# STREAMOE: SCALING OVER ENCODING WITH N-GRAM STREAMS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Sparse parameter access enables large language models (LLMs) to expand their capacity with minimal computational overhead. Over Encoding (OE) uses this principle through N-gram embedding tables, introducing an additional scaling axis beyond the backbone architecture. Specifically, OE hashes local N-grams into table indices to retrieve embeddings that augment standard token representations. An open question is how to determine appropriate sizes for these N-gram tables and allocate capacity across different N-gram orders. To address this, we conduct extensive scaling measurements using StreamOE, an architecture that directly injects N-gram information into the network's Hyper-Connections (HC). Based on these experiments and statistical corpus analysis, we identify two key findings. First, useful table capacity is influenced by dataset scale, enabling N-gram slot counts calibrated on a small proxy model to transfer to larger backbones. Second, an asymmetric capacity allocation, such as the 1:6.75 reference ratio between 2-gram and 3-gram tables, can outperform conventional uniform splits under a fixed parameter budget. We further scale StreamOE to 80B-A3B and 309B-A15B Mixture-of-Experts (MoE) models, achieving substantial improvements in both average benchmark scores and long-context performance.

## 1 INTRODUCTION

Over Encoding (OE) enhances language model performance through N-gram embedding tables with little computational overhead (Huang et al., 2025a). At each token position, OE hashes local N-gram sequences into table indices and retrieves the corresponding embeddings to augment the model's representations. With recent frontier large language models (LLMs) adopting N-gram based methods to improve performance on knowledge-intensive and long-context tasks (Cheng et al., 2026), OE has emerged as a promising approach for boosting capabilities under a limited compute budget. However, its sparse access pattern raises a fundamental question: *how can we estimate an appropriate size for these N-gram tables?*

Prior studies have examined N-gram table capacity through the lenses of vocabulary scaling, hash collision analysis, and parameter allocation between memory and Mixture-of-Experts (MoE) modules (Cheng et al., 2026; Qwen Team, 2026; Liu et al., 2026). Furthermore, several recently released models allocate approximately uniform capacities across different N-gram orders (Qwen Team, 2026; Liu et al., 2026; DeepSeek-AI, 2026). A critical, yet underexplored, problem is how to relate these capacity choices to actual N-gram coverage under a specific training token budget. Specifically, the principles governing N-gram *table size selection and capacity allocation*, especially how candidate sizes transfer across varying training budgets and model scales, and how capacity should be distributed among N-gram orders with vastly different coverage profiles, remain largely unresolved.

To address these questions, we conduct extensive embedding scaling measurements based on *StreamOE*, a novel architecture variant introduced in this work that directly injects N-gram information into the network's Hyper-Connections (HC) streams. Alongside this empirical search, we perform a statistical analysis of our training dataset to quantify distinct N-gram counts. Based on these experiments, we identify two key findings:

- **N-gram table capacity is influenced by dataset scale, allowing calibration on a small proxy model to guide capacity selection for larger backbones.** Distinct 2-gram counts in our corpus



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

grow more slowly than the processed token volume. Increasing the training budget from 169B to 500B tokens raises the estimated distinct count from 0.602B to 0.856B (Figure 1). Using capacity sweeps on 24B-A1B at 169B tokens as the reference, we scale 2-gram slot counts in proportion to estimated coverage to predict candidate capacities for 24B-A1B and 48B-A2B at 500B tokens. On both backbones, these predictions yield fitted gain fractions close to the 80% and 90% targets (Figure 3b; Table 2). These results support using a small-model calibration to guide 2-gram capacity selection across training budgets and backbone sizes.

- **An asymmetric 1:N capacity allocation between 2-gram and 3-gram tables outperforms the conventional uniform 1:1 split.** Our measurements show that under a fixed 18.6B-parameter budget, a 1:6.75 ratio between 2-gram and 3-gram tables reduces the evaluation loss compared to a uniform split on the 24B-A1B model (Figure 3a). This situation arises because 3-grams exhibit a substantially larger number of distinct combinations within the dataset; thus, allocating proportionally more capacity to them is essential for better performance.

**Empirical insights effectively guide large-scale model design.** Guided by these measurements, we selected 2-gram and 3-gram table sizes to construct large-scale MoE models (80B and 309B parameters) and validated their effectiveness. Our evaluations demonstrate that StreamOE substantially improves overall performance compared with No-OE, increasing the average benchmark score from 59.67 to 62.23 on the 80B model and from 68.63 to 71.56 on the 309B model (see Section 5.1 and Table 3). StreamOE also enhances long-context capabilities (see Section 5.2).

Our main contributions can be summarized as follows:

- We find that N-gram table capacity is influenced by dataset scale, allowing calibration on a small proxy model to guide capacity selection for larger backbones.
- We demonstrate that an asymmetric capacity allocation can outperform conventional uniform splits under a fixed parameter budget.
- Guided by these measurements, we scaled StreamOE to 80B-A3B and 309B-A15B MoE models, achieving substantial improvements in both average benchmark scores and long-context capabilities.

