eta\lambda \big\langle \nabla_{o_i} L,\ v_j \big\rangle \\ &= -\eta\lambda \big\langle -\nabla_{o_i} L,\ v_j \big\rangle. \qquad (22) \end{aligned}$$

Substituting Definition 3.5:

- If j is a *signal* token for query i, then $\langle -\nabla_{o_i} L, v_j \rangle > 0$, so by equation 21–equation 22, $\Delta A_+^{ij} > 0$ and $\Delta A_-^{ij} < 0$. The positive branch raises its attention on j; the negative branch lowers its attention on j.
- If j is a *distractor* token for query i, then $\langle -\nabla_{o_i} L, v_j \rangle < 0$, so $\Delta A_+^{ij} < 0$ and $\Delta A_-^{ij} > 0$. The positive branch lowers its attention on j; the negative branch *raises* its attention on j.

The two branches therefore move in opposite directions on every key position j, with the negative branch tracking distractors and the positive branch tracking signal — exactly the role split posited informally in the noise hypothesis. The split is enforced by the chain rule alone; no auxiliary supervision is introduced. □

Remark C.1 (Propagation through softmax). Equation equation 5 establishes sign opposition at the level of *ambient* partial derivatives in (A_+, A_-). Pushing one further step through the row-wise softmax with Jacobian $J(p) = \mathrm{diag}(p) - pp^\top \succeq 0$ (PSD since $u^\top J(p)u = \mathrm{Var}_p(U)$), the chain rule yields

$$\nabla_{S_+^{i,:}} L = J(A_+^{i,:})\, g_i,$$
$$\nabla_{S_-^{i,:}} L = -\lambda\, J(A_-^{i,:})\, g_i, \tag{23}$$

where $S_\pm = \mathrm{softmax}^{-1}(A_\pm)$ are the pre-softmax logits and $g_i := \nabla_{A_+^{i,:}} L \in \mathbb{R}^N$. The global $-\lambda < 0$ factor encodes qualitative opposition along a one-dimensional scalar axis. However, $J(A_+^{i,:})$ and $J(A_-^{i,:})$ are in general distinct PSD operators, so the two logit-gradient vectors are *not* antiparallel: per-coordinate sign opposition can fail at individual key indices, and $\langle J(A_+^{i,:})\, g_i,\ J(A_-^{i,:})\, g_i\rangle$ need not have a fixed sign. Theorem 3.6 therefore guarantees an opposite-signed incentive at the *ambient attention-entry level*, while per-coordinate logit-level opposition is softer because of data-dependent reweighting by $J(A_\pm)$.

Remark C.2 (Two empirical predictions). The ambient gradient on A_+ in DAA coincides with the gradient acting on the lone attention map in standard fine-tuning: $\partial L/\partial A_+^{ij} = \langle \nabla_{o_i} L, v_j\rangle$. Two parameter-level couplings prevent this from forcing identical training-loss curves: (i) the forward o_i differs by $-\lambda \sum_j A_-^{ij} v_j$, which feeds back into $\nabla_{o_i} L$; (ii) in DAA the shared W_Q, W_K receive gradients through *both* branches. Both couplings vanish near initialization where $\lambda \to 0^+$ and W_{DAA} starts at identity (so $A_- \approx A_+$). Theorem 3.6 therefore predicts (P1) *near-initialization training-loss similarity* between DAA and standard fine-tuning, and (P2) *persistent downstream gains* driven by the opposite-signed incentive on A_-, which routes the negative branch onto distractor tokens without altering the cross-entropy target on signal tokens. The empirical observation that DAA and FT track each other on cross-entropy while diverging on noise-bottleneck downstream tasks (Figure 2, Table 1) is exactly this pair of predictions.

The argument requires only that L is differentiable in o_i and contains no direct regularizer on $A_\pm^{ij}$. Cross-entropy on a gold target is the canonical case; the same opposition holds for any differentiable loss factoring through o_i (masked LM, regression, preference surrogates), with the labels "signal" / "distractor" in Definition 3.5 re-instantiated per loss.

