Do Value Vectors in Deep Layers Need Context from the Residual Stream?

Bank of Values (BoV) — research paper discussion

Paper: Muyu He, Yuchen Liu, Qingya Huang, Li Zhang

arXiv:2606.02780v5 (23 August 2026)
Reading: 17 September 2026

Deep-layer values can be a token content bank

Main result

Replace the value projection in the last third of layers with a layer-specific token lookup. Keep contextual queries and keys.

- 780M-scale, 100B tokens: BPB 0.671 $\rightarrow$ 0.663; CORE 0.334 $\rightarrow$ 0.351.
- 16 of 21 tasks improve; 5 regress.
- Evidence covers 135M/780M scales; deployment gains remain conditional.

Paper, Secs. 4–5; Tables 2 and 6, PDF pp. 6, 13.

The question is what deep values need to carry

- Queries and keys determine which positions receive attention.
- Values supply the content mixed by those attention weights.
- Earlier work adds token information, or shares first-layer values across all layers.
- BoV isolates depth, value source, scaling, and addition versus replacement.

Paper, Secs. 1–3, PDF pp. 1–4.

Context survives through queries, keys, and residuals

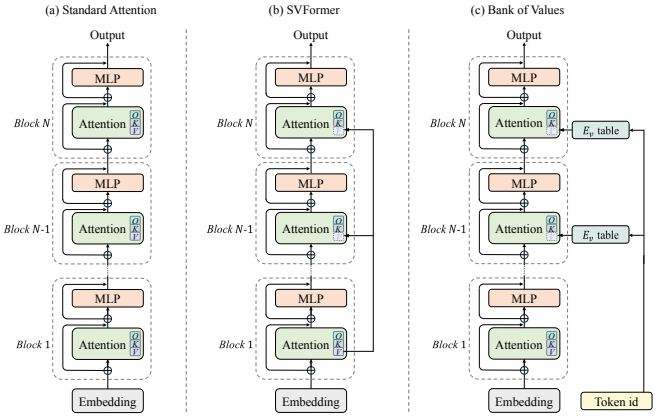
$$\mathbf{Q} = \mathbf{x}_i \mathbf{W}_Q, \quad \mathbf{K} = \mathbf{x}_i \mathbf{W}_K, \quad \mathbf{V} = \mathbf{x}_i \mathbf{W}_V$$

$$\mathbf{O}^{(h)} = \mathrm{softmax}\left(\frac{\mathbf{Q}^{(h)} \mathbf{K}^{(h)\top}}{\sqrt{d_h}} \right) \mathbf{V}^{(h)}$$

- $\mathbf{x}_i$: contextual residual input; $d_h = d/H$.
- $\mathbf{x}_0 = \mathrm{RMSNorm}(\mathbf{e})$: original token embedding.
- BoV changes only the value source in selected layers. Causal masking is omitted above.

Paper, Sec. 2, Eqs. (1)–(2), PDF p. 3.

BoV gives each deep layer its own table



Paper, Fig. 1, PDF p. 2. Original figure; only the last third of layers use BoV.

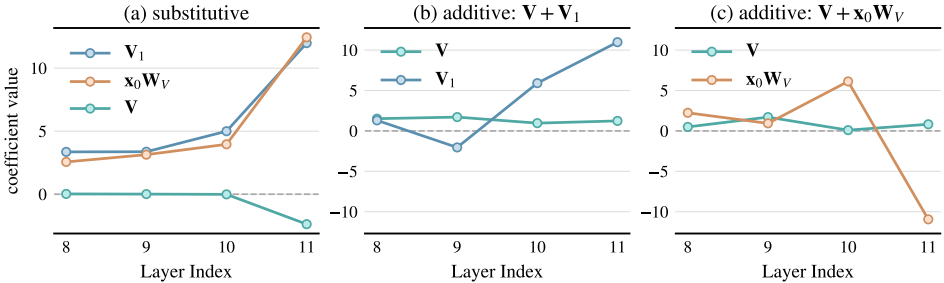
Depth restriction is essential in the 135M study

Value source	Target / coefficient	BPB ↓
Standard $\mathbf{V}$	Baseline	0.854
Shared $\mathbf{V}_1$	All layers / fixed 1	0.872
Shared $\mathbf{V}_1$	Last 4 / fixed 1	0.849
Shared $\mathbf{V}_1$	Last 4 / learned	0.847
Layer-specific $\mathbf{x}_0\mathbf{W}_V$	Last 4 / learned	**0.845**
$\mathbf{V} + \mathbf{x}_0\mathbf{W}_V$	Last 4 / learned	**0.845**

Adding contextual values gives no gain in the last pair.