## 2 Related Works

**Residual Architectures.** Integrating memory also depends on how the backbone carries and mixes representations. Hyper-Connections provide multiple residual streams (Zhu et al., 2025), while mHC and mHC-lite constrain how those streams are mixed (Xie et al., 2026; Yang & Gao, 2026). Engram already uses this multi-branch setting, sharing the embedding table and value projection while keeping branch-specific gates (Cheng et al., 2026). StreamOE uses the stream dimension to keep vocabulary, 2-gram, and 3-gram inputs separate.

**Parametric Memory and Sparse Lookup.** Additional memory can supply a language model with retrieved text, as in RETRO (Borgeaud et al., 2022), or with learned parameters. Sparse memory layers follow the latter approach, adding capacity while reading only a small subset of entries for each token. Product-Key Memory (PKM) enables efficient retrieval from large trainable key-value tables (Lample et al., 2019); Memory Layers at Scale and UltraMem examine how such stores scale and how their access costs can be reduced (Berges et al., 2025; Huang et al., 2025b).

How the memory is addressed determines which parameters a token can use. Hash Layers route token identities to FFN experts through fixed hashes (Roller et al., 2021), and Baykal et al. (2023) compare token-ID lookup with locality-sensitive hashing and learned routing. When FFNs take fixed token embeddings as inputs, their outputs can also be precomputed for inference. Mixture of Lookup Experts (MoLE) applies this to its experts (Jie et al., 2025), while MemoryLLM separates FFNs from self-attention to make their outputs token-specific (Jaiswal et al., 2026). MeKi learns token-level memory with additional transformations during training, then precomputes the resulting lookup representations for inference (Ding et al., 2026). Lookup can also replace only part of the computation: STEM substitutes the FFN up-projection while retaining dense gate and down projections (Sadhukhan et al., 2026). $L^3$ keeps contextual processing in the aggregation step by attending over token-selected embeddings, with data-derived per-token embedding allocations (Tseng & De Sa, 2026). JTok and JTok-M use token-indexed modulation vectors to augment Transformer

layers. Their scaling experiments examine the effect of increasing embedding parameters (Yang et al., 2026).

**N-Gram Embeddings and Conditional Memory.** Local context can enter the lookup key itself through N-grams. N-Grammer introduces latent N-gram representations (Roy et al., 2022), while Over-Tokenized Transformer enlarges the input vocabulary independently of the output vocabulary (Huang et al., 2025a). SCONE applies the separation between training computation and inference lookup to frequent N-grams: a separate embedding model learns their representations, which are then precomputed and offloaded for inference (Yu et al., 2025).

N-gram memories can also be integrated inside the backbone. Engram combines hashed N-gram lookup with contextual gating and studies how memory and MoE experts should share a parameter budget (Cheng et al., 2026). Related designs appear in DeepSeek-V4.1-Flash, Qwen3.8-Flash-Next, and LongCat-Flash-Lite (DeepSeek-AI, 2026; Qwen Team, 2026; Liu et al., 2026). These studies inform capacity selection under different constraints: Qwen varies vocabulary size under different parameter budgets, while LongCat analyzes hash collisions and backbone effects (Qwen Team, 2026; Liu et al., 2026). In a parameter-matched study with a 185M backbone trained on 100M tokens, Lin (2026) finds no consistent validation-loss gain from collision-free lookup for frequent N-grams. We examine how N-gram coverage can guide candidate slot counts across training budgets and backbone sizes, and whether individual orders' capacity needs can guide joint 2-gram/3-gram allocation at fixed total capacity.

## 3 StreamOE

Before studying N-gram table scaling, we introduce StreamOE, an OE variant that assigns vocabulary and N-gram embeddings to separate residual streams in Hyper-Connections. We use StreamOE throughout the scaling experiments in Section 4.

Let $\mathbf{z} = (z_1, \ldots, z_L) \in \mathcal{V}^L$ denote a token id sequence of length $L$, where $\mathcal{V}$ represents the vocabulary. The corresponding vocabulary embedding sequence is given by $\mathbf{e}_{\text{vocab}} = E_{\text{vocab}}(\mathbf{z}) \in \mathbb{R}^{L\times C}$. In standard HC integration, this identical vocabulary embedding is duplicated across all $n$ streams at the input layer. Following the original HC notation, the resulting stream is formulated as:

$$x_0^{\text{HC}} = \text{Repeat}_{\text{stream},n}(\mathbf{e}_{\text{vocab}}) \in \mathbb{R}^{L\times n\times C}. \tag{1}$$

All streams therefore enter the first block with the same vocabulary representation.

To address this limitation, StreamOE injects explicit N-gram information at the input rather than uniformly populating every stream with the same vocabulary embedding. Let $\mathbf{e}_2, \mathbf{e}_3 \in \mathbb{R}^{L\times C}$ denote the 2-gram and 3-gram representations, respectively. Assuming a configuration with four residual streams, the initialization in StreamOE is defined as:

$$x_0^{\text{StreamOE}} = \text{Stack}_{\text{stream}}(\mathbf{e}_{\text{vocab}},\, \mathbf{e}_{\text{vocab}},\, \mathbf{e}_2,\, \mathbf{e}_3) \in \mathbb{R}^{L\times 4\times C}. \tag{2}$$

At initialization, StreamOE allocates vocabulary, 2-gram, and 3-gram embeddings to distinct streams, enabling HC to assign different weights to their contributions prior to aggregation. In contrast, averaging these embeddings before replication would populate every stream with an identical mixture, restricting the initial HC mapping to reweighting the combined representation as a whole.