D Extended Related Work

Parameter-efficient fine-tuning (PEFT). LoRA (Hu et al., 2022), prefix tuning (Li and Liang, 2021), and adapter methods (Houlsby et al., 2019) are the dominant fine-tuning strategies for pre-trained Transformers. All of them stay within the existing softmax-attention function class: LoRA injects low-rank perturbations $\Delta W = BA$ into the linear weights W_Q, W_K, W_V but leaves $A \in \Delta^N$ structurally non-negative; prefix tuning and adapter-style methods modify activations without altering the simplex constraint. DAA is orthogonal to these methods at the level of expressivity: where LoRA increases the family of pre-softmax logits reachable, DAA adds an architectural primitive (signed effective attention via $A_+ - \lambda A_-$) that no purely-positive parameterization of W_Q, W_K, W_V can realize. DAA could in principle be stacked on top of a LoRA adapter; we leave such composition to future work.

Attention sinks and noise structure. Xiao et al. (2024) documented that pre-trained Transformers accumulate disproportionate attention on a small number of "sink" tokens—typically the BOS token and high-frequency function words—and used this to motivate StreamingLLM-style cache management. Gu et al. (2025) subsequently traced when this sink behavior emerges during pre-training, showing it is a structural consequence of softmax rather than an artifact of any specific corpus or scale; Sun et al. (2024) further connected sink positions to "massive activations" in the residual stream, suggesting the phenomenon spans attention and feed-forward layers. Our work targets the same phenomenon from an architectural-expressivity angle: a sink is a token that softmax cannot reduce attention to without compensating mass elsewhere, and DAA's A_- branch is observed (§4.3) to concentrate on exactly the high-frequency / BOS positions identified by this line of work.

Softmax-level alternatives to sink mitigation. A parallel line of work removes the sink by replacing the softmax itself. Qiu et al. (2025) insert a head-specific sigmoid gate that empirically eliminates attention sinks under matched parameter counts, and Zuhri et al. (2025) propose a rectified softmax ("Softpick") that drops the sum-to-one constraint and removes massive activations. These methods are orthogonal to DAA in two ways: (i) they require either pre-training or non-trivial re-training to integrate with existing checkpoints, whereas DAA is a fine-tuning-time intervention; (ii) they modify the activation function, whereas DAA leaves the softmax intact and adds a subtractive branch on top. The two approaches address the same underlying defect from opposite sides.

Position bias and long-context attention. Liu et al. (2023) documented the "lost-in-the-middle" phenomenon, where attention degrades for middle-of-context tokens; Wu et al. (2025) recently unified this with attention sinks under a single graph-theoretic position-bias framework. While DAA does not directly target position bias, both phenomena are instances of attention noise that the negative-capacity branch can cancel at the source. A controlled evaluation of DAA on long-context retrieval beyond NIAH (e.g., RULER, LongBench) is left to future work (§5).

Missing-primitive precedents. The framing of an adaptation as restoring a missing architectural primitive has direct precedents in the Transformer literature: relative-position primitives (RoPE (Su et al., 2024), ALiBi (Press et al., 2022)) restore translation-equivariant positional information that vanilla absolute encodings cannot express; FlashAttention (Dao et al., 2022) restores I/O-efficient softmax computation without changing the function. Our claim is analogous in structure: *negative capacity* is an expressivity primitive absent from pretrained softmax attention, and DAA restores it at the minimal parameter cost compatible with retrofit to existing checkpoints.

E Qualitative Failure Modes of Standard Attention

The $+4.6$ accuracy gain on needle retrieval over the Base model (§4.1) and the 56% entropy reduction in Figure 3 point to a consistent failure mode in the unadapted model: when the prompt places the answer-bearing token deep in a context dominated by syntactic glue (articles, prepositions, BOS), the model's attention at the final query position spreads probability across the glue and never concentrates on the answer. Standard fine-tuning shifts the simplex slightly but cannot *reduce* the share on glue tokens without compensating elsewhere; the entropy stays high and retrieval accuracy moves only marginally (Base $84.9 \rightarrow$ FT 85.3, Table 1). DEX likewise leaves the underlying attention distribution unchanged and corrects only the value-weighted output, gaining $+2.5$. DAA, by contrast, learns an A_- branch that absorbs the glue mass at the source—the subtraction $A_+ - \lambda A_-$ then frees probability for the answer token, and the entropy drop on this same family of examples is exactly the $2.01 \rightarrow 0.88$ visualized in Figure 3. The pattern is consistent across the entropy-reduction examples we inspected: DAA's gains concentrate on inputs where the gold token's neighborhood is dense with attention sinks, and shrink to near zero on inputs where the standard attention map is already low-entropy. This is the regime negative-capacity injection is designed for, and is consistent with the noise-bottleneck framing of §4.1 and §4.2.

