KNOWLEDGE CAPACITY BEYOND INDEPENDENT FACTS: SHARED STRUCTURE, ABSTRACTION AND FREQUENCY

Anonymous authors
Paper under double-blind review

ABSTRACT

Empirical studies find that Transformers store 2–4 bits per parameter when trained on uniform random data, but this linear capacity scaling has been established only for independent facts that are spelled out token by token in the training sequences. We ask how knowledge capacity changes when data share compositional structure, when knowledge is latent and expressed through many different surface sequences, and when items occur with unequal frequencies. Using the Random Hierarchy Model, we introduce a measure of stored knowledge that credits a model for what it has learned about a latent knowledge piece rather than about any particular sequence. For verbatim memorization of structured sequences, we find a transition from learning the shared grammar to a regime in which stored information grows linearly with parameter count, with an offset reflecting incomplete grammar learning. The capacity ratio reaches 2.8–3.0 bits per parameter after 1000 epochs and scales with the total number of training exposures C as $r \propto C^{0.128}$. For latent knowledge, stored information still grows linearly with parameter count even though no training sequence is ever repeated, showing that linear scaling is a property of memorization itself rather than a result of repeated training data. At the same exposure budget, the capacity ratio falls from 2.3 to 0.5 bits per parameter as the knowledge becomes more abstract. For unequal frequencies, capacity is allocated to frequent items first, rare items are suppressed even at fixed exposure count, and total stored information departs from a single linear law. An associative memory optimized under frequency-weighted cross-entropy reproduces this suppression and traces it to interference between stored associations.

1 INTRODUCTION

Memory and storage capacity in neural networks have been studied from several perspectives. On the theoretical side, one line of work characterizes capacity through the VC-dimension, the largest number of inputs a network can label in every possible way, and hence a worst-case measure of how many arbitrary labels it can memorize. It is $O(N^2)$ for piecewise-polynomial activations (Goldberg & Jerrum, 1995), and $O(N \log N)$ times the network depth for ReLU networks (Bartlett et al., 2019). Another line of work constructs neural networks that can memorize near quadratically many examples in the number of parameters (Vardi et al., 2021; Kim et al., 2023; Kajitsuka & Sato, 2024). This parameter efficiency relies on allowing weight precision to increase with the number of memorized examples, and it is unclear whether such high-capacity solutions can be reached through gradient-based training. Neural associative memories offer a third perspective, in which capacity is set by interference between stored associations. In particular, Nichani et al. (2025) construct shallow Transformers whose storage capacity scales nearly linearly with the number of parameters, with facts stored in either attention or MLP parameters. Empirically, studies on synthetic memorization tasks observe a linear relationship between stored information and parameter count (Collins et al., 2017; Allen-Zhu & Li, 2024; Morris et al., 2025; Lu et al., 2024). However, both theoretical and empirical characterizations have largely focused on collections of unstructured, independent facts. Real-world data instead exhibits substantial shared and hierarchical structure, allowing information to be represented through reusable statistical regularities rather than independent associations. How such structure changes the effective storage capacity of neural networks remains poorly understood.

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.

In this work, we extend capacity scaling to structured and latent knowledge using the Random Hierarchy Model (RHM), where the amount and level of shared structure can be controlled precisely.

A related question is how linear memorization capacity connects to neural scaling laws. On a fixed dataset, linear growth in stored information implies a linear decrease in training loss with parameter count (Collins et al., 2017; Allen-Zhu & Li, 2024; Morris et al., 2025). By contrast, test loss in language models is observed to decay as a power law in parameters, data, and compute (Hestness et al., 2017; Kaplan et al., 2020; Hoffmann et al., 2022), typically with exponents below 0.5. A popular suggestion is that the power-law scaling originates from power laws already present in the data (Hutter, 2021; Michaud et al., 2023). Barkeshli et al. (2026) show that this is not necessary: for transformers trained to predict random walks on graphs, neural scaling laws arise even without power-law structure in the data. These observations raise the question of whether the linear capacity scaling observed in prior studies depends on their use of uniformly sampled data. Does this scaling persist when items occur with unequal frequencies, and how is capacity allocated across frequent and rare items? In Section 6, we address this question by training Transformers on random key–value associations with unequal frequencies, and use an associative-memory model to explain the resulting interference.

Our contributions are as follows.

- **Linear capacity on structured data.** Using the Random Hierarchy Model, we show that verbatim memorization of sequences with shared compositional structure exhibits a two-phase behavior: models first learn the shared grammar, after which stored information grows linearly with parameter count, with an offset reflecting incomplete learning of the shared structure. The capacity ratio is around 2–3 bits per parameter with 1000 epochs of training and scales with the total number of training exposures C as $r \propto C^{0.128}$.
- **Linear capacity on latent knowledge.** We model a piece of knowledge as a latent variable that can be expressed by many different token sequences, and introduce a measure of how much a model has learned about the latent variable itself rather than about any particular sequence. This goes beyond prior work, in which knowledge takes the form of (key, value) pairs that appear verbatim many times in the training corpus. We find that the loss on new sequences expressing the same latent variables decreases linearly with model size, even though no training sequence is ever repeated, so linear scaling is a property of memorization itself rather than of verbatim repetition. The capacity ratio decreases as the knowledge piece becomes more abstract, from 2.3 bits per parameter for verbatim memorization to 0.5 for the most abstract knowledge we study.
- **Frequency interference.** When items occur with unequal frequencies, capacity is allocated sequentially by frequency: rare items are not stored until frequent ones are nearly exhausted. Total stored information then departs from the single linear law of the uniform case. We show that a simple associative memory trained with frequency-weighted cross-entropy behaves the same way. The optimal solution stores frequent items with larger weights, and these larger weights add noise to the retrieval of rare items.

2 Latent knowledge and its measurement

2.1 Latent knowledge to surface token

We model language as a two-stage generative process. A *knowledge piece* $x \in \mathcal{V}^d$ is a vector of d features over a finite vocabulary $\mathcal{V}$, drawn uniformly from a sampling space $\Omega \subseteq \mathcal{V}^d$, and is then realized as a token sequence $t \in \mathcal{W}^T$ drawn from an emission distribution $\mathcal{Q}(\cdot \mid x)$. The same x can generate many different t. For example, take x to encode the Pythagorean theorem. Its realizations t include the formula $a^2 + b^2 = c^2$, the sentence "the square of the hypotenuse equals the sum of the squares of the other two sides", a worked problem in which a 3–4–5 triangle appears, and a derivation of the distance formula in coordinates. These strings share almost no tokens, yet they express the same knowledge.

Marginalizing over the sampling space gives the distribution over surface sequences,

$$\mathcal{M}(t) \;=\; \frac{1}{|\Omega|} \sum_{x \in \Omega} \mathcal{Q}(t \mid x). \tag{1}$$

A *knowledge set* $K = \{x^{(1)}, \ldots, x^{(n)}\}$ consists of n distinct pieces drawn from Ω. The sampling space Ω contains every piece that could be expressed, and K selects the ones that are true. For

example, Ω contains both the Pythagorean theorem and a false variant such as $a^3+b^3=c^3$, but only the former is in K. Since $\mathcal{M}$ averages over all of Ω, it generates realizations of both. Marginalizing over the knowledge set gives the distribution over realizations of K,

$$\mathcal{M}_K(t) \;=\; \frac{1}{n}\sum_{x\in K}\mathcal{Q}(t \mid x). \tag{2}$$

2.2 Stored knowledge as code-length reduction

The *training corpus* D consists of token-level realizations of the pieces in K, where each piece may appear with any number of realizations. A neural network with parameters $\theta(D)$ is trained on D, and we denote its predictive distribution by $\mathcal{M}_{\theta(D)}(t)$. By Shannon's source-coding theorem, a sequence t can be encoded in $\log_2 1/\mathcal{M}_{\theta(D)}(t)$ bits using $\mathcal{M}_{\theta(D)}$ (Cover & Thomas, 2006). We define the information the model stores about a knowledge piece $x^{(i)}$ as the number of bits $\mathcal{M}_{\theta(D)}$ saves relative to the marginal $\mathcal{M}$ in Equation (1) on fresh realizations $t\sim\mathcal{Q}(\cdot \mid x^{(i)})$. The expected code lengths under $\mathcal{M}$ and under $\mathcal{M}_{\theta(D)}$ are the cross-entropies

$$H_i(\mathcal{M}) = \mathbb{E}_{t\sim\mathcal{Q}(\cdot|x^{(i)})}\left[\log_2\frac{1}{\mathcal{M}(t)}\right], \qquad H_i(\mathcal{M}_{\theta(D)}) = \mathbb{E}_{t\sim\mathcal{Q}(\cdot|x^{(i)})}\left[\log_2\frac{1}{\mathcal{M}_{\theta(D)}(t)}\right], \tag{3}$$

and the knowledge stored about K is

$$b(\theta(D),K) = \sum_{i=1}^{n}\big[H_i(\mathcal{M}) - H_i(\mathcal{M}_{\theta(D)})\big] = \sum_{i=1}^{n}\mathbb{E}_{t\sim\mathcal{Q}(\cdot|x^{(i)})}\left[\log_2\frac{\mathcal{M}_{\theta(D)}(t)}{\mathcal{M}(t)}\right]. \tag{4}$$

Verbatim memorization is the special case in which each piece is itself a token sequence and the emission is deterministic, $t=x$. Then $\mathcal{M}(x)=1/|\Omega|$ and Equation (4) reduces to

$$b_{\text{verbatim}}(\theta(D),K) = \sum_{i=1}^{n}\log_2\big(|\Omega|\,\mathcal{M}_{\theta(D)}(x^{(i)})\big). \tag{5}$$