StreamOE can also be inserted at intermediate layers. In this layer-wise variant, we add the 2-gram and 3-gram representations directly to the last two residual streams, respectively.

## 4 Scaling Analysis

Under a fixed training token budget, the total number of N-gram tokens remains exactly equivalent across different orders. However, the diversity of covered unique N-grams and their frequency distributions vary significantly. Whether these differences translate to distinct capacity requirements must be examined independently. Scaling the tables for different N-grams simultaneously with a fixed ratio would only reveal their joint effect on performance, obfuscating the specific capacity demands of each individual order. Therefore, using 2-grams and 3-grams as examples, we first

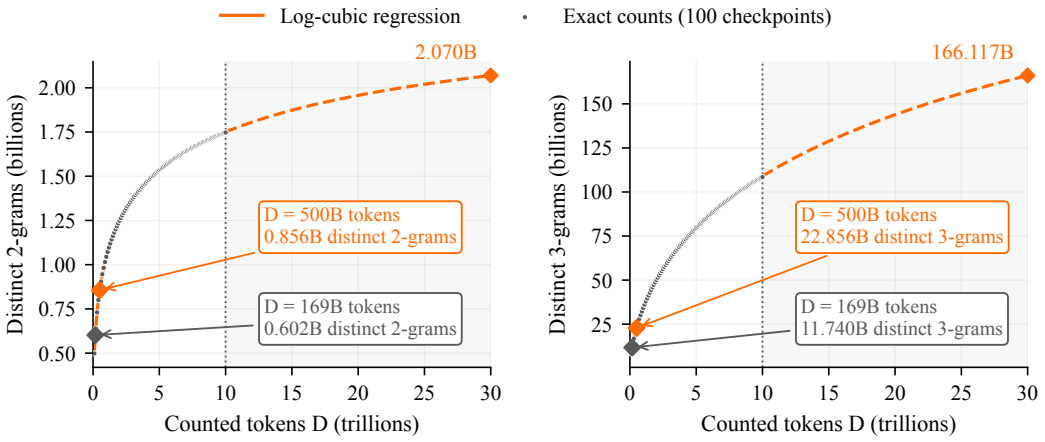



Figure 1: Distinct 2-gram and 3-gram counts with log-cubic fits up to 10T tokens and extrapolations to 30T. Dots show exact counts. Diamonds at 169B and 500B tokens mark fitted values. Within the observed range, 3-grams exhibit substantially larger distinct counts and more sustained growth than 2-grams, motivating order-specific capacity analysis.

investigate their individual capacity responses before examining the optimal capacity allocation for their joint integration.

We hypothesize that two main factors should guide the selection of table sizes: the N-gram coverage in the training data, and the trade-off between collision reduction and storage cost. To explore this, we first analyze distinct N-gram counts in the training dataset. Next, we run capacity experiments on small models for each N-gram order separately to find candidate configurations. Specifically, our 2-gram experiments cover various token budgets and backbone sizes, while our 3-gram experiments focus on the capacity upper bound. Finally, we evaluate if these capacity estimates generalize across different settings. We also verify whether the allocation ratios derived from individual N-gram orders remain effective in the joint N-gram model.

**Experimental setup.** We use 24B-A1B and 48B-A2B MoE backbones based on HySparse2 (Wei et al., 2026), with configurations summarized in Table 1. We train 24B-A1B for 169B and 500B tokens, and 48B-A2B for 500B tokens. The 500B tokens configurations use a 32k context and cosine learning-rate decay. We optimize the backbones with Muown (Lion et al., 2026) and Adam, and the StreamOE embedding tables with Sparse Adam at three times the default Adam learning rate. The default Adam peak learning rates are $6.08 \times 10^{-3}$ and $6.57 \times 10^{-3}$ for 24B-A1B and 48B-A2B, respectively.

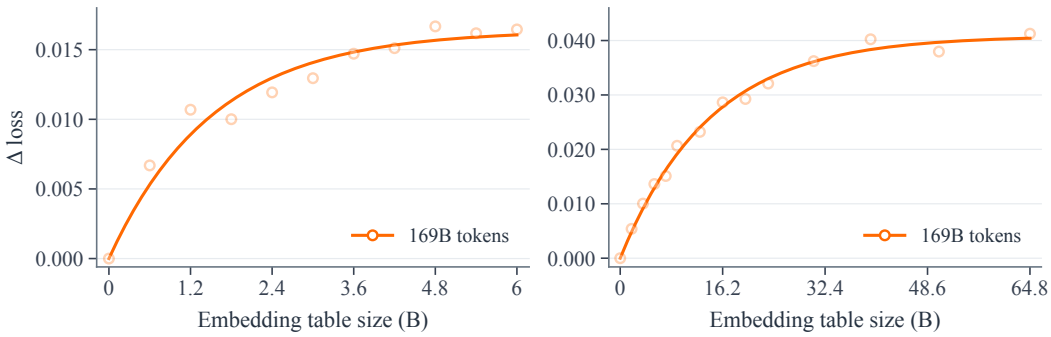
Table 1: Backbone configurations. #Full denotes the number of full-attention layers.

| Model | Layers | Hidden size | #Full | Heads (Q/KV) | Head dim. (QK/V) |
|---|---|---|---|---|---|
| 24B-A1B | 12 | 1,408 | 1 | 40/1 | 128/128 |
| 48B-A2B | 15 | 1,792 | 2 | 56/1 | 128/128 |

### 4.1 N-Gram Statistics

Using a vocabulary-compressed Qwen2.5 tokenizer, we tracked the cumulative unique 2-grams and 3-grams across a 10T tokens training dataset. Figure 1 shows 100 observed data points from 0.1T to 10T tokens, alongside a log-cubic regression curve extrapolating up to 30T tokens.