Paper, Table 1, PDF p. 3. All runs: 1.50×10^{18} FLOPs. These are projection variants, not the BoV lookup table.

Learned scales favor the context-free component



- Horizontal axis: layer indices 8–11 (zero-based).
- Scales are unbounded and can be negative.
- Interpretation: combine this pattern with ablations; scale magnitude alone is not a causal contribution measure.

Paper, Fig. 2, PDF p. 4; final bullet is reading analysis.

BoV learns values directly as parameters

$$\mathbf{E}_v \in \mathbb{R}^{|\mathcal{V}| \times d}, \qquad \boxed{\mathbf{v}_p = \gamma_v \mathbf{E}_v[i_p]}$$

- i_p: token ID at position p; γ_v: learned per-layer scalar.
- One independent table for each of the last $L/3$ layers; remove their $\mathbf{W}_V$.
- Initialize each row from $\mathrm{RMSNorm}(\mathbf{e}_i)\mathbf{W}_V$.
- Then train table entries directly; the initial factorization need not persist.

Paper, Secs. 4–5, Eqs. (3)–(5), PDF pp. 5–6.

A fixed value does not imply a fixed interpretation

Teaching example, not a paper experiment

The token “bank” has the same lookup value in two contexts at the same BoV layer. Its contextual keys, attention weights, and final residual state can still differ.

- Repeated IDs reuse content vectors.
- Context determines how those vectors are selected and combined.
- Earlier layers still compute standard contextual values.

Derived from the data flow in Sec. 4 and Fig. 1, PDF pp. 2, 5.

Decode keeps keys and IDs, then gathers values

- Training / prefill: look up the current token sequence.
- Decode: retain historical K and token IDs at target layers.
- Gather historical V for one layer, compute attention, release temporary V.
- History gathering moves $O(Td)$ data; it is not zero-cost memory access.
- Host offload / prefetch is proposed, without measured serving throughput.

Paper, Fig. 3 and Sec. 4, PDF pp. 5–6.

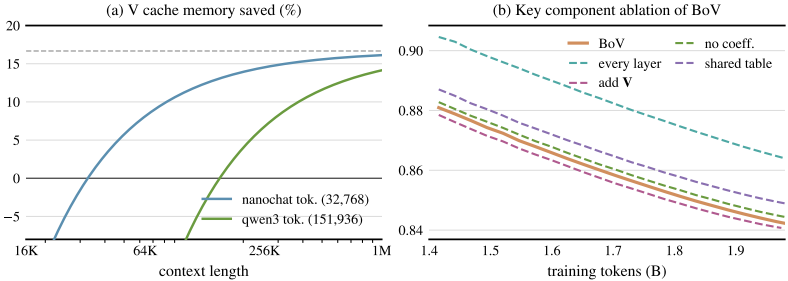
Memory savings require an explicit accounting model

$$\Delta_{\mathrm{KV}} = \frac{L}{3} d (T - |\mathcal{V}|)$$

- Paper equation counts elements; multiply by bytes per element.
- One-sequence break-even: $T = |\mathcal{V}|$ under the simplified model.
- Tables cost fixed memory; temporary V and K caching still exist.
- Batching, sliding windows, KV width, and offload change the accounting.

Paper, Eq. (6), PDF p. 5. Accounting qualifications: reading analysis.

The memory figure has a denominator ambiguity



$$\underbrace{\frac{\Delta_{\mathrm{KV}}}{2LTd}}_{\text{fraction of total K+V cache}} = \frac{1}{6}\left(1 - \frac{|\mathcal{V}|}{T}\right) \to 16.7\%$$

The left panel says “V cache”, but its asymptote matches total K+V.