If the model has learned the grammar but nothing about K, then $\mathcal{M}_{\theta(D)}=\mathcal{M}$ and $b=0$. If it has not fully learned the grammar, it predicts realizations of K worse than $\mathcal{M}$ and $b<0$. The information upper bound is reached when the model has stored all of K. Such a model knows that a fresh realization comes from one of the n pieces and predicts with $\mathcal{M}_K$ in Equation (2). To evaluate b for this model, we assume that each sequence can be generated by only one knowledge piece. For a realization t of $x^{(i)}$, the sums in Equations (1) and (2) then reduce to $\mathcal{M}(t)=\mathcal{Q}(t \mid x^{(i)})/|\Omega|$ and $\mathcal{M}_K(t)=\mathcal{Q}(t \mid x^{(i)})/n$, and Equation (4) gives the maximum available knowledge in K,

$$b^{\max}(K) \;=\; \sum_{i=1}^{n}\mathbb{E}_{t\sim\mathcal{Q}(\cdot|x^{(i)})}\Big[\log_2\frac{\mathcal{Q}(t \mid x^{(i)})/n}{\mathcal{Q}(t \mid x^{(i)})/|\Omega|}\Big] \;=\; n\big(\log_2|\Omega| - \log_2 n\big). \tag{6}$$

3 Experimental Setup

3.1 Random Hierarchy Models

We consider the Random Hierarchy Model (RHM) (Cagnetta et al., 2024), a probabilistic context-free grammar that generates sequences from a hierarchy of latent variables (Figure 1a). Samples are generated top-down over L levels: the root at level $L+1$ is a class label drawn uniformly from c classes, and every symbol at level $\ell+1$ is expanded into s symbols at level ℓ by selecting uniformly among m rules associated with that symbol, over a vocabulary of size v. The vm rules within each level are distinct. Therefore different features will never generate the same lower level outcome. The leaves at level 1 are the surface sequence t of length $T=s^L$. Let Ω_ℓ denote the sampling space of level ℓ. The number of possible configurations of level ℓ is

$$|\Omega_\ell| \;=\; c\cdot m\cdot m^{s}\cdots m^{s^{L-\ell}} \;=\; c\,m^{\sum_{i=0}^{L-\ell}s^i} \;=\; c\,m^{\frac{s^{L-\ell+1}-1}{s-1}}. \tag{7}$$

The RHM allows us to study how the level of abstraction of a knowledge piece affects memoriza-

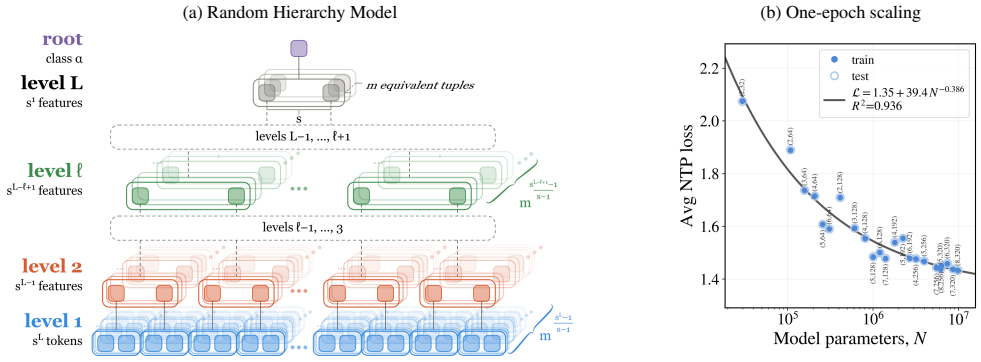


Figure 1: **The RHM generative tree and one-epoch scaling.** **(a)** Each symbol expands into s children through one of m production rules, each specifying an s-tuple over a vocabulary of size v. Level ℓ contains $s^{L-\ell+1}$ features, with s^L observed tokens at level 1. Braces indicate the number of possible level configurations for a fixed root class. **(b)** Average next-token prediction loss after one training epoch for $(v, c, m, s, L) = (64, 64, 4, 2, 6)$ and $n = 2.5 \times 10^7$.

tion. For intuition, the least abstract case is analogous to memorizing a specific poem verbatim; this corresponds to knowledge piece at level 1 of the RHM. A more abstract concept such as the Pythagorean theorem is closer to a higher-level latent state, which can be expressed through many more surface realizations. As shown in Figure 1(b), Transformers trained on the RHM reproduce the power-law scaling of language models: after a single epoch on 2.5×10^7 sequences, the test loss decays as a power law in N.

3.2 Latent memorization dataset

For each experiment, we fix a memorization level $\ell_{\text{mem}} > 1$. A knowledge piece x is the vector of the $s^{L-\ell_{\text{mem}}+1}$ symbols at level ℓ_{mem}, so its sampling space is $\Omega = \Omega_{\ell_{\text{mem}}}$ in Equation (7), and its realizations are the leaf sequences generated below it. We sample n distinct knowledge pieces,

$$K = \{x^{(1)}, \dots, x^{(n)}\} \subset \Omega_{\ell_{\text{mem}}}. \tag{8}$$

Each $x^{(i)}$ generates n_{exp} independently sampled surface realizations,

$$t_{\text{train}}^{(i,e)} \sim \mathcal{Q}(\cdot \mid x^{(i)}), \qquad e = 1, \dots, n_{\text{exp}}, \tag{9}$$

which form the training corpus

$$D = \left\{ t_{\text{train}}^{(i,e)} : i = 1, \dots, n,\ e = 1, \dots, n_{\text{exp}} \right\}. \tag{10}$$

To measure stored knowledge, we evaluate the trained model on fresh realizations of the same knowledge pieces,

$$t_{\text{test}}^{(i,j)} \sim \mathcal{Q}(\cdot \mid x^{(i)}), \qquad j = 1, \dots, n_{\text{test}}, \tag{11}$$

all distinct from the training realizations. Since the marginal RHM distribution is uniform over Ω_1, $\mathcal{M}(t) = 1/|\Omega_1|$, and Equation (4) becomes

$$\begin{aligned} b(\theta(D), K) &= n \log_2 |\Omega_1| - \frac{1}{n_{\text{test}}} \sum_{i=1}^{n} \sum_{j=1}^{n_{\text{test}}} \log_2 \frac{1}{\mathcal{M}_{\theta(D)}\left(t_{\text{test}}^{(i,j)}\right)} \\ &= n \log_2 |\Omega_1| - \mathcal{L}_{\text{CE}}^{\text{test}}, \end{aligned} \tag{12}$$

where $\mathcal{L}_{\text{CE}}^{\text{test}}$ is the test cross-entropy loss in bits, averaged over the n_{test} realizations $t_{\text{test}}^{(i,j)}$ of each knowledge piece $x^{(i)}$ and summed over the n pieces.

3.3 Verbatim memorization dataset

For verbatim memorization $\ell_{\text{mem}} = 1$, the model observes K directly and $t^{(i)} = x^{(i)}$, the training set is $D = \{x^{(i)} : i = 1, \dots, n\}$ and the model is trained on D for n_{exp} epochs. The stored

knowledge is evaluated on t_{train}, the same sequences the model is trained on:

$$b(\theta(D), K) = n \log_2 |\Omega_1| - \sum_{i=1}^{n} \log_2 \frac{1}{\mathcal{M}_{\theta(D)}\left(x^{(i)}\right)} = n \log_2 |\Omega_1| - \mathcal{L}_{\text{CE}}^{\text{train}}. \tag{13}$$

$\mathcal{L}_{\text{CE}}^{\text{train}}$ is the training cross-entropy loss in bits.

3.4 Models and training

All experiments use a decoder-only GPT-2 transformer (Radford et al., 2019) with rotary positional embeddings (Su et al., 2024), parameterized under the maximal update parametrization (μP) (Yang et al., 2022). We train with AdamW (Loshchilov & Hutter, 2019) under linear warmup followed by cosine decay, tuning the peak learning rate and weight decay for each configuration.

4 Linear capacity emerges in the over-parameterized regime

In this section, we set $\ell_{\text{mem}} = 1$ and study the verbatim memorization of RHM sequences.

Grammar learning followed by linear memorization. Figure 2 shows how stored information scales with model size on two RHMs with $(v, c, s, L) = (128, 32, 4, 4)$. The two grammars differ only in synonymy, with $m = 2$ or $m = 128$ expansions per symbol. The datasets contain $n =$ 65,536 and 68,267 sequences respectively, and each model is trained for $n_{\text{exp}} = 1000$ epochs. In both grammars, scaling separates into two phases: a generalization phase in which train and test losses decrease together, followed by a memorization phase in which a train–test gap opens and stored information grows linearly with parameter count until saturation. We define N_{mem} as the onset of a train–test gap, operationally identified as the first of two consecutive model sizes satisfying $(\mathcal{L}_{\text{test}} - \mathcal{L}_{\text{train}})/\mathcal{L}_{\text{train}} > 5\%$, and N_{sat} as the point at which stored information reaches 99% of the dataset ceiling $b^{\max}$. For $N_{\text{mem}} < N < N_{\text{sat}}$, we fit stored information to the linear form

$$b(\theta(D), K) = rN - n\Delta, \tag{14}$$

with two fitting parameters, the capacity ratio r in bits per parameter and the per-sequence offset Δ in bits. As shown in Figure 2, the two RHMs have similar capacity ratios, with $r = 2.99$ bits per added parameter for $m = 2$ and $r = 2.84$ for $m = 128$.

Capacity ratio grows with training budget. Figure 3(a,b) shows parameter scaling as the dataset size increases from $n =$ 4,267 to 68,267, with training budgets of 1000 and 100 epochs, respectively. At a fixed epoch budget, the capacity ratio r increases with dataset size. For each dataset size, extending training from 100 to 1000 epochs further increases r. To characterize the combined effect of dataset size and training duration, Figure 3(c) plots the capacity ratio r against the total number of sequence exposures,

$$C = n\, n_{\text{exp}}. \tag{15}$$

A power-law fit yields

$$r \approx 0.263\, C^{0.128}. \tag{16}$$

Over the observed range, a $10\times$ increase in C corresponds to about $1.3\times$ increase in r. The capacity ratio is therefore a property of the model together with its training budget.