At 10T tokens, unique 3-grams ($1.09 \times 10^{11}$) outnumber 2-grams ($1.75 \times 10^{9}$) by roughly 62 times. While the discovery rate of new N-grams decelerates for both, 2-grams plateau much earlier, whereas 3-grams maintain noticeable growth throughout the observation window.




(a) 2-gram table only

(b) 3-gram table only

Figure 2: Capacity scaling of (a) 2-gram and (b) 3-gram tables on 24B-A1B trained for 169B tokens. Loss reduction is defined as $\Delta\mathcal{L} = \mathcal{L}_{\text{baseline}} - \mathcal{L}_{\text{OE}}$. Circles show observations. Solid curves show exponential fits. Both orders exhibit diminishing returns as table capacity increases.

These statistics guide our order-specific capacity selection. Since the number of unique N-grams in a given dataset is bounded, expanding table capacity only mitigates hash collisions without increasing intrinsic data diversity. *These observations motivate analyzing each N-gram order separately and testing whether data coverage limits useful table capacity.*

### 4.2 Capacity Scaling and Generalization

We independently vary the 2-gram and 3-gram table capacities on 24B-A1B trained for 169B tokens. Figure 2 shows diminishing returns at different capacity scales. The 2-gram gain changes little between 4.8B and 6B table parameters, whereas 3-gram gains extend to much larger tables. Even for 3-grams, increasing capacity from 39.6B to 64.8B parameters yields only about 0.0010 additional loss reduction. These trends support selecting capacity separately for each order.

Combined with the N-gram statistics in the previous section, we analyze the choice of table capacity from two aspects: data coverage and hash collisions. The number of unique N-grams in the training data reflects the scale of patterns that need to be accommodated. Under a fixed hashing scheme, expanding the table capacity can usually reduce the situation where different N-grams share table entries, while increasing storage overhead. Therefore, we pay attention to the position on the capacity curve where the performance gain begins to slow down significantly, and select a configuration that balances model performance and storage cost.

We next examine how the 2-gram capacity response changes with training budget and backbone size. Figure 3b compares 24B-A1B trained for 169B and 500B tokens with 48B-A2B trained for 500B tokens. In each setting, loss reduction is measured relative to the corresponding zero-table baseline.

We propose predicting N-gram table slots by scaling a reference configuration with data coverage. The working hypothesis is that, for a fixed N-gram order, comparable relative gains correspond to similar ratios of unique N-grams to slots across training budgets and backbone sizes. Let $\widehat{U}_N(T)$ denote the estimated number of distinct $N$-grams covered by $T$ training tokens. Given a reference training budget $T_{\text{ref}}$ and a slot count $S_N^{\text{ref}}$ selected on a small model, we calibrate the reference load ratio as

$$K_N^{\text{ref}} = \frac{\widehat{U}_N(T_{\text{ref}})}{S_N^{\text{ref}}}. \tag{3}$$

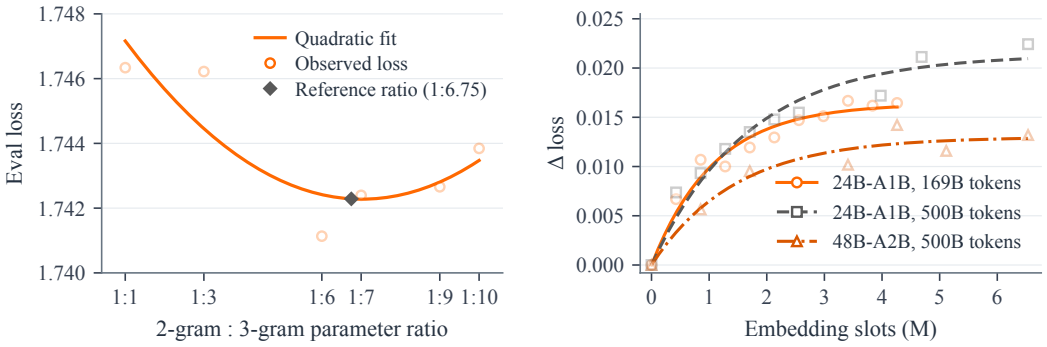
Holding this ratio fixed gives the target slot prediction

$$\widehat{S}_N(T) = \frac{\widehat{U}_N(T)}{K_N^{\text{ref}}} = S_N^{\text{ref}} \frac{\widehat{U}_N(T)}{\widehat{U}_N(T_{\text{ref}})}. \tag{4}$$

Each N-gram order is calibrated separately. After calibration, target predictions require only $K_N^{\text{ref}}$ and the estimated N-gram coverage at the target training budget.