F DAA Forward Pass Pseudocode

Algorithm 1 states the DAA forward pass procedurally. The equivalent eager-mode pseudocode used in our PyTorch implementation appears in Appendix J.

Algorithm 1 Differential Attention Adaption (DAA) forward pass.

Require: Hidden states $X \in \mathbb{R}^{N \times d_{\text{model}}}$; pretrained W_Q, W_K, W_V; training step t, horizon T.
Require: Learnable $W_{\text{DAA}} \in \mathbb{R}^{d_k \times d_k}$ (init. I); λ_{init} (const.); λ_{learn} (init. 0).
Ensure: Attention output $O' \in \mathbb{R}^{N \times d_v}$.

```
Q, K, V ← XW_Q, XW_K, XW_V {standard projections (frozen)}
A_+ ← softmax(QK^⊤/√d_k)  {primary attention map}
A_- ← softmax(Q W_DAA K^⊤/√d_k)  {negative-capacity branch}
α ← min(1, t/T)  {annealing weight}
λ(t) ← (1 − α) (t/T) λ_init + α λ_learn
O' ← (A_+ − λ(t) A_-) V  {signed effective attention}
return O'
```

G Parameter Count of Each Attention Architecture

In this section, we provide a detailed derivation of the number of parameters for each attention layer in the architectures discussed in this paper.

G.1 Standard Transformer Attention

A standard multi-head attention layer consists of four main learnable weight matrices: W_Q, W_K, W_V, W_O, each of size $d_{\text{model}} \times d_{\text{model}}$. The total number of parameters is:

$$N_{\text{Standard}} = 4d_{\text{model}}^2$$

G.2 Differential (DIFF) Transformer

The Differential Transformer essentially creates two parallel attention streams and computes their difference.

$$N_{\text{DIFF}} = 3d_{\text{model}}^2 + 3d_{\text{model}}^2 + d_{\text{model}}^2 = 7d_{\text{model}}^2$$

G.3 DEX (Differential Extension)

DEX adds a learnable matrix W_{DEX} per head of size $d_v \times d_v$.

$$N_{\text{DEX}} = 4d_{\text{model}}^2 + \frac{d_{\text{model}}^2}{h} = \left(4 + \frac{1}{h}\right) d_{\text{model}}^2$$

G.4 DAA (Differential Attention Adaption)

DAA introduces a learnable matrix W_{DAA} per head of size $d_k \times d_k$.

$$N_{\text{DAA}} = 4d_{\text{model}}^2 + \frac{d_{\text{model}}^2}{h} = \left(4 + \frac{1}{h}\right) d_{\text{model}}^2$$

G.5 Input Differentiation Architectures (DiffQ, DiffK, DiffV)

These methods introduce a single learnable matrix of size $d_{\text{model}} \times d_{\text{model}}$.

$$N_{\text{DiffQ/DiffK/DiffV}} = 4d_{\text{model}}^2 + d_{\text{model}}^2 = 5d_{\text{model}}^2$$

H Complexity Analysis: Computational Overhead

While the parameter efficiency of DAA is discussed in Appendix G, it is equally important to analyze the computational complexity, specifically the Floating Point Operations (FLOPs) required during the forward pass. This analysis highlights the trade-offs between inference latency and noise mitigation capability.

We use the following notation in Appendix G. The analysis focuses on a single attention layer.

H.1 Breakdown of Operations

The computational cost of attention mechanisms splits into two components:

1. *Linear operations* ($\mathcal{O}(N)$): projections of inputs (Q, K, V) and outputs, which scale linearly with sequence length but quadratically with model dimension (d_{model}^2).