The offset Δ reflects incomplete grammar learning. The offset Δ reflects imperfect learning of the surface sequence distribution $\mathcal{M}$. A model that has not learned the complete RHM grammar cannot narrow the sampling space of surface sequences from v^{s^L} to Ω_1, and generate instead from a sampling space $\Omega_1^{\text{learned}}$ with $|\Omega_1^{\text{learned}}| > |\Omega_1|$, shifting b down by $\Delta = \log_2(|\Omega_1^{\text{learned}}|/|\Omega_1|)$ per sequence. In Table 1 we compute the test loss and offset Δ assuming a model that knows the rules of the lowest k levels and treats the symbols at level $k{+}1$ as uniformly random over the vocabulary, and compare them with the test loss measured at N_{mem} and the fitted Δ. For $m = 2$, the observed values are closest to the full-grammar row. For $m = 128$, they are closest to the level-1 row. This follows from the data scaling of the RHM, where the data needed to learn each level grows with

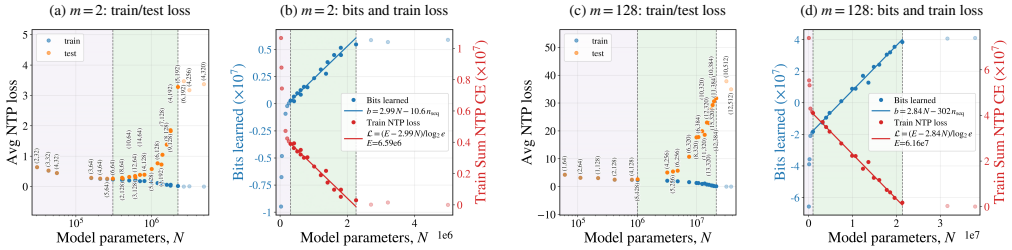


Figure 2: **Linear capacity emerges once the model is over-parameterized relative to the grammar.** Parameter scaling for two RHMs at v=128, c=32, s=4, L=4, differing only in synonymy m: *(a,b)* m=2, *(c,d)* m=128. Panels (a,c) show average train and test loss, with points labelled by depth and embedding width $(n_{\text{layer}}, d_{\text{embd}})$; the test-loss minimum marks the memorization onset N_{mem}. Panels (b,d) show stored bits (blue, left axis) and summed train CE loss (red, right axis); between N_{mem} and saturation, stored bits grow linearly at $r = 2.99$ and 2.84 bits per parameter.

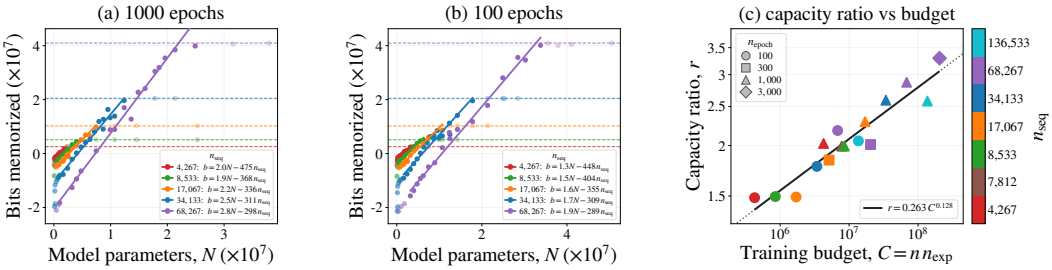


Figure 3: **Training exposure sets the ratio; dataset size sets the offset.** Parameter scaling for the RHM with $(v, c, m, s, L) = (128, 32, 128, 4, 4)$. Panels (a,b) show stored information versus parameter count after 1000 and 100 epochs, respectively, for different dataset sizes n. Solid lines fit $b = rN - n\Delta$ over the memorization window; faded points lie outside the fitting interval, and dashed horizontal lines mark dataset ceilings. Panel (c) shows the capacity ratio r versus total sequence exposures $C = n\,n_{\text{exp}}$, with the fit $r \approx 0.263\,C^{0.128}$. Colors indicate dataset size and marker shapes indicate epoch count.

m (Cagnetta et al., 2025a). Consistently, Δ decreases with the number of distinct sequences n but not with additional epochs (Figure 3(a,b)). Grammar learning is limited by data diversity, not repetition. We analyze this in Appendix E.

Connection to double descent. The generalization and memorization phases in Figure 2 match the model-wise double-descent picture (Belkin et al., 2019; Nakkiran et al., 2020; Hernandez et al., 2022). The memorization onset N_{mem} lies near the test-loss minimum, and the saturation point N_{sat} corresponds to the interpolation threshold. Appendix C shows the second descent of test loss at larger model sizes.

5 Linear capacity for latent knowledge

We now turn to latent knowledge at $\ell_{\text{mem}} > 1$. We fix the RHM grammar at $(v, c, m, s, L) = (16, 16, 4, 2, 6)$. The knowledge set $K = \{x^{(i)}\}_{i=1}^{n}$ contains $n = 2.5 \times 10^6$ pieces. The training set D consists of $n_{\text{exp}} = 1000$ fresh realizations of each knowledge piece.

Models memorize latent pieces, not training sequences. Figure 4(a) shows how the training loss and two test losses scale with model size for $\ell_{\text{mem}} = 2$ and $n_{\text{layer}} = 8$. The in-distribution (ID) test loss is evaluated on fresh realizations $t_{\text{ID}} \sim \mathcal{Q}(\cdot \mid x^{(i)})$ of the pieces $x^{(i)} \in K$. The out-of-distribution (OOD) test loss is evaluated on realizations $t_{\text{OOD}} \sim \mathcal{Q}(\cdot \mid x)$ of pieces $x \in \Omega_{\ell_{\text{mem}}} \setminus K$. For small models the three losses coincide and decrease together, as the model learns the grammar. Beyond N_{mem} (dashed line), the training and ID test losses continue to decrease together while the OOD test loss rises sharply. The model is therefore not memorizing the training sequences, which would open a gap between training and ID test loss, but the knowledge pieces themselves, which lets it predict unseen realizations of K and makes it assign less probability to pieces outside K.

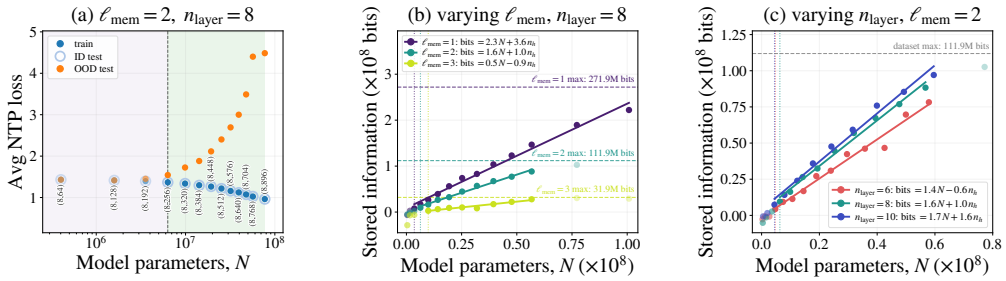


Figure 4: **Latent memorization in the RHM. (a)** Training, in-distribution (ID) test and out-of-distribution (OOD) test loss as a function of model parameters. Although no surface sequence is ever repeated, there is a memorization phase in which OOD test loss rises while training and ID test loss fall. **(b)** Capacity ratio at memorization levels $\ell_{\text{mem}} = 1, 2, 3$ for transformers of depth 8 and increasing number of heads. The capacity ratio drops rapidly as the knowledge becomes more abstract. **(c)** Capacity ratio at fixed $\ell_{\text{mem}} = 2$ as a function of transformer depth.

Capacity is linear at every abstraction level, with a ratio that decreases with abstraction. Raising ℓ_{mem} makes a knowledge piece more abstract and reduces how many bits it carries. From Equation (7), each piece at level ℓ_{mem} carries

$$\log_2 |\Omega_{\ell_{\text{mem}}}| \;=\; \log_2 c + \frac{s^{L-\ell_{\text{mem}}+1}-1}{s-1}\log_2 m \tag{17}$$

bits of knowledge, which decreases geometrically with ℓ_{mem}. The remaining $\log_2 |\Omega_1| - \log_2 |\Omega_{\ell_{\text{mem}}}|$ bits are details sampled freely in each realization, and a more abstract piece admits more realizations.

Figure 4(b) compares $\ell_{\text{mem}} = 1, 2, 3$ at fixed depth $n_{\text{layer}} = 8$, with the same n and n_{exp} at every level. A knowledge piece carries 130, 66 and 34 bits, and the ceiling

$$b^{\max}(K) \;=\; n\big(\log_2 |\Omega_{\ell_{\text{mem}}}| - \log_2 n\big) \tag{18}$$

drops from 271.9M to 111.9M to 31.9M bits. Stored information grows linearly with parameter count at every level, even though no training sequence is repeated. Linear scaling is therefore a property of memorization itself, not of repeated token patterns in the training data. The fitted offsets Δ are near zero, because the grammar is essentially learned from the $n\, n_{\text{exp}} = 2.5 \times 10^9$ distinct sequences in D. The capacity ratio falls sharply from 2.3 to 1.6 to 0.5 bits per parameter, so abstract knowledge is more expensive to store per bit. This drop is not explained by the smaller number of bits per piece, since in Section 4 the capacity ratio is $r = 2.99$ with 90 bits per piece at $m = 2$ and $r = 2.84$ with 600 bits per piece at $m = 128$.

The depth dependence is accounted for by identification. At fixed $\ell_{\text{mem}} = 2$, stored information grows linearly with parameter count for every depth (Figure 4(c)). Fitting Equation (14) gives $r = 1.4$, 1.6 and 1.7 bits per parameter for depths 6, 8 and 10. This ordering can be explained by where the knowledge piece is identified. Linear probes on the residual stream (Appendix G) show that for $\ell_{\text{mem}} = 2$ the piece can be read out after the third Transformer block, whether the model has 6, 8 or 10 blocks in total, whereas for $\ell_{\text{mem}} = 3$ it can be read out only after the fourth to sixth block. Counting only the parameters after this identification depth (Figure 14), the three depths store the same 2.2 bits per parameter, and the ℓ_{mem} dependence is reduced from $(2.3, 1.6, 0.5)$ to $(2.4, 2.2, 1.5)$. Deeper Transformers are more efficient at storing latent knowledge because they have more layers left after the fixed cost of identification. The probes show where a piece is identified but give no direct evidence that the later layers are what store it. Establishing this would require intervening on those layers, which we leave to future work.