Table 2: Predicted 2-gram slots and fitted gain fractions at 500B tokens for 80% and 90% gain targets. Fractions are normalized by each setting's fitted plateau gain.

| Target gain fraction | Backbone | Predicted slots (M) | Fitted gain fraction | Gap to target (percentage points) |
|---|---|---|---|---|
| 80% | 24B-A1B | 2.487 | 77.35% | −2.65 |
| 80% | 48B-A2B | 2.487 | 82.05% | +2.05 |
| 90% | 24B-A1B | 3.559 | 88.05% | −1.95 |
| 90% | 48B-A2B | 3.559 | 91.43% | +1.43 |




(a) N-gram allocation

(b) 2-gram capacity scaling

Figure 3: (a) Under a fixed 18.6B-parameter OE budget on 24B-A1B, the 2-gram:3-gram split of 1:6 gives the lowest observed evaluation loss. The diamond marks the 1:6.75 single-order reference ratio on the quadratic fit. (b) Scaling 2-gram slots with N-gram coverage yields comparable fractions of the fitted plateau gains across training budgets and backbone sizes (Table 2). Markers show loss reductions ($\Delta\mathcal{L}$) relative to matching zero-table baselines; curves show exponential fits.

We select reference capacities in the diminishing-returns region of the fitted gain curve, balancing performance gains against table storage cost. We use 80% and 90% of the fitted plateau gain as two representative operating points for this trade-off. On 24B-A1B trained for 169B tokens, we calibrate reference load ratios of 344.17 and 240.57 for the two targets, respectively. We hold each ratio fixed and use the 500B-token coverage estimate to predict slots for both 24B-A1B and 48B-A2B (Table 2).

Across both backbones, the fitted gain fractions align closely with the intended targets, deviating by at most 2.7 percentage points for the 80% target and 2.0 percentage points for the 90% target. These results support our working hypothesis that comparable relative gains correspond to similar coverage-to-slot ratios. For each gain target, the reference calibration transfers across the tested training budgets and backbone sizes to identify candidate 2-gram capacities.

### 4.3 Ratio of N-gram Orders

The preceding analysis shows that 2-gram and 3-gram tables respond to capacity expansion at different scales. We next examine how these differences inform their allocation under a fixed total budget. We select the 2.4B parameter 2-gram table and the 16.2B parameter 3-gram table as illustrative reference configurations, considering both loss reduction and table storage in the single-order sweeps. Their ratio of 1:6.75 provides a starting point for the joint-allocation study. On 24B-A1B, we fix the total OE budget at 18.6B parameters and compare six 2-gram-to-3-gram ratios from 1:1 to 1:10. Figure 3a shows that evaluation loss first decreases and then increases as more capacity is assigned to 3-grams. The best observed ratio, 1:6, reduces loss by approximately 0.0052 relative to uniform allocation, while the quadratic fit places its minimum near 1:7.02. *Under the tested budget, asymmetric allocation outperforms a uniform split, and the single-order reference ratio provides a useful starting point for joint capacity selection.*

# 5 EVALUATION

The preceding scaling experiments use StreamOE to incorporate N-gram embeddings. We now show that StreamOE improves overall benchmark and long-context performance on larger MoE backbones, and examine the effect of N-gram embedding table quantization.

**Experimental setup.** We evaluate StreamOE on an 80B-A3B MoE backbone based on HySparse (Gao et al., 2026) and a 309B-A15B MoE backbone based on MiMo-V2-Flash (LLM-Core Xiaomi, 2026). The former combines full and sparse attention with cross-layer KV-cache sharing, while the latter interleaves sliding-window and global attention. Both backbones are pretrained at 32k; the 80B-A3B long-context comparison uses checkpoints after 256k-context extension. These settings assess StreamOE's applicability across model scales and attention architectures. Vanilla OE averages the vocabulary and N-gram representations before replicating the fused representation across HC streams. Comparing it with StreamOE on 80B-A3B evaluates whether separate stream inputs improve performance over this fused initialization.

**Evaluation metrics.** We use the same evaluation metrics as HySparse2 (Wei et al., 2026). Repo Code PPL is an internal benchmark that reports byte perplexity on long code repositories with cross-file dependencies. Agent PPL is an internal benchmark built from multi-turn agent trajectories and reports byte perplexity on their reasoning, tool-call, and response segments. LongPPL (Fang et al., 2025) measures perplexity on key tokens requiring long-range context, using examples from agent trajectories and long-context tasks.