2. *Quadratic operations* ($\mathcal{O}(N^2)$): query-key matrix multiplication and the attention-value weighted summation, which scale quadratically with sequence length.

Standard Transformer. The standard self-attention consists of projecting inputs ($3Nd_{\text{model}}^2$), computing attention scores ($N^2 d_{\text{model}}$), computing the weighted sum ($N^2 d_{\text{model}}$), and the final output projection (Nd_{model}^2):

$$\mathcal{C}_{\text{Standard}} \approx 4Nd_{\text{model}}^2 + 2N^2 d_{\text{model}}. \tag{24}$$

DEX (Differential Extension). DEX applies a differential correction to the value outputs: standard attention followed by a per-head linear transformation on the output value vectors. This introduces an additional linear term in N but no extra quadratic operations.

$$\begin{aligned}\mathcal{C}_{\text{DEX}} &\approx \mathcal{C}_{\text{Standard}} + \mathcal{O}\big(Nhd_k^2\big) \\ &= 4Nd_{\text{model}}^2 + 2N^2 d_{\text{model}} + \tfrac{Nd_{\text{model}}^2}{h}.\end{aligned} \tag{25}$$

DEX is computationally the cheapest adaptation method, adding only a marginal linear overhead.

DAA (ours). DAA intervenes in the attention score calculation. Beyond the standard operations, DAA requires two extra pieces: a per-head linear transformation $Q' = QW_{\text{DAA}}$ at cost $\mathcal{O}(Nd_{\text{model}}^2/h)$, and a secondary attention map $A_2 = \text{softmax}(Q'K^\top)$, which introduces a second $QK^\top$ multiplication at cost $\mathcal{O}(N^2 d_{\text{model}})$.

$$\mathcal{C}_{\text{DAA}} \approx \left(4Nd_{\text{model}}^2 + \frac{Nd_{\text{model}}^2}{h}\right) + \mathbf{3}N^2 d_{\text{model}} \tag{26}$$

While DAA increases the quadratic complexity coefficient (from 2 to 3 roughly, accounting for the second score computation), it avoids heavy global linear projections.

Input differentiation (DiffQ, DiffK, DiffV). These methods transform the global input hidden states X via a large matrix $W_D \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ before the standard projections. The input transform

contributes a substantial linear cost of Nd_{model}^2, and DiffQ / DiffK additionally compute a second set of attention scores for the subtraction (as in DAA).

$$\mathcal{C}_{\text{DiffQ/K}} \approx \mathbf{5}Nd_{\text{model}}^2 + \mathbf{3}N^2 d_{\text{model}} \tag{27}$$

DiffV avoids the second attention score computation but still incurs the heavy input projection cost ($\approx 5Nd_{\text{model}}^2 + 2N^2 d_{\text{model}}$).

H.2 Summary of Computational Complexity

Table 4 summarizes the FLOPs comparison.

Two takeaways from the comparison. DAA pays the same quadratic cost as DiffQ/DiffK (the dual attention map needed for cancellation) but pays a smaller linear cost than they do ($4ND^2$ vs. $5ND^2$). DEX is the cheapest of the adaptation methods, since it avoids a second attention map; the main experiments show that its post-hoc correction is also less effective at noise mitigation than DAA's intervention inside the score calculation. DAA therefore trades a quadratic-term increase for improved noise-bottleneck performance.

I Input-Differentiation Variants

For completeness, we describe three additional implementations of negative-capacity injection at the *input* position, used as baselines in our experimental comparisons (Table 2). By Theorem 3.1 in the main text, the K-side family below (DiffK) is a strict over-parameterization of the Q-K logit family that DAA reaches with d_k^2 parameters per head; the Q-side family (DiffQ) is symmetric, and DiffV operates on the value stream instead. All three introduce a single learnable matrix $W_D \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$, initialized to the identity so training begins from a standard attention configuration. See §3.2 for the theoretical relationship to DAA.