6 Frequency-dependent allocation of finite capacity

Natural language exhibits highly non-uniform, often heavy-tailed frequency distributions, with a small number of patterns occurring frequently and many others appearing rarely. We therefore investigate how Transformers allocate memorization capacity across knowledge with different frequencies. We then compare this behavior with associative-memory models to characterize how

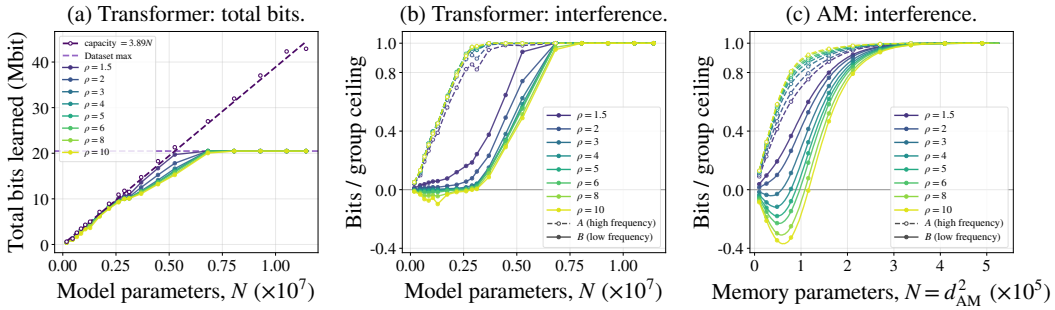


Figure 5: **Frequency redistributes model capacity in Transformers and associative memory (AM).** (a) Total knowledge scaling of Transformers; $3.89N$ is a reference slope measured on uniform single-group data. (b,c) Memorized knowledge in each group, normalized by that group's ceiling, showing that capacity goes to the frequent group and the rare group is suppressed.

strengthening one group of associations can interfere with the recall of another. To isolate the effect of frequency from that of dataset structure, we focus in this section on unstructured random key–value associations.

6.1 Frequency-dependent allocation in Transformers

We consider a two-group dataset consisting of a high-frequency group A and a low-frequency group B. Each group contains $n_A = n_B = 14{,}545$ sequences, with 4-token keys and 64-token values over a vocabulary of size $2{,}048$. During training, we show each group-B sequence $1{,}000$ times and each group-A sequence $1{,}000\rho$ times, where $\rho \in \{1.5, 2, 3, 4, 5, 6, 8, 10\}$. The group-wise capacity scaling in Figure 5(b) shows interference between the two groups. Information on B remains near zero until A is nearly exhausted, showing that capacity is allocated sequentially by frequency rather than shared between the groups. This interference intensifies with ρ: the information on B turns negative at small model sizes for $\rho \geq 3$. With sequential allocation, total stored information no longer follows a single linear law, as shown in Figure 5(a), where the reference capacity is measured on uniform single-group data at $1{,}000$ exposures per sequence. While the model is memorizing the frequent group A, total stored information tracks the linear capacity $3.89N$ closely; once the model begins spending parameters on the rare group B, the capacity ratio drops, and this decline steepens with ρ, falling from 3.7 bits per parameter at $\rho = 1.5$ to 2.8 bits per parameter at $\rho = 10$. Knowledge from the rare group is stored at a reduced rate even though its exposure count is unchanged.

6.2 Capacity allocation of associative memory with cross-entropy loss

Cabannes et al. (2024) show that embedding overlap induces cross-talk in associative memories, so that capacity depends on the relative weighting of stored associations; we ask which weighting a frequency-weighted cross-entropy objective selects, and whether it produces the suppression of Section 6.1.

Model. We store the $n = n_A + n_B$ associations $x_i \mapsto y_i$ of the two groups in an outer-product memory with one weight per group,

$$W(q_A, q_B) = q_A \sum_{j \in A} u_j e_j^\top + q_B \sum_{j \in B} u_j e_j^\top, \qquad q_A, q_B \geq 0, \tag{19}$$

with input and output embeddings e_j, u_j drawn independently and uniformly from the unit sphere in $\mathbb{R}^{d_{\mathrm{AM}}}$. On input x_i the memory scores every candidate output by $z_{ik} = u_k^\top W e_i$; the loss is the softmax cross-entropy over the n candidates, averaged over the embeddings and weighted by query frequency, with $p_A = \rho p_B$ (Appendix I.1). Averaging the weights within a group cannot increase the expected loss (Appendix I.3), so one weight per group loses nothing, and we minimize the loss numerically over (q_A, q_B) for each d_{AM} (Appendix I.2).

The optimal weighting reproduces the suppression. Evaluating the group-wise information at the optimum $(\widehat{q}_A, \widehat{q}_B)$ gives the same interference pattern as in the Transformer (Figure 5(c)): group A approaches its ceiling first, and the information about group B is delayed and turns negative at small d_{AM}. The ratio $\widehat{q}_A/\widehat{q}_B$ starts near ρ and decreases toward one as d_{AM} grows (Figure 15(a)), and along this path $\widehat{q}_B$ increases while the information about B initially decreases (Figure 15(b)). The optimizer does not ignore B; the weight it gives to A suppresses B.

The frequent group suppresses the rare one through cross-talk. For a query $i \in B$ and a competing label $k \in A$, the logit margin $z_{ik} - z_{ii}$ is a correct-label advantage, $-q_B(1 - u_k^\top u_i)$, plus cross-talk from every other stored association, each term proportional to an embedding overlap $e_j^\top e_i$ (Equation (33)). For independent unit vectors $\mathbb{E}[(e_j^\top e_i)^2] = 1/d_{\mathrm{AM}}$, so the ratio of the RMS cross-talk to the correct-label advantage is

$$\frac{\mathrm{RMS(cross\text{-}talk)}}{\mathrm{RMS(correct\text{-}label\ advantage)}} \approx \frac{\sqrt{n}}{d_{\mathrm{AM}}} \sqrt{1 + \left(\frac{q_A}{q_B}\right)^2} \tag{20}$$

(Appendix I.4). When this ratio is of order one, some of the n_A competing labels score above the correct one, and each contributes $e^{z_{ik} - z_{ii}} > 1$ to the cross-entropy of the rare query. Small d_{AM} and a large q_A/q_B therefore raise the loss of group B: the weight that the objective gives to the frequent group is the noise that suppresses the rare one.

7 Related work

Linear knowledge capacity. Large models memorize training data verbatim (Carlini et al., 2021; 2023; Tirumala et al., 2022) and store relational facts recoverable by probing or question answering (Petroni et al., 2019; Roberts et al., 2020; Geva et al., 2021; Meng et al., 2022). On synthetic data, the information a network stores is *linear* in its parameter count: 3–6 bits per parameter for RNNs on random labels (Collins et al., 2017); 2 bits per parameter for Transformers on synthetic biographies, across architectures and down to `int8` quantization (Allen-Zhu & Li, 2024); ≈ 3.6 bits per parameter on uniform random strings, with double descent beginning where the data's information content exceeds capacity (Morris et al., 2025); and linear in non-embedding parameters for atomic (key, attribute, value) facts in pretrained LLMs (Lu et al., 2024). Allen-Zhu & Li (2023a;b) find that stored facts become extractable only when pretraining exposes each fact in varied phrasings.

Associative memories. Associative memories provide a classical model of neural storage and recall (Hopfield, 1982; Amit et al., 1985; Personnaz et al., 1986; Gardner, 1988; Krotov & Hopfield, 2016; Demircigil et al., 2017). Bietti et al. (2023) study the emergence of associative memories, and of an induction head, during Transformer training. Cabannes et al. (2024) derive scaling laws for outer-product memories with random embeddings, relating prediction error to cross-talk noise and the relative weights of stored associations; they further analyze cross-entropy training of an unrestricted memory matrix by approximating the resulting dynamics through association weights. Nichani et al. (2025) establish near-linear storage capacity in parameter count for linear and MLP associative memories, and show how shallow Transformers can compose these modules for structured factual recall. Relative to Cabannes et al. (2024), who analyze cross-talk at given association weights, we optimize the group weights under frequency-weighted cross-entropy and show that the optimum itself suppresses rare associations, reproducing the pattern in our Transformer experiments.

Random hierarchy model. The RHM (Cagnetta et al., 2024) provides exact control over compositional structure: data are generated by a tree of random composition rules. Transformers trained on such hierarchical grammars exhibit predictable sample-complexity and scaling behaviour (Cagnetta & Wyart, 2024; Cagnetta et al., 2025a;b), and recent work extends the analysis to learning from latent rather than token targets (Korchinski et al., 2026).

8 Limitations

Our measure of stored knowledge requires the marginal $\mathcal{M}$ and therefore applies only to data with a known surface distribution, not to natural language. The latent-knowledge results are for a single grammar, and although we measure 3.9 bits per parameter on unstructured key–value data but 2–3 on the ten RHM configurations in Section 4 and Appendix D, we do not determine which properties of the grammar account for this gap. The frequency results use two groups of equal size, and heavy-tailed frequency distributions are not studied. The probes of Appendix G locate where a knowledge piece is identified but give no direct evidence of where it is stored, which would require intervening on the later layers. We leave a measurement on natural language, a systematic sweep over (v, c, m, s, L), heavy-tailed frequencies and a causal test of the storage hypothesis to future work.

Reproducibility statement

All datasets are synthetic and can be regenerated from the grammar parameters, dataset sizes and exposure settings given in the text and appendix. Training and evaluation settings are described in Section 3 and Appendix A.

AI use statement

We used generative AI tools to write and debug experiment code and to assist in writing the derivations. We also used them to draft and edit text, identify related work, format references and check consistency. We did not use them for any other task with required disclosure. The authors tested all code, checked all derivations and take full responsibility for the content of this work.