Table 3: Benchmark scores (%) and byte perplexity at the final 32k-pretraining checkpoints. Average is the unweighted mean of all 17 benchmarks. Bold marks the best result within each backbone; arrows indicate metric direction.

| | 80B-A3B | | | 309B-A15B | |
|---|---|---|---|---|---|
| Benchmark | No-OE | Vanilla OE | StreamOE | No-OE | StreamOE |
| *Reasoning and knowledge* ↑ | | | | | |
| ARC-C | 80.63 | **82.94** | 82.51 | 84.98 | **86.09** |
| HellaSwag | 79.74 | 80.22 | **80.68** | 82.85 | **83.33** |
| WinoGrande | 73.56 | **74.51** | 73.95 | 77.03 | **78.30** |
| BBH (CoT) | 61.79 | **64.34** | 63.57 | 80.29 | **80.76** |
| C-Eval | 68.57 | 70.13 | **70.73** | 72.36 | **76.45** |
| CMMLU | 70.62 | 72.17 | **72.62** | 73.90 | **76.27** |
| MMLU | 64.81 | **66.36** | 65.92 | 68.94 | **70.28** |
| MMLU-Pro | 37.21 | 37.82 | **37.85** | 44.89 | **45.20** |
| MMLU-Redux | 68.24 | 69.36 | **70.09** | 72.16 | **73.19** |
| DROP | 62.32 | 62.61 | **66.47** | 70.37 | **71.57** |
| GSM8K | 66.03 | 65.13 | **66.64** | 76.80 | **79.08** |
| MATH | 36.62 | 34.98 | **37.24** | 52.40 | **53.10** |
| TriviaQA | 61.80 | 63.04 | **63.45** | 67.19 | **69.00** |
| *Coding* ↑ | | | | | |
| HumanEval | 36.59 | 44.51 | **46.34** | **71.34** | **71.34** |
| HumanEval+ | 29.27 | **37.80** | **37.80** | 62.20 | **65.85** |
| MBPP | 64.55 | **67.20** | 65.87 | 59.26 | **76.19** |
| MBPP+ | 52.12 | **56.61** | 56.08 | 49.74 | **60.58** |
| Average ↑ | 59.67 | 61.75 | **62.23** | 68.63 | **71.56** |
| *Language modeling* ↓ | | | | | |
| Pile-10K byte PPL | 1.5190 | 1.5155 | **1.5134** | 1.4921 | **1.4866** |
| Repo Code byte PPL | 1.1575 | 1.1564 | **1.1556** | 1.1309 | **1.1279** |

## 5.1 STREAMOE OVERALL COMPARISON

We compare No-OE, Vanilla OE, and StreamOE on 80B-A3B after 500B training tokens, and No-OE with StreamOE on 309B-A15B after 752B tokens. The 309B-A15B variant additionally applies OE at layer 41. Table 3 reports the 17 benchmark scores and byte perplexity on Pile-10K and repository code. StreamOE increases the average benchmark score from 59.67 to 62.23 on 80B-A3B and from 68.63 to 71.56 on 309B-A15B, with lower byte perplexity on both datasets. On 80B-A3B,

its average exceeds Vanilla OE's 61.75, although Vanilla OE leads on six individual benchmarks. On 309B-A15B, StreamOE improves 16 of the 17 benchmarks and matches No-OE on HumanEval.

On 80B-A3B, StreamOE changes only how vocabulary and N-gram embeddings enter the HC streams, without adding parameters relative to Vanilla OE. The standard benchmark average improves by 0.48 percentage points, while RULER-v2 improves by 5.91 percentage points at 256k after context extension (Section 5.2). Separate inputs allow HC to modulate vocabulary and N-gram contributions independently before their initial aggregation. The larger RULER-v2 gain suggests that keeping these features separately accessible at the input is particularly useful for long-context retrieval.

### 5.2 Long-Context Evaluation

We extend the context length of both 80B-A3B Vanilla OE and StreamOE from 32k to 256k and continue pretraining for approximately 100B additional tokens. Figure 4(a) reports RULER-v2 scores from 16k to 256k. StreamOE scores higher at all five lengths, reaching 35.88% at 256k versus 29.97% for Vanilla OE.

On 309B-A15B, we compare No-OE and StreamOE at the final 32k pretraining checkpoints. Figure 4(b–c) shows lower LongPPL and slightly lower agent byte perplexity for StreamOE across all reported length bins.

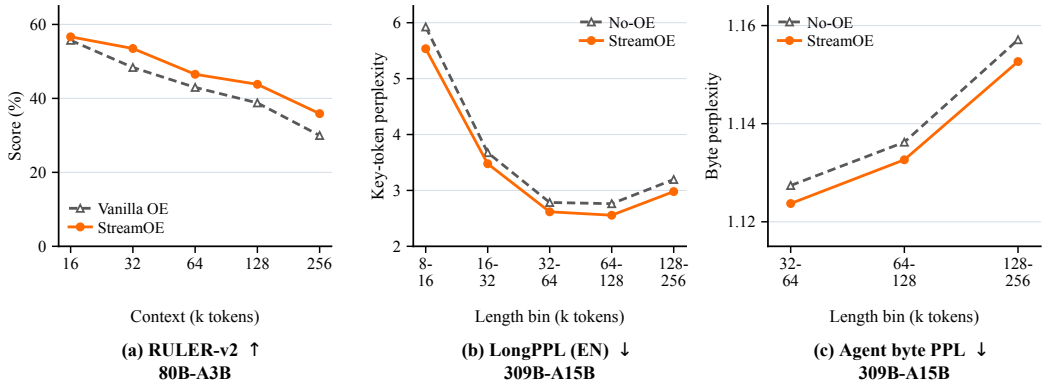



Figure 4: Long-context comparisons. (a) 80B-A3B RULER-v2 after 256k-context extension: Vanilla OE versus StreamOE. (b–c) 309B-A15B English LongPPL and agent byte PPL after 32k-context pretraining: No-OE versus StreamOE. The x-axes show fixed contexts in (a) and length bins in (b–c); arrows indicate metric direction.