Differential Adaption via Query (DiffQ). A second query matrix Q_2 is generated from the transformed input $X' = XW_{D_Q}$, while the standard query Q_1 is computed from the original input X. The attention output is the difference between the two attention maps:

$$\begin{aligned} Q_1 = XW_Q, \quad Q_2 = X'W_Q = (XW_{D_Q})W_Q \\ A_1 = \text{softmax}\left(\frac{Q_1 K^T}{\sqrt{d_k}}\right) \\ A_2 = \text{softmax}\left(\frac{Q_2 K^T}{\sqrt{d_k}}\right) \\ O' = (A_1 - \lambda(t) A_2)V \end{aligned} \tag{28}$$

Differential Adaption via Key (DiffK). Symmetrically, a second key matrix K_2 is generated from the transformed input X', while the standard key K_1 is derived from X. This evaluates the query against two different content representations:

$$\begin{aligned} K_1 = XW_K, \quad K_2 = X'W_K = (XW_{D_K})W_K \\ A_1 = \text{softmax}\left(\frac{QK_1^T}{\sqrt{d_k}}\right) \\ A_2 = \text{softmax}\left(\frac{QK_2^T}{\sqrt{d_k}}\right) \\ O' = (A_1 - \lambda(t) A_2)V \end{aligned} \tag{29}$$

Differential Adaption via Value (DiffV). This approach modifies the value stream directly rather than subtracting attention distributions. A modified value matrix V' is computed by applying a dynamically weighted differential transformation to the input sequence, before the standard attention-weighted sum. The method effectively creates a primary value stream (V_1) and a secondary stream (V_2) that is subtracted from it:

$$\begin{aligned} W_E &= I - \lambda(t)\, W_{D_E} \\ V' &= (XW_E)W_V \\ &= XW_V - \lambda(t)\, XW_{D_E}W_V \\ &= V_1 - \lambda(t)\, V_2 \\ A &= \text{softmax}\left(\tfrac{QK^\top}{\sqrt{d_k}}\right) \\ O' &= AV' = AV_1 - \lambda(t)\, AV_2. \end{aligned} \tag{30}$$

Here, W_E serves as an effective transformation matrix that directly modulates the information carried by the value vectors. DiffV is distinct from DEX in that the modification occurs before the final attention-weighted sum (acting on inputs to the V projection) rather than after.

J Implementation of Differential Adaption

In this section, we provide the pseudocode for our proposed differential fine-tuning architectures. These implementations illustrate how each method modifies the standard self-attention mechanism in a lightweight manner. The variable `X` represents the input tensor of hidden states, `W_q`, `W_k`, and `W_v` are the standard projection matrices for query, key, and value, respectively. The parameter `lambda` is the learnable, time-annealed scalar that controls the magnitude of the differential component. All newly

Table 4: Computational Complexity Comparison per Attention Layer. The "Quadratic Term" is the bottleneck for long sequences ($N \gg d_{\text{model}}$), while the "Linear Term" dominates for shorter sequences.

Method	Linear Term (Projections)	Quadratic Term (Attention)	Major Overhead Source
Standard	$4Nd_{\text{model}}^2$	$2N^2d_{\text{model}}$	-
DEX	$4Nd_{\text{model}}^2 + \frac{1}{h}Nd_{\text{model}}^2$	$2N^2d_{\text{model}}$	Output Correction
DiffQ / DiffK (Ours)	$\mathbf{5}Nd_{\text{model}}^2$	$\mathbf{3}N^2d_{\text{model}}$	Global Input Proj. + Extra Attn Map
DiffV (Ours)	$\mathbf{5}Nd_{\text{model}}^2$	$2N^2d_{\text{model}}$	Global Input Projection
DAA (Ours)	$4Nd_{\text{model}}^2 + \frac{1}{h}Nd_{\text{model}}^2$	$\mathbf{3}N^2d_{\text{model}}$	Extra Attn Map ($QW_{\text{DAA}}K^\top$)

introduced matrices (W_daa, W_dq, etc.) are initialized as identity matrices to preserve the model's pre-trained knowledge at the start of fine-tuning.

We use a Python-like syntax for clarity. The operator @ denotes matrix multiplication, and tensor shapes are provided in comments, where b is the batch size, n is the sequence length, and d is the dimension of the head.

J.1 DAA (Differential Attention Adaption)

DAA directly intervenes in the attention score computation by introducing a learnable matrix W_daa into the query-key interaction. This allows the model to learn a transformation that creates a secondary, noise-focused attention map, which is then subtracted from the original. This is our primary and most effective proposed method.