References

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. *arXiv preprint arXiv:2309.14316*, 2023a.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.2, knowledge manipulation. *arXiv preprint arXiv:2309.14402*, 2023b.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws. *arXiv preprint arXiv:2404.05405*, 2024.

Daniel J. Amit, Hanoch Gutfreund, and H. Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55(14):1530–1533, 1985. doi: 10.1103/PhysRevLett.55.1530.

Maissam Barkeshli, Alberto Alfarano, and Andrey Gromov. On the origin of neural scaling laws: From random graphs to natural language, 2026. URL `https://arxiv.org/abs/2601.10684`.

Peter L. Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. In *Advances in Neural Information Processing Systems*, 2023.

Vivien Cabannes, Elvis Dohmatob, and Alberto Bietti. Scaling laws for associative memories. In *International Conference on Learning Representations*, 2024.

Francesco Cagnetta and Matthieu Wyart. Towards a theory of how the structure of language is acquired by deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2406.00048.

Francesco Cagnetta, Leonardo Petrini, Umberto M. Tomasini, Alessandro Favero, and Matthieu Wyart. How deep neural networks learn compositional data: The random hierarchy model. *Physical Review X*, 14(3):031001, 2024.

Francesco Cagnetta, Alessandro Favero, Antonio Sclocchi, and Matthieu Wyart. Scaling laws and representation learning in simple hierarchical languages: Transformers vs. convolutional architectures. *arXiv preprint arXiv:2505.07070*, 2025a.

Francesco Cagnetta, Hyunmo Kang, and Matthieu Wyart. Learning curves theory for hierarchically compositional data with power-law distributed features. In *International Conference on Machine Learning (ICML)*, 2025b. arXiv:2505.07067.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium*, 2021.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. Quantifying memorization across neural language models. In *International Conference on Learning Representations*, 2023.

Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. Capacity and trainability in recurrent neural networks. In *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1611.09913.

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

Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, 2017.

Elizabeth Gardner. The space of interactions in neural network models. *Journal of Physics A: Mathematical and General*, 21(1):257–270, 1988.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Paul W. Goldberg and Mark R. Jerrum. Bounding the Vapnik-Chervonenkis dimension of concept classes parameterized by real numbers. *Machine Learning*, 18(2–3):131–148, 1995.

Danny Hernandez, Tom Brown, Tom Conerly, Nova DasSarma, Dawn Drain, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Tom Henighan, Tristan Hume, Scott Johnston, Ben Mann, Chris Olah, Catherine Olsson, Dario Amodei, Nicholas Joseph, Jared Kaplan, and Sam McCandlish. Scaling laws and interpretability of learning from repeated data, 2022. URL `https://arxiv.org/abs/2205.10487`.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

John J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

Marcus Hutter. Learning curve theory. *arXiv preprint arXiv:2102.04074*, 2021.

Tokio Kajitsuka and Issei Sato. On the optimal memorization capacity of transformers. *arXiv preprint arXiv:2409.17677*, 2024.

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

Junghwan Kim, Michelle Kim, and Barzan Mozafari. Provable memorization capacity of transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Daniel J. Korchinski, Alessandro Favero, and Matthieu Wyart. Learn from your own latents and not from tokens: A sample-complexity theory. *arXiv preprint arXiv:2605.27734*, 2026.

Dmitry Krotov and John J. Hopfield. Dense associative memory for pattern recognition. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Xingyu Lu, Xiaonan Li, Qinyuan Cheng, Kai Ding, Xuanjing Huang, and Xipeng Qiu. Scaling laws for fact memorization of large language models. *arXiv preprint arXiv:2406.15720*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, 2022.

Eric J. Michaud, Ziming Liu, Uzay Girit, and Max Tegmark. The quantization model of neural scaling. In *Advances in Neural Information Processing Systems*, 2023.

John X. Morris, Chawin Sitawarin, Chuan Guo, Narine Kokhlikyan, G. Edward Suh, Alexander M. Rush, Kamalika Chaudhuri, and Saeed Mahloujifar. How much do language models memorize? *arXiv preprint arXiv:2505.24832*, 2025.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *International Conference on Learning Representations*, 2020.

Eshaan Nichani, Jason D. Lee, and Alberto Bietti. Understanding factual recall in transformers via associative memories. *arXiv preprint arXiv:2412.06538*, 2025.

L. Personnaz, I. Guyon, and G. Dreyfus. Collective computational properties of neural networks: New learning mechanisms. *Physical Review A*, 34(5):4217–4228, 1986. doi: 10.1103/PhysRevA.34.4217.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

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

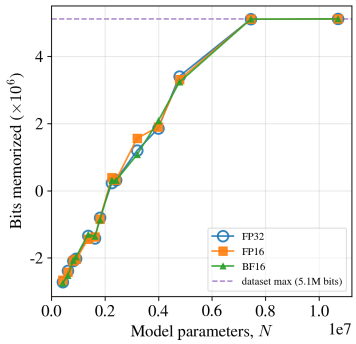
Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

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

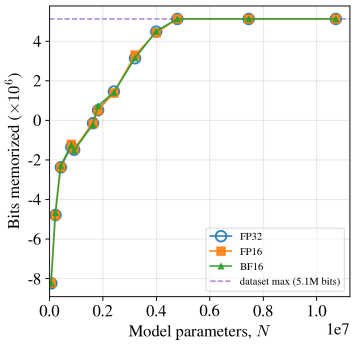
Kushal Tirumala, Aram H. Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. In *Advances in Neural Information Processing Systems*, 2022.

Gal Vardi, Gilad Yehudai, and Ohad Shamir. On the optimal memorization power of ReLU neural networks. *arXiv preprint arXiv:2110.03187*, 2021.

Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466*, 2022.



(a) 100 epochs ($B = 64$, $T = 13{,}300$).



(b) 1000 epochs ($B = 256$, $T = 33{,}000$).

Figure 6: **Measured capacity is invariant to training precision.** Bits memorized versus model parameters N on the RHM classification task (v=128, c=32, m=128, s=4, L=4, n=8,533), for FP32, FP16 and BF16. The dashed line is the dataset ceiling $b^{\max} = 5{,}119{,}800$ bits.

A Experiment details

Models. All Transformers are decoder-only GPT-2 models with rotary position embeddings, pre-LayerNorm, GELU MLPs of width $4d_{\text{embd}}$ and head dimension 64. A model size is specified by $(n_{\text{layer}}, d_{\text{embd}})$, and N counts all trainable parameters including the embeddings.

Optimization. All Transformer runs use AdamW with $(\beta_1, \beta_2) = (0.9, 0.999)$, gradient clipping at norm 1.0, batch size 256, and linear warmup followed by cosine decay. For every model size we scan the peak learning rate over a small grid and keep the run with the most stored bits (verbatim and key–value experiments) or the lowest final training loss (latent experiments). The associative-memory coefficients (q_A, q_B) are fitted as described in Appendices I.2 and I.5.

Evaluation. Verbatim runs are evaluated on the training set and on 4,000 fresh sequences, and the reported point is the epoch with the most stored bits. Latent runs are evaluated at the final epoch on fresh realizations of 2,000 pieces in the knowledge set (ID) and of 2,000 pieces outside it (OOD). Key tokens are excluded from every loss and bit count.

B Training precision

Figure 6 checks that training precision does not affect the measured capacity. We repeat one sweep ($n = 8{,}533$, 14 sizes from 6×10^4 to 10^7 parameters) in FP32, FP16 and BF16 with otherwise identical hyperparameters, at 100 and at 1000 epochs. The three precisions give the same stored bits at every size, in both the grammar-learning regime (negative bits) and the memorization regime, and saturate at the same ceiling. The largest deviation is a single FP16 point at 100 epochs ($N = 3 \times 10^6$, $+0.4$M bits).

C Model-wise double descent

Figure 2 covers model sizes up to saturation. Here we extend the sweep to thirty times the saturation size to show that the test loss descends again once the model is far into the over-parameterized regime. We use a shorter RHM, $(v, c, m, s, L) = (128, 32, 128, 6, 2)$, so that saturation is reached at a smaller model size. The setup is otherwise as in Section 4.

The test loss rises from 2.3 nats per token at N_{mem} to a peak of 17.6 at N_{sat}, where the training loss reaches zero, and then decreases monotonically to 3.4 at $N = 8.5 \times 10^7$. The memorization onset N_{mem} therefore sits near the first test-loss minimum and N_{sat} at the interpolation threshold, as in model-wise double descent (Belkin et al., 2019; Nakkiran et al., 2020). The linear capacity regime of Section 4 is the ascending part of the curve, between the first minimum and the peak.

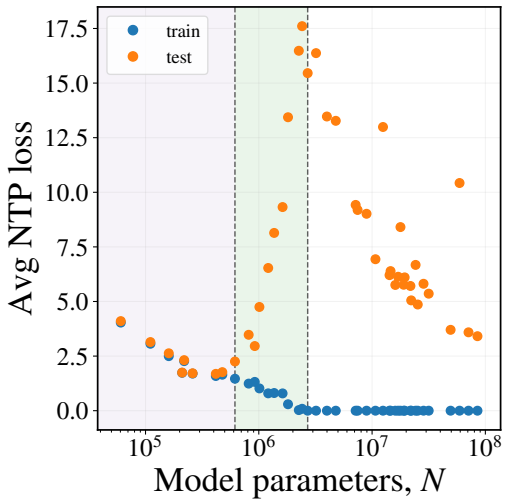


Figure 7: **Model-wise double descent: test loss peaks at saturation and descends again in larger models.** Verbatim memorization on an RHM with $(v, c, m, s, L) = (128, 32, 128, 6, 2)$, so each sequence has $T = 36$ tokens and carries 54 bits. The training set holds $n = 47{,}407$ sequences, each prefixed with a 4-token key, giving $b^{\max} = 2.56$M bits. We train 44 Transformers with $n_{\text{layer}} \in \{1, \dots, 12\}$ and $d_{\text{embd}} \in \{64, \dots, 768\}$, spanning $N = 6 \times 10^4$ to 8.5×10^7, for 1000 epochs. Blue: training loss. Orange: test loss on 4,000 held-out sequences. Dashed lines mark $N_{\text{mem}} = 6.2 \times 10^5$ and $N_{\text{sat}} = 2.7 \times 10^6$, and shading marks the grammar-learning (purple) and memorization (green) regimes. The test loss rises from 2.3 nats per token at N_{mem} to 17.6 at N_{sat}, where the training loss reaches zero, and then decreases to 3.4 at $N = 8.5 \times 10^7$, thirty times the saturation size.