Paper, Fig. 5(a), PDF p. 8; denominator check derived from Eq. (6). Qwen3 is a vocabulary-size example, not a tested model.

Three budgets answer different questions

Model scale	Controlled budget	Training tokens
135M	1.50×10^{18} FLOPs	Baseline: 1.97B
780M	3.91×10^{19} FLOPs	Baseline: 8.19B BoV: 8.39B
780M	100B tokens	100B each

- Sequence length 2048; vocabulary 32,768; bf16.
- 12/24 layers, width 768/1536, 6/12 attention and KV heads.
- Muon for matrices; AdamW for other parameters, including the BoV table.

Paper, Secs. 3, 5; Table 8, PDF pp. 3–4, 6–7, 14.

CORE is a centered aggregate, not raw accuracy

$$c_j = \frac{\mathrm{acc}_j - r_j}{1 - r_j}, \qquad \mathrm{CORE} = \frac{1}{21} \sum_{j=1}^{21} c_j$$

- 21 tasks; equal weight after centering against a task baseline r_j.
- Validation BPB: lower is better; held-out ClimbMix, about 41.9M tokens.
- CORE $+0.017$ is a score change, not raw accuracy $+1.7$ percentage points.
- No multi-seed uncertainty or significance tests are reported.

Paper, evaluation setup, PDF pp. 6–7. Task-specific baselines are not tabulated.

At 100B tokens, BoV improves aggregate quality

Metric	Baseline	BoV	Difference
BPB $\downarrow$	0.671	0.663	-0.008
CORE $\uparrow$	0.334	0.351	$+0.017$
CommonsenseQA	0.437	0.523	$+0.086$
SQuAD	0.506	0.531	$+0.025$
CoQA	0.342	0.372	$+0.030$

Reported training FLOPs: $4.78 \times 10^{20} \rightarrow 4.66 \times 10^{20}$, approximately 2.51% lower.

Paper, Table 2 and training setup, PDF p. 6. Task entries reproduce reported scores.

Aggregate gains hide substantial task regressions

Task	Baseline	BoV	Difference
Repeat-Copy	0.000	0.094	+0.094
Dyck	0.087	0.154	+0.067
LSAT-AR	0.300	0.187	−0.113
CS Algorithms	0.442	0.414	−0.028
Winograd	0.766	0.740	−0.026

16 improvements and 5 regressions across 21 tasks. Symbolic reasoning does not improve uniformly.

Paper, Table 6, PDF p. 13; 780M-scale, 100B tokens.

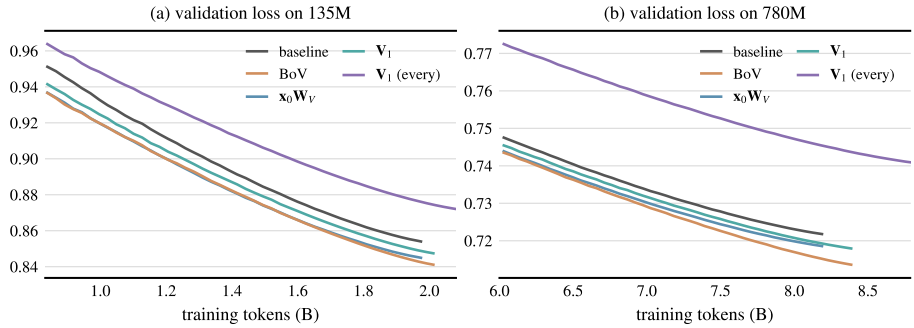
At equal FLOPs, BoV also sees more tokens

Value variant	BPB ↓	CORE ↑
Standard attention	0.722	0.260
BoV	**0.714**	**0.272**
$\mathbf{x}_0\mathbf{W}_V$, last third	0.719	0.259
$\mathbf{V}_1$, last third	0.718	0.242
$\mathbf{V}_1$, all layers	0.741	0.210

BoV: 8.39B versus baseline 8.19B tokens (about 2.44% more). Parameterization, optimization, and data exposure are not fully disentangled.

Paper, Table 3 and Sec. 5.2, PDF p. 7; baseline token count: Table 8, p. 14.