Listing 1: Pseudocode for Differential Attention Adaption (DAA).

```
def DAA(X, W_q, W_k, W_v, W_daa,
    lambda):
     # Project inputs to query,
    key, and value
     Q = X @ W_q
     K = X @ W_k
     V = X @ W_v

     # Scaling factor
     s = 1 / sqrt(d)

     # Calculate the primary
    attention scores
     A1 = softmax(Q @
    K.transpose(-1, -2) * s)

     # Calculate the secondary,
    differentiated attention
    scores
     A2 = softmax(Q @ W_daa @
    K.transpose(-1, -2) * s)

     # Return the differentially
    weighted value
     return (A1 - lambda * A2) @ V
```

J.2 Architectures via Input Differentiation

As an alternative to DAA, we explored three methods that apply the differential mechanism at the input level. These approaches create a secondary, differentiated version of either the query, key, or value stream by transforming the input hidden states X with a learnable matrix before the standard projection.

Differential Adaption via Query (DiffQ). In this variant, we generate two distinct sets of queries. The first, Q1, is standard, while the second, Q2, is derived from a transformed input. The final output is based on the difference between the attention maps produced by these two queries.

Listing 2: Pseudocode for Differential Adaption via Query (DiffQ).

```
def DiffQ(X, W_q, W_k, W_v,
    W_dq, lambda):
     # Standard K and V
    projections
     K = X @ W_k
     V = X @ W_v

     # Generate primary and
    secondary queries
     Q1 = X @ W_q
     Q2 = (X @ W_dq) @ W_q
     # Q1, Q2, K, V: [b, n, d]

     # Scaling factor
     s = 1 / sqrt(d)

     # Calculate attention scores
    for each query
     A1 = softmax(Q1 @
    K.transpose(-1, -2) * s)
```

```
    A2 = softmax(Q2 @
   K.transpose(-1, -2) * s)

    # Return the differentially
   weighted value
    return (A1 - lambda * A2) @ V
```

Differential Adaption via Key (DiffK). This approach is analogous to DiffQ, but the differentiation is applied to the key stream. The model learns to compare the same query against two different representations of the input content (keys `K1` and `K2`).

Listing 3: Pseudocode for Differential Adaption via Key (DiffK).

```
def DiffK(X, W_q, W_k, W_v,
   W_dk, lambda):
    # Standard Q and V
   projections
    Q = X @ W_q
    V = X @ W_v

    # Generate primary and
   secondary keys
    K1 = X @ W_k
    K2 = (X @ W_dk) @ W_k
    # Q, K1, K2, V: [b, n, d]

    # Scaling factor
    s = 1 / sqrt(d)

    # Calculate attention scores
   for each key
    A1 = softmax(Q @
   K1.transpose(-1, -2) * s)
    A2 = softmax(Q @
   K2.transpose(-1, -2) * s)

    # Return the differentially
   weighted value
    return (A1 - lambda * A2) @ V
```

Differential Adaption via Value (DiffV). Unlike the other methods, DiffV applies the differential mechanism directly to the value stream after the attention scores have been computed. It calculates a single attention map `A` and uses it to weigh the difference between a primary value `V1` and a secondary value `V2`.

Listing 4: Pseudocode for Differential Adaption via Value (DiffV).

```
def DiffV(X, W_q, W_k, W_v,
   W_dv, lambda):
    # Standard Q and K
   projections
    Q = X @ W_q
    K = X @ W_k

    # Generate primary and
   secondary values
    V1 = X @ W_v
    V2 = (X @ W_dv) @ W_v
    # Q, K, V1, V2: [b, n, d]

    # Scaling factor
    s = 1 / sqrt(d)

    # Calculate a single
   attention score matrix
    A = softmax(Q @
   K.transpose(-1, -2) * s)

    # Apply attention to the
   difference of the values
    return A @ (V1 - lambda * V2)
```

K Implementation details

All experiments were conducted using the Hugging Face `transformers`, `datasets`, and `safetensors` (Penedo et al., 2022) libraries, with PyTorch (Paszke et al., 2019) as the backend framework. Below we detail the specific configurations for each model family.