D Results on additional RHM configurations

Figures 8 and 9 repeat the verbatim-memorization sweep of Section 4 on eight further RHM grammars with different vocabulary size v, number of classes c, synonymy m, branching factor s and depth L, given in each panel title, with dataset sizes between $n = 5{,}633$ and 77,576 sequences and 1000 training epochs. The onset N_{mem}, saturation N_{sat} and fitting window follow the same rules as in the main text.

Every grammar shows the same two-phase behavior. Train and test loss decrease together until N_{mem}, after which the test loss rises while the training loss continues to fall and stored information grows linearly with parameter count until saturation (Figure 9). The fitted capacity ratios lie between 2.05 and 2.65 bits per parameter across all eight grammars (Figure 8), somewhat below the 2.8–3.0 measured in Section 4 and well below the 3.9 measured on unstructured key–value data. The fitted offset Δ varies more widely, from 13 to 1,560 bits per sequence, and is largest for the grammars with the largest branching factor s and synonymy m, consistent with the interpretation in Section 4 that Δ reflects the part of the grammar not learned before memorization begins.

E The floor on Δ set by n

The linear fits in Figure 3 return per-sequence offsets Δ that decrease with n, from 475 bits at $n = 4267$ toward roughly 300 bits at the largest n. Here we show that this behaviour is set by what is statistically learnable from n sequences: Δ relaxes toward a parameter-free floor determined by the leaf-tuple marginals, and cannot descend below it in our experimental range.

Setup. We consider the RHM instance with $s = 4$, $L = 4$, $m = 128$, $v = 128$, $c = 32$, so each sequence consists of $T = s^L = 256$ tokens organised into $s^{L-1} = 64$ leaf patches of s tokens each. The true sequence entropy is $\log_2 |\Omega_1| = \log_2 c + \frac{s^L - 1}{s-1} \log_2 m = 600$ bits.

What is learnable below the first resolution threshold. Identifying the hierarchical structure proper — grouping the mv valid s-tuples into their v parent symbols — requires detecting

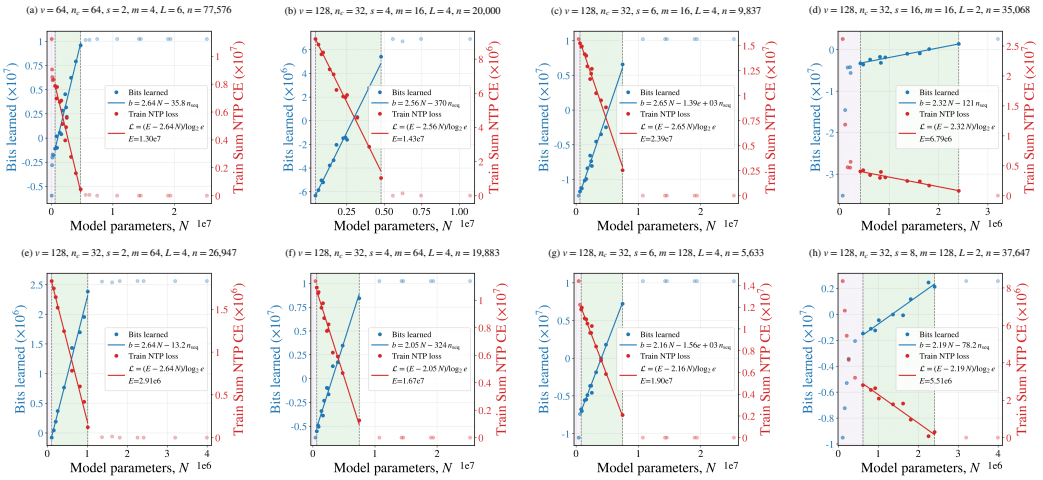


Figure 8: **Stored information versus parameter count on eight additional RHM grammars.** Blue: stored bits (left axis), with the linear fit $b = rN - n\Delta$ over the memorization window (shaded green). Red: summed training cross-entropy (right axis). Grammar parameters and dataset size n are given in each panel title.

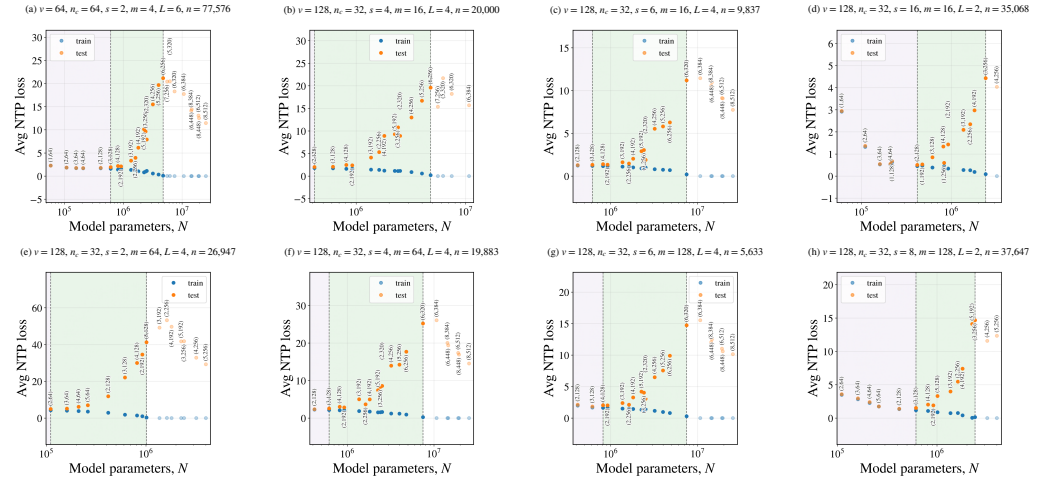


Figure 9: **Train and test loss for the grammars in Figure 8.** Average next-token loss on training (blue) and held-out (orange) sequences versus parameter count, with points labeled by $(n_{\text{layer}}, d_{\text{embd}})$. Dashed lines mark N_{mem} and N_{sat}; shading marks the grammar-learning (purple) and memorization (green) regimes.

correlations between tuples and the remainder of the sequence, with sample complexity $P_1 = (1-f)^{-1} vm^3 \approx 2.7 \times 10^8$, where $f = m/v^{s-1}$ (Cagnetta et al., 2025a, Eq. 15). All our training-set sizes satisfy $n \ll P_1$, so no synonym structure is statistically resolvable. What remains learnable is the *marginal distribution of s-tuples* at each leaf patch: which of the $v^s \approx 2.7 \times 10^8$ conceivable tuples occur at all (only $mv = 16{,}384$ do, each with frequency $\approx 1/(mv)$), and their frequencies. Acquiring this requires only that each valid tuple be observed: a coupon-collector condition, $n \gtrsim \frac{mv}{s^{L-1}} \ln(mv) \approx 2.5 \times 10^3$, satisfied by even our smallest training set. We verified directly that at $n = 4267$ every valid tuple appears in the training data (mean count 16.7, minimum 3, count variance consistent with Poisson sampling).

The tuple-marginal floor. A model that knows the tuple marginals but not the synonym grouping samples each of the s^{L-1} leaf patches independently and uniformly over the mv valid tuples, so its effective sampling space has $\log_2 |\Omega_1^{\text{learned}}| = s^{L-1} \log_2(mv)$. The corresponding excess entropy is

$$\Delta_{\min} \;=\; s^{L-1} \log_2(mv) \;-\; \frac{s^L - 1}{s - 1} \log_2 m \;-\; \log_2 c \;=\; 296 \text{ bits/seq}, \tag{21}$$

Table 1: Predicted test loss (nats per token) and offset Δ for a model that knows the rules of the lowest k levels and treats level-$(k{+}1)$ symbols as uniformly random, for the two RHMs in Figure 2. The last row gives the observed test loss at N_{mem} and the fitted Δ. Matching predictions are in bold.

		$m = 2$		$m = 128$	
Rules learned	$\lvert\Omega_1^{\mathrm{learned}}\rvert$	test loss	Δ	test loss	Δ
none	v^{s^L}	4.85	1702	4.85	1192
level 1	$(vm)^{s^{L-1}}$	1.39	422	**2.43**	**296**
full grammar	$\lvert\Omega_1\rvert = c\,m^{(s^L-1)/(s-1)}$	**0.24**	**0**	1.63	0
Observed		0.26	10.6	2.61	302.4

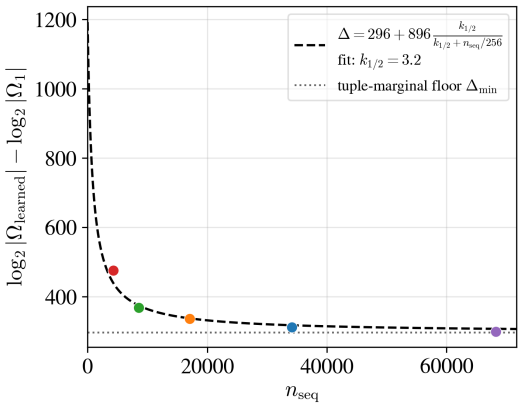


Figure 10: Excess entropy Δ versus training-set size, with the theory-fixed floor and amplitude. Points are the per-sequence offsets obtained from the linear fits of Figure 3(a). The dashed curve is Equation (24) with the floor $\Delta_{\min} = 296$ bits and amplitude 896 bits fixed by Equations (21) and (22), and the single kinetic parameter fitted to $k_{1/2} = 3.2$ tuple exposures. The dotted line marks the tuple-marginal floor, below which no model can fall for $n \ll P_1$.