Table 4: Effect of OE-table precision on 80B-A3B. Higher benchmark scores (%) and lower byte perplexity are better. Average is the unweighted mean of all 17 benchmarks. Bold marks the better result.

| Metric | FP8 | MXFP4 | Metric | FP8 | MXFP4 |
|---|---|---|---|---|---|
| ARC-C | **82.51** | 81.06 | GSM8K | **67.93** | 65.66 |
| HellaSwag | **80.21** | 80.14 | MATH | **37.26** | 37.08 |
| WinoGrande | **74.43** | 74.19 | TriviaQA | 63.26 | **63.58** |
| BBH (CoT) | 63.91 | **64.66** | HumanEval | **47.56** | 44.51 |
| C-Eval | **73.11** | 71.17 | HumanEval+ | 36.59 | **37.80** |
| CMMLU | **72.68** | 72.62 | MBPP | 64.81 | **67.46** |
| MMLU | **65.95** | 65.82 | MBPP+ | 53.44 | **55.03** |
| MMLU-Pro | **37.72** | 37.38 | Average ↑ | **62.16** | 61.80 |
| MMLU-Redux | **70.32** | 68.71 | Pile-10K byte PPL ↓ | 1.5143 | **1.5139** |
| DROP | **65.02** | 63.72 | Repo Code byte PPL ↓ | **1.1555** | 1.1557 |

### 5.3 StreamOE Table Quantization

To reduce table storage, we evaluate FP8 (E4M3) and MXFP4 quantization of a 6.2B-parameter StreamOE table on the 80B-A3B backbone. We train two models from scratch with FP8 and MXFP4 StreamOE embedding tables, respectively, and compare their final 32k-pretraining checkpoints under the same FP8 inference setting. Table 4 reports all 17 benchmark scores, their average, and byte perplexity. MXFP4 reaches an average score of 61.80, 0.36 percentage points below FP8, and scores higher on five individual benchmarks. Byte perplexity remains close on both datasets. Moving from 8-bit to 4-bit weights halves raw weight storage, excluding scale metadata.

## 6 Conclusion

We studied the capacity scaling of N-gram embedding tables using StreamOE. Under fixed training budgets, table expansion yields diminishing returns, and unequal allocation across N-gram orders outperforms a uniform split in our experiments. We use distinct N-gram coverage and a load ratio calibrated on a reference model to identify candidate slot counts across training budgets and backbone sizes. At the predicted 2-gram capacities, the target fits reach 77.35% and 82.05% of their plateau gains, close to the 80% target. Experiments on 80B-A3B and 309B-A15B backbones, together with table quantization results, support StreamOE's practical use. Our method provides a starting point for capacity selection and focused validation on the target model.

### AI use statement

The authors utilized generative AI tools to assist with language refinement, structural editing, and general manuscript preparation. All technical claims, equations, citations, experimental results, and final wording were rigorously verified and approved by the authors, who take full responsibility for the integrity and contents of the final manuscript.

### Reproducibility statement

Section 3 describes StreamOE's input initialization and layer-wise variant. Section 4 reports the scaling experiment settings, including optimizers and learning rates, and describes the N-gram statistics and capacity calibration procedure. Backbone configurations are summarized in Table 1. Section 5 reports the evaluation settings, metric definitions, and embedding table quantization experiments.

## References

Cenk Baykal, Dylan J. Cutler, Nishanth Dikkala, Nikhil Ghosh, Rina Panigrahy, and Xin Wang. The power of external memory in increasing predictive model capacity. *arXiv preprint arXiv:2302.00003*, 2023.

Vincent-Pierre Berges, Barlas Oğuz, Daniel Haziza, Wen-tau Yih, Luke Zettlemoyer, and Gargi Ghosh. Memory layers at scale. In *International Conference on Machine Learning*, volume 267, pp. 3831–3842, 2025.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, volume 162, pp. 2206–2240, 2022.

Xin Cheng, Rui Tian, Wangding Zeng, Damai Dai, Qinyu Chen, Bingxuan Wang, Zhenda Xie, Kezhao Huang, Xingkai Yu, Chengqi Deng, Shangyan Zhou, Chenggang Zhao, Zhewen Hao, Yukun Li, Han Zhang, Zhengyan Zhang, Yixuan Wei, M. Y. Xu, Huishuai Zhang, Dongyan Zhao, and Wenfeng Liang. Conditional memory via scalable lookup: A new axis of sparsity for large language models. *arXiv preprint arXiv:2601.07372v2*, 2026.

DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression. *arXiv preprint arXiv:2609.19969*, 2026.

Ning Ding, Fangcheng Liu, Kyungrae Kim, Linji Hao, Kyeng-Hun Lee, Hyeonmok Ko, and Yehui Tang. MeKi: Memory-based expert knowledge injection for efficient LLM scaling. *arXiv preprint arXiv:2602.03359*, 2026.

Lizhe Fang, Yifei Wang, Zhaoyang Liu, Chenheng Zhang, Stefanie Jegelka, Jinyang Gao, Bolin Ding, and Yisen Wang. What is wrong with perplexity for long-context language modeling? In *International Conference on Learning Representations*, 2025.