K.1 GPT-2 Experiments

Model and Data. The experiments were based on the publicly available GPT-2 base model (117M parameters). For fine-tuning, we used a subset of the OpenWebText corpus (Peterson et al., 2019), which is textually similar to the model's original pre-training data, to ensure that the model retained its fundamental language capabilities. The models were evaluated on perplexity using held-out portions of OpenWebText (both seen and unseen during fine-tuning) and WikiText-2 (Kwiatkowski et al., 2019). We also evaluated task-specific performance using the LAMBADA dataset (accuracy) (Grave et al., 2016) and a Needle-in-a-Haystack retrieval task (accuracy).

Training Hyperparameters. For all GPT-2 based experiments, we fine-tuned only the parame-

ters within the attention modules (`attn`), freezing all other model weights. This amounted to training approximately 28.3M parameters (standard attention), 28.9M parameters (DAA, DEX), 35.4M parameters (DiffQ, DiffK, DiffV). The shared training configuration was as follows:

- Optimizer: AdamW (`adamw_torch`).
- Learning rate: 3e-5.
- LR scheduler: cosine decay with 500 warmup steps.
- Epochs: 3.
- Effective batch size: 32.
- Sequence length: 512 tokens.
- Precision: FP16 mixed-precision.
- Weight decay: 0.01.

Hardware. All GPT-2 fine-tuning experiments were conducted on a single server equipped with one NVIDIA A800 80GB GPU.

K.2 Llama Experiments

Model and Data. To validate the general applicability of our methods, we conducted further experiments on more recent and larger models: Llama-3.2-1B and Llama-3.1-8B. For supervised fine-tuning (SFT), we utilized a subset of the `allenai/tulu-v2-sft-mixture` dataset (Ivison et al., 2023), which is a collection of high-quality instruction-following data. Model performance was evaluated on a suite of common sense reasoning benchmarks, including ARC-Easy, ARC-Challenge, BoolQ, Hellaswag, OpenBookQA, WIC, Winogrande, and WSC (Levesque et al., 2012; Zellers et al., 2019).

Training Hyperparameters. For the Llama models, we adopted a parameter-efficient fine-tuning strategy where only the weights of the self-attention modules (`self_attn`) and the language model head (`lm_head`) were updated. All other parameters, including embeddings and feed-forward layers, remained frozen. The key training parameters are listed below:

- Optimizer: AdamW (`adamw_torch`).
- Learning rate: 1e-4.
- LR scheduler: cosine decay with 500 warmup steps.
- Epochs: 3.
- Effective batch size: 32.
- Precision: BF16 mixed-precision (native hardware support).
- Gradient clipping: max norm 1.0.
- Gradient checkpointing: enabled, to reduce memory.

The tokenizer's padding token was set to its end-of-sequence (EOS) token, and a custom data collator was used to correctly handle the masking of labels for instruction-formatted data.

Hardware. The all Llama experiments were conducted on a server with NVIDIA A800 80GB GPUs.

K.3 Ablation on λ_{init}

Table 5 shows DAA performance on the language modeling benchmarks (average over 8 tasks from the table 1, using Llama-3.2-1B) when varying the λ_{init} strategy. The results indicate relative robustness to different fixed scalar initializations (0.2–0.8).

Table 5: Ablation on the annealing initialization λ_{init} for DAA on Llama-3.2-1B (8-task reasoning average). Performance is robust across the tested range; the no-annealing baseline ($\lambda_{\text{init}} = 0$) is significantly lower (51.98) than any annealed variant.

λ_{init}	0 (no anneal)	0.2	0.5	0.8
LM Acc (%)	51.98	53.37	53.25	**53.3**

L Alternative Interpretations

The main paper develops the case for DAA through three formal devices: the K-side equivalence theorem (Theorem 3.1), the post-softmax non-negativity obstruction (Proposition 3.3), and the gradient-incentive theorem (Theorem 3.6). For completeness, we record here a heuristic Taylor-expansion view that motivated an earlier version of this work; we present it as an *alternative interpretation* rather than as a stand-alone justification, since the gradient-incentive theorem is the optimization-theoretic statement that actually does the work in the present paper.