with no adjustable parameters. Because no further structure is statistically accessible for $n < P_1$, Equation (21) is a floor: Δ cannot fall below 296 bits anywhere in our sweep. The fitted offsets at the two largest training sets (310.8 and 298.1 bits) sit at this floor within fit uncertainty. The total height of the relaxation onto the floor is likewise fixed: a fully ignorant model predicts all leaves uniformly, giving $\Delta(0) = s^L \log_2 v - \log_2 |\Omega_1| = 1192$ bits, so the amplitude is

$$\Delta(0) - \Delta_{\min} \;=\; s^{L-1}\big(s \log_2 v - \log_2 mv\big) \;=\; -\,s^{L-1} \log_2 f \;=\; 896 \text{ bits}. \tag{22}$$

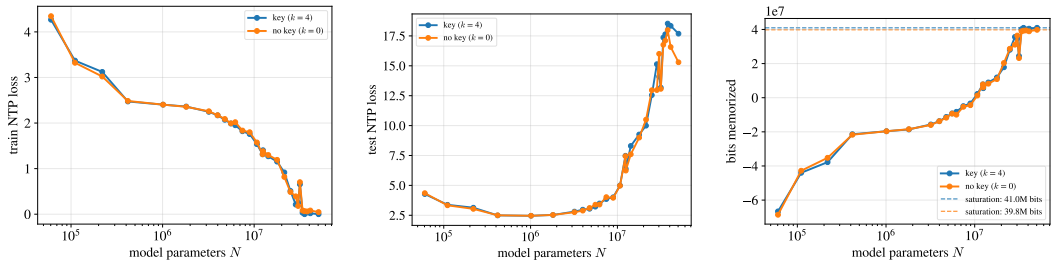
Exposure-limited approach to the floor. Although all tuples are present in the data at every n we use, the trained models reach the floor only gradually. The natural exposure variable is the mean number of occurrences of each tuple type in the training set,

$$\bar{k} \;=\; \frac{n\, s^{L-1}}{mv} \;=\; \frac{n}{256}, \tag{23}$$

which we verified against the empirical counts (e.g. $\bar{k} = 16.7$ at $n = 4267$). Modelling the consolidation of each tuple's probability as a saturating function of its exposure count, $\phi(\bar{k}) = \bar{k}/(\bar{k}{+}k_{1/2})$, the excess entropy interpolates hyperbolically between the two theory-fixed anchors:

$$\Delta(n) \;=\; \Delta_{\min} \;+\; \big(\Delta(0) - \Delta_{\min}\big)\, \frac{k_{1/2}}{k_{1/2} + n/256}. \tag{24}$$

Figure 10 shows the fitted offsets from Figure 3(a) against Equation (24) with the single kinetic parameter $k_{1/2} = 3.2$: a tuple is half-consolidated after roughly three occurrences, and the floor is approached once $\bar{k} \gg k_{1/2}$. We emphasise the asymmetry of these claims: the floor (Equation (21)) and amplitude (Equation (22)) are equilibrium properties of the data distribution and are predicted exactly, whereas $k_{1/2}$ characterises the training dynamics and is fitted. A statistically ideal learner would sit essentially on the floor at all our training-set sizes (its only excess being the plug-in estimation error $\frac{mv-1}{2\ln 2}\, n^{-1} \le 3$ bits); the measured relaxation therefore reflects optimisation kinetics



(a) train NTP loss (b) test NTP loss (c) bits memorized

Figure 11: **Keying the sequences leaves the capacity curve invariant and moves only the ceiling.** Same RHM (v=128, c=32, s=4, L=4, m=128) at ℓ_{mem}=1, n=68,267, 100 epochs, run with a k=4 token key prefixed to each stored sequence (blue) and without any key (orange). *(a,b)*. The two conditions coincide at every model size; only the saturation level differs, by the identification term $n \log_2 n$ (dashed lines: 41.0M bits keyed, 39.8M keyless).

— the number of gradient exposures needed to consolidate each tuple — rather than information availability.

Implication for the main-text fits. For $n \gtrsim 3{\times}10^4$ (i.e. $\bar{k} \gtrsim 130 \gg k_{1/2}$), Δ is pinned at the floor and can be treated as the known constant $\Delta_{\min} = 296$ bits in the capacity ledger $\text{bits} = rN - n\Delta$. At smaller n the offset exceeds the floor by the exposure-limited term of Equation (24), which should be borne in mind when interpreting the corresponding fitted capacities.

F KEYS DO NOT CHANGE CAPACITY, ONLY SATURATION

In the verbatim setting ℓ_{mem}=1 the knowledge piece is the leaf sequence itself, so it can be addressed by a key, as in capacity studies on unstructured data (Allen-Zhu & Li, 2024; Lu et al., 2024; Morris et al., 2025); this is how the sequences of Section 4 are stored. Keys are defined only in this setting—the latent-memorization experiments are unkeyed throughout. This appendix shows that the key changes neither the capacity curve nor the capacity ratio, only the saturation ceiling—even though the model does use it as a retrieval address.

We repeat the sweep at ℓ_{mem}=1, n = 68,267, 100 epochs, in two conditions. *Keyed*: each stored sequence is prefixed with $k = 4$ tokens drawn uniformly at random from the vocabulary, fixed across its exposures; held-out sequences are given freshly sampled keys, so a test key identifies nothing. *Keyless*: $k = 0$. A loss mask excludes the key positions, so both conditions are scored over the same $s^L = 256$ content tokens. Everything else—grammar instance, sequence set, model family, optimizer, hyperparameter tuning—is shared.

Figure 11 shows the two conditions are indistinguishable across the sweep: same test-loss minimum, same onset N_{mem}, same capacity ratio r, same offset Δ, and the same non-monotonic excursion near $N \approx 3 \times 10^7$, which is therefore an optimization artefact rather than a key effect. Only saturation differs, and by a fixed amount. Without a key the model cannot know *which* of the n stored sequences to emit; its best predictor is the uniform mixture over them, which costs $\log_2 n$ bits per sequence and floors the training loss at $\ln n/s^L = 0.043$ nats/token. A key removes this ambiguity, so the training loss reaches zero and the ceiling rises by exactly $n \log_2 n$:

$$b^{\max}_{k>0} \;=\; n \log_2 |\Omega_1|, \qquad b^{\max}_{k=0} \;=\; n\big(\log_2 |\Omega_1| - \log_2 n\big). \tag{25}$$

At n = 68,267 and $\log_2 |\Omega_1| = 600$ bits this gives 40.96M bits keyed against 39.86M keyless, measured as 41.0M and 39.8M in Figure 11c—a 2.7% gap. Capacity is thus set by how much the parameters store, not by how the stored items are addressed; the two ledgers differ by a known constant.

Keys act as addresses. That the two ledgers coincide does not mean the model ignores the key. Prompted with the key alone and left to decode all 256 tokens with no leaf context, a saturated model reproduces its training sequences exactly. 100% token accuracy, first divergence at position 256, indistinguishable from teacher forcing (Table 2). Memorization is therefore retrieved by the key.

Table 2: **The key is a genuine address.** Greedy generation from a saturated keyed model (n_{layer}=12, d_{embd}=512, N=37.9M, 1000 epochs; same grammar and keys as training), scored against the stored sequence on 512 sampled sequences per split. *Teacher forcing* supplies the true leaf prefix at every step; *key only* prompts with the key and then self-feeds for all 256 tokens. "First div." is the mean position of the first incorrect token. Slots 0–3 are leaf position $\bmod s$, i.e. accuracy on the 1st–4th child within each production.

split	prompt	tok. acc.	exact	first div.	slots 0/1/2/3
train	teacher forcing	1.000	1.000	—	1.00 1.00 1.00 1.00
train	key only	1.000	1.000	256.0	1.00 1.00 1.00 1.00
test	teacher forcing	0.406	0.000	—	0.01 0.02 0.62 0.98
test	key only	0.008	0.000	0.0	0.01 0.01 0.01 0.01

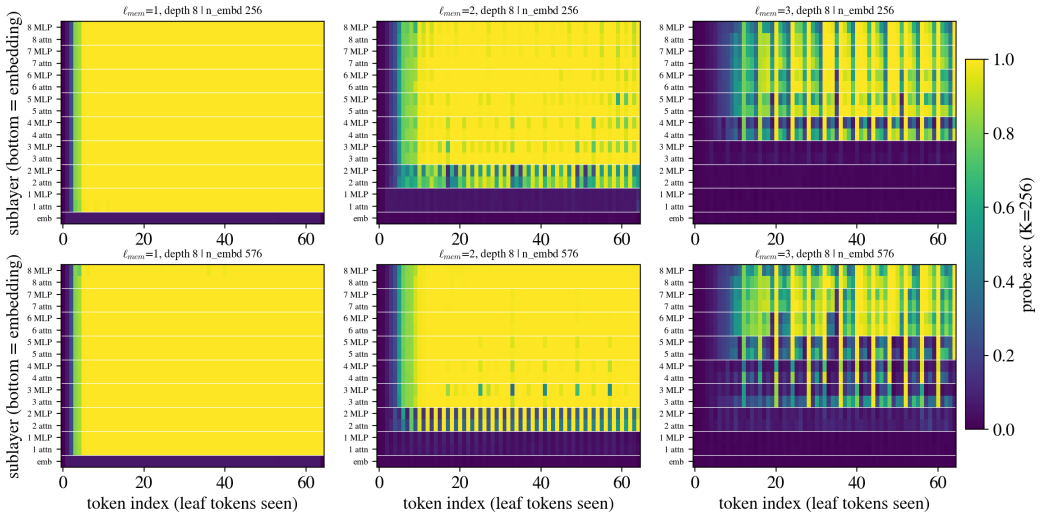


Figure 12: **Piece identity becomes decodable deeper in the network as ℓ_{mem} grows.** Probe accuracy for piece identity ($K = 256$) at every sublayer (rows, bottom = embedding) and token position (columns) for depth-8 models at $\ell_{\text{mem}} = 1, 2, 3$ (left to right) and widths $d_{\text{embd}} = 256$ (top) and 576 (bottom).