Yizhao Gao, Jianyu Wei, Qihao Zhang, Yu Cheng, Shimao Chen, Zhengju Tang, Zihan Jiang, Yifan Song, Hailin Zhang, Liang Zhao, Bo Yang, Gang Wang, Shijie Cao, and Fuli Luo. HySparse: A hybrid sparse attention architecture with oracle token selection and KV cache sharing. *arXiv preprint arXiv:2602.03560*, 2026.

Hongzhi Huang, Defa Zhu, Banggu Wu, Yutao Zeng, Ya Wang, Qiyang Min, and Xun Zhou. Over-Tokenized Transformer: Vocabulary is generally worth scaling. In *International Conference on Machine Learning*, volume 267, pp. 26261–26282, 2025a.

Zihao Huang, Qiyang Min, Hongzhi Huang, Defa Zhu, Yutao Zeng, Ran Guo, and Xun Zhou. Ultra-sparse memory network. In *International Conference on Learning Representations*, 2025b.

Ajay Jaiswal, Lauren Hannah, Han-Byul Kim, Duc Hoang, Arnav Kundu, Mehrdad Farajtabar, and Minsik Cho. MemoryLLM: Plug-n-play interpretable feed-forward memory for transformers. In *International Conference on Machine Learning*, 2026.

Shibo Jie, Yehui Tang, Kai Han, Yitong Li, Duyu Tang, Zhi-Hong Deng, and Yunhe Wang. Mixture of lookup experts. In *International Conference on Machine Learning*, volume 267, pp. 27929–27940, 2025.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Large memory layers with product keys. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Tao Lin. A collision-free hot-tier extension for Engram-style conditional memory: A controlled study of training dynamics. *arXiv preprint arXiv:2601.16531v2*, 2026.

Kai Lion, Florian Hübler, Bingcong Li, Antonio Orvieto, and Niao He. Muown: Row-norm control for Muon optimization. *arXiv preprint arXiv:2605.10797*, 2026.

Hong Liu, Jiaqi Zhang, Chao Wang, Xing Hu, Linkun Lyu, Jiaqi Sun, Xurui Yang, Bo Wang, Fengcun Li, Yulei Qian, Lingtong Si, Yerui Sun, Rumei Li, Peng Pei, Yuchen Xie, and Xunliang Cai. Scaling embeddings outperforms scaling experts in language models. *arXiv preprint arXiv:2601.21204*, 2026.

LLM-Core Xiaomi. MiMo-V2-Flash technical report. *arXiv preprint arXiv:2601.02780*, 2026.

Qwen Team. On the design of Qwen3.8-Next architecture: Evaluation, efficiency, and training stability. *arXiv preprint arXiv:2608.30320*, 2026.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. Hash layers for large sparse models. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17555–17566, 2021.

Aurko Roy, Rohan Anil, Guangda Lai, Benjamin Lee, Jeffrey Zhao, Shuyuan Zhang, Shibo Wang, Ye Zhang, Shen Wu, Rigel Swavely, Tao (Alex) Yu, Phuong Dao, Christopher Fifty, Zhifeng Chen, and Yonghui Wu. N-Grammer: Augmenting transformers with latent n-grams. *arXiv preprint arXiv:2207.06366*, 2022.

Ranajoy Sadhukhan, Sheng Cao, Harry Dong, Changsheng Zhao, Attiano Purpura-Pontoniere, Yuandong Tian, Zechun Liu, and Beidi Chen. STEM: Scaling transformers with embedding modules. In *International Conference on Learning Representations*, 2026.

Albert Tseng and Christopher De Sa. $L^3$: Large lookup layers. In *International Conference on Machine Learning*, 2026.

Jianyu Wei, Yizhao Gao, Qihao Zhang, Shimao Chen, Zhengju Tang, Yu Cheng, Shengjie Zhou, Zihan Jiang, Yifan Song, Hailin Zhang, Liang Zhao, Bo Yang, Gang Wang, Shijie Cao, and Fuli Luo. HySparse2: Hybrid sparse attention with two-level KV sharing. *arXiv preprint arXiv:2609.26368*, 2026.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Kuai Yu, Liang Zhao, Shangyan Zhou, Zhean Xu, Zhengyan Zhang, Wangding Zeng, Shengding Hu, Yuqing Wang, Jingyang Yuan, Lean Wang, and Wenfeng Liang. mHC: Manifold-constrained hyper-connections. In *International Conference on Machine Learning*, 2026.

Yebin Yang, Huaijin Wu, Fu Guo, Lin Yao, Xiaohan Qin, Jingzhi Wang, Debing Zhang, and Junchi Yan. JTok: On token embedding as another axis of scaling law via joint token self-modulation. *arXiv preprint arXiv:2602.00800*, 2026.

Yongyi Yang and Jianyang Gao. mHC-lite: You don't need 20 Sinkhorn-Knopp iterations. *arXiv preprint arXiv:2601.05732*, 2026.

Da Yu, Edith Cohen, Badih Ghazi, Yangsibo Huang, Pritish Kamath, Ravi Kumar, Daogao Liu, and Chiyuan Zhang. Scaling embedding layers in language models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 31969–31995, 2025.

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