Connection to Taylor approximation. If one models the computed attention logits as $S_{\text{computed}} = S_{\text{ideal}} + \xi$ with ξ an additive perturbation, a first-order Taylor expansion of the row-wise softmax gives

$$\begin{aligned}\mathrm{softmax}(S_{\text{ideal}}) \approx\ & \mathrm{softmax}(S_{\text{computed}}) \\ & - J\big(\mathrm{softmax}(S_{\text{computed}})\big)\,\xi,\end{aligned}$$

where J is the row-wise softmax Jacobian. The DAA forward pass equation 6 has the matching structure $A_+ - \lambda A_-$ with A_- produced by a bilinear $QW_{\text{DAA}}K^\top$ form. If the noise admits a low-rank bilinear approximation $\xi \approx QM_{\text{noise}}K^\top$, the architectural form of DAA aligns with the noise generation source, providing an inductive-bias argument for why the parameterization is well-matched to the task. This view is consistent with equation 5 of Theorem 3.6: the gradient on A_- tracks the negative of the gradient on A_+, which is the optimization-level shadow of the Taylor-expansion correction. We retain this view here only as motivation; the formal claims used in the main paper rest on Theorems 3.1–3.6 and Proposition 3.3, not on the Taylor approximation.

M Responsible NLP Checklist

We summarize the information requested by the ACL Responsible NLP Checklist; full hyperparameters and infrastructure details appear in Appendix K.

Models and code. We adapt three publicly available pre-trained Transformers: GPT-2 (117M parameters; MIT license) (Radford et al., 2019), Llama-3.2-1B, and Llama-3.1-8B (Meta Llama 3 Community License) (Dubey et al., 2024). Our use of these checkpoints is consistent with their respective licenses, which permit research use. All experimental code is built on Hugging Face `transformers`, `datasets`, and `safetensors` (Penedo et al., 2022) with PyTorch (Paszke et al., 2019) as the backend. Source code and trained DAA weights will be released upon publication under an open-source license.

Data. For GPT-2 fine-tuning we use a subset of OpenWebText (Peterson et al., 2019) (publicly released, English). For Llama fine-tuning we use a subset of the `allenai/tulu-v2-sft-mixture` instruction dataset (Ivison et al., 2023) (publicly released, primarily English). Evaluation datasets—WikiText-2 (Kwiatkowski et al., 2019), LAMBADA (Grave et al., 2016), ARC-Easy/Challenge, BoolQ, HellaSwag, OpenBookQA, WIC, WinoGrande, WSC (Clark et al., 2018; Wang et al., 2019; Mihaylov et al., 2018; Bisk et al., 2020; Sakaguchi et al., 2021; Zellers et al., 2019; Levesque et al., 2012)—are all publicly available, English-language NLP benchmarks. No new data are collected; no personally identifying information or offensive content is introduced by our method.

Computing infrastructure. All experiments run on NVIDIA A800 80GB GPUs. GPT-2 fine-tuning uses one GPU; Llama experiments use the same A800-class hardware. Total compute for the experiments reported in this paper is on the order of 10^2 GPU-hours.

Parameter counts and hyperparameter search. DAA adds d_k^2 parameters per head ($<1\%$ of base; full derivation in Appendix G). We did not perform hyperparameter search; learning rates and schedules follow community defaults for each backbone (Appendix K). The only DAA-specific hyperparameter is the annealing schedule $\lambda(t)$; we ablate λ_{init} in Appendix K (Table 5) and find performance robust across the tested range.

Statistical reporting. Reported accuracies and perplexities are single-seed runs; we flag this as a limitation in §5 and note that the consistency of gains across three model scales (GPT-2 / 1B / 8B) provides indirect robustness evidence.

Human annotation. No human annotation, crowdsourcing, or human-subject evaluation was conducted.

Risks and intended use. The proposed method is intended for research on attention mechanisms and parameter-efficient fine-tuning of pre-trained language models. Risk discussion appears in the Ethical Considerations section.

Use of AI assistants. LLM-assisted editing of the manuscript is disclosed in the "Use of Large Language Models" section.