The teacher-forced accuracy on held-out sequences (0.406) is grammar generalization, concentrated in the later children of each production (0.62 and 0.98 at slots 2 and 3 against 0.01 at slots 0 and 1), where the earlier siblings already pin down the rule. Addressing consumes no measurable capacity, and its only remove the $n \log_2 n$ term of Equation (25).

G Where the knowledge piece is identified

We hypothesize that the dependence of the capacity ratio on depth and on ℓ_{mem} in Section 5 arises because the early layers of the Transformer are used to identify the knowledge piece generating the surface tokens and only the remaining layers store it. A higher ℓ_{mem} requires more layers for identification, and a shallower network has fewer layers in total, so both reduce the layers available for storage. To test this, we probe the residual stream of trained models for the identity of the knowledge piece.

Probe. For every trained model of Figure 4 we sample $K = 256$ stored pieces and 40 fresh realizations of each, 32 for fitting and 8 held out. We record the residual stream after the embedding and after every attention and MLP sublayer, at every token position, and fit a ridge-regression probe to the one-hot piece identity. Residuals of width $d_{\text{embd}} > 256$ are projected to 256 dimensions with a fixed random matrix so that the probe has the same capacity at every width. Accuracy is the top-1 rate on held-out realizations; chance is $1/K = 0.4\%$.

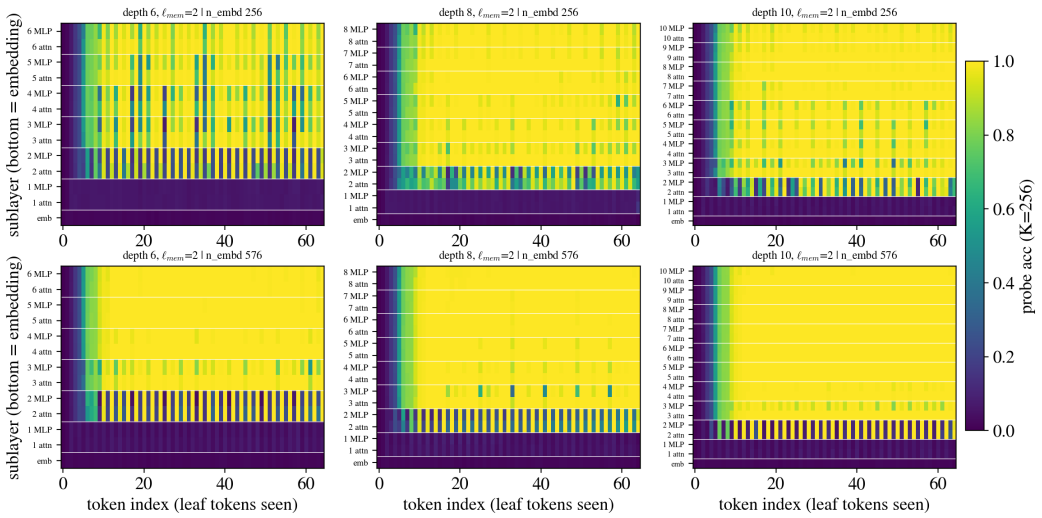


Figure 13: **The identification depth does not change with model depth.** As Figure 12, for ℓ_{mem}=2 at depths 6, 8 and 10 (left to right).

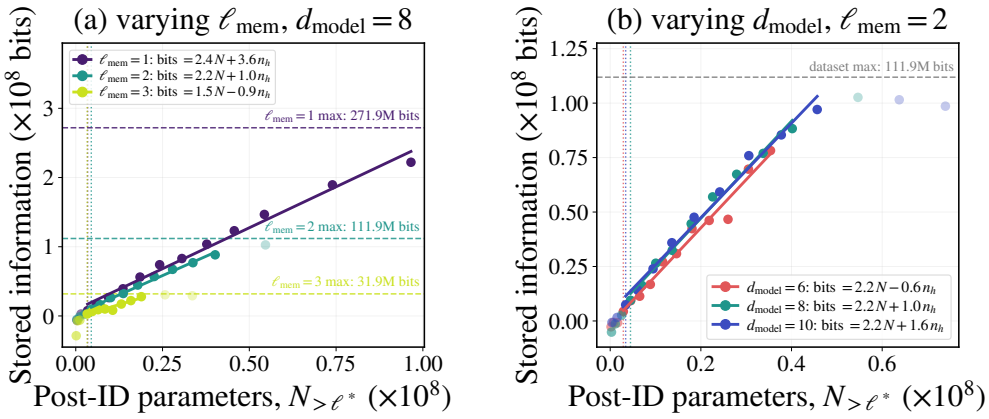


Figure 14: **Counting only the parameters after identification collapses the depth dependence of the capacity ratio.** Figure 4(b,c) redrawn against the post-identification parameter count $N_{>\ell^*}$, with the same runs and fits as the main-text figure. *(a)* Depth 8, $\ell_{\text{mem}} = 1, 2, 3$. *(b)* ℓ_{mem}=2, depths 6, 8, 10.

Identification depth. Figure 12 shows that verbatim pieces (ℓ_{mem}=1) are identified by the first attention sublayer from the first few tokens. Level-2 pieces are identified in two steps, partially after the attention of block 2 and at every position after the attention of block 3. Level-3 pieces are identified only in blocks 4–6, and the average accuracy stays below 0.9. At fixed ℓ_{mem}=2 (Figure 13), the identity is complete after the attention of block 3 at depths 6, 8 and 10 alike. These observations hold at every width $d_{\text{embd}} \geq 256$. We define the identification depth ℓ^* as the first sublayer whose accuracy, averaged over positions ≥ 16, reaches 90% of the model's peak, counting the attention of block l as $l - \frac{1}{2}$ and its output as l. Taking the most common value over widths gives $\ell^* = \frac{1}{2}$ for ℓ_{mem}=1, $\ell^* = 2\frac{1}{2}$ for ℓ_{mem}=2 at all three depths, and $\ell^* = 5\frac{1}{2}$ for ℓ_{mem}=3.

Capacity per post-identification parameter. If the blocks up to ℓ^* are spent on identification, stored bits should scale with the parameters after ℓ^*. We define $N_{>\ell^*}$ as the parameters in the MLP of the block whose attention is ℓ^* plus all later blocks, and redraw Figure 4(b,c) against it (Figure 14). The depth dependence disappears: depths 6, 8 and 10 each store 2.2 bits per post-identification parameter, where they stored 1.4, 1.6 and 1.7 bits per total parameter. The ℓ_{mem} dependence is reduced but does not disappear, with ratios of 2.4, 2.2 and 1.5 for $\ell_{\text{mem}} = 1, 2, 3$ against 2.3, 1.6 and 0.5 per total parameter. The remaining drop at ℓ_{mem}=3 coincides with identification being

incomplete, so that the storage blocks receive a noisier key. A direct test of the hypothesis, for example freezing or ablating the blocks before ℓ^*, is left to future work.

H EXPOSURE SETTINGS FOR THE TRANSFORMER FREQUENCY SWEEP

The random-map experiment uses $n_A = n_B = 14{,}545$, a vocabulary of $2{,}048$ tokens, four key tokens, and 64 independently sampled value tokens per sequence. Information is measured on value tokens. If ℓ_g^{tok} denotes the group's mean value-token CE in nats, then

$$b_g = 14{,}545 \times 64 \left(\log_2 2{,}048 - \frac{\ell_g^{\mathrm{tok}}}{\ln 2} \right). \tag{26}$$

The key identifies the association; its four tokens are not included in the value-information ceiling. Each group has $10{,}239{,}680$ bits at zero value loss, for a total ceiling of $20{,}479{,}360$ bits.

Let E_A, E_B be exposures *per sequence*. The design fixes

$$E_B = 1{,}000, \qquad E_A = 1{,}000\rho, \qquad \text{total sequence presentations} = 14{,}545 \times 1{,}000(1+\rho). \tag{27}$$

We vary $\rho \in \{1.5, 2, 3, 4, 5, 6, 8, 10\}$ while keeping the number of presentations of each B sequence fixed. Models are trained in BF16 with μP parameterization, batch size 256, and zero weight decay. We evaluate 16 model sizes and sweep three learning rates.

This protocol controls exposure to B, while total training duration and the spacing of its presentations can vary with ρ. Consequently, the comparison measures changes under increased exposure to A at fixed B exposure, while training duration and presentation schedule remain possible contributors. The reported curves characterize variation with model size and frequency; they do not estimate variability across independent training seeds.

I ASSOCIATIVE MEMORY

I.1 MEMORY, SOFTMAX SCORES AND CROSS-ENTROPY LOSS

Consider the outer-product memory with one weight per association,

$$W(\mathbf{q}; E, U) = \sum_{i=1}^{n} q_i u_i e_i^\top, \qquad q_i \geq 0, \tag{28}$$

where q_i is the memory weight of association i. The input embeddings e_i and output embeddings u_i are drawn independently and uniformly from the unit sphere in $\mathbb{R}^{d_{\mathrm{AM}}}$, and we write $E = \{e_i\}_{i=1}^n$ and $U = \{u_i\}_{i=1}^n$. The $n = n_A + n_B$ associations are partitioned into equally sized groups A and B; each association in group $g \in \{A, B\}$ has query probability p_g, with $p_A = \rho p_B$ and $n_A p_A + n_B p_B = 1$. Equation (19) is the special case $q_i = q_{g(i)}$.

On input x_i, the model scores each candidate output y_k and assigns softmax probabilities

$$z_{ik} = u_k^\top W e_i, \qquad P_W(y_k \mid x_i) = \frac{\exp(z_{ik})}{\sum_{j=1}^n \exp(z_{ij})}.$$

The CE loss is

$$\ell_i(\mathbf{q}; E, U) = -\ln P_W(y_i \mid x_i) = \ln \left(1 + \sum_{k \neq i} e^{z_{ik} - z_{ii}} \right). \tag{29}$$

The frequency-