Training curves support both tested model scales



- Axes: training tokens versus validation BPB.
- Equal FLOPs can produce different final token counts.
- These curves do not measure wall-clock training speed.

Paper, Fig. 4, PDF p. 6.

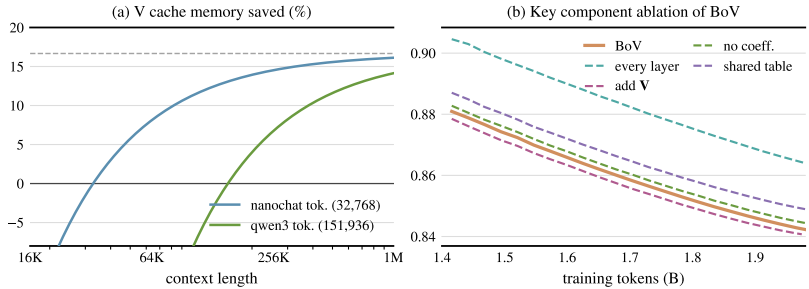
The additive competitor is close, with mixed wins

Metric	Additive $\mathbf{V} + \mathbf{E}_v[i]$	BoV
BPB ↓	**0.712**	0.714
CORE ↑	0.268	**0.272**
SQuAD	**0.439**	0.416
BoolQ	0.594	**0.635**
Winograd	**0.692**	0.645

- Across 21 tasks: BoV wins 10, loses 8, ties 3.
- The additive baseline also differs in layer selection and bounded, input-dependent gating.

Paper, Sec. 5.2, Table 7, PDF pp. 7, 13. Counts computed from the table.

Layer-specific tables and learned scales matter



- Right panel: applying BoV to every layer hurts most.
- Sharing tables or fixing the scale also increases loss.
- Retaining contextual V gives only a small loss reduction.

Paper, Sec. 5.3 and Fig. 5(b), PDF pp. 7–8. Exact endpoints and uncertainty are not tabulated.

Sparse access still adds stored parameters

$$\begin{aligned}
\text{Tables:} &\quad 8 \times 32768 \times 1536 = 402.7\text{M elements} \\
\text{Removed projections:} &\quad 8 \times 1536^2 = 18.9\text{M elements} \\
\text{Net addition:} &\quad \approx 383.8\text{M parameters}
\end{aligned}$$

- Derived for the 24-layer setting; excludes a few scalars.
- bf16 tables alone occupy 768 MiB.
- The 780M scale label is not an equal-total-parameter comparison.

Reading derivation from Sec. 4 and Table 8, PDF pp. 5, 14; not a reported memory measurement.

Evidence is promising within a narrow tested scope

- Established: useful deep-layer value replacement at 135M/780M scales.
- Not established: larger-model scaling or long-context task quality.
- Not measured: decode throughput, offload latency, and full training-memory tradeoffs.
- Next tests: matched parameters/tokens/FLOPs, multiple seeds, task regressions, and serving benchmarks.

Paper, Limitations, PDF p. 9. Proposed tests are reading analysis.

Discussion: what should a deep value represent?

Takeaway

Contextual routing can combine token-specific content effectively in deep layers. The value path need not mirror the query/key paths in the tested setting.

- The useful design is depth-specific, layer-specific, and trainable.
- Quality improves on average; task tradeoffs remain.
- Memory and speed claims require a concrete workload and implementation.

Paper, Secs. 3–5 and 7, PDF pp. 3–9.

Sources and reproducibility notes

Muyu He, Yuchen Liu, Qingya Huang, and Li Zhang.

Do Value Vectors in Deep Layers Need Context from the Residual Stream?
arXiv:2606.02780v5, 23 August 2026.

Author code: github.com/RiddleHe/nanochat

Read: author TeX, all substantive sections and appendices, and the matching 14-page PDF. Original figures reused. Experiments were not rerun.

Configuration caveat: Sec. 5 states matrix LR 0.02; Table 8 lists 0.0283 for 780M and describes scaling. Table 8 covers the shorter budget, not a complete 100B recipe.

All page locators use PDF file pages, starting at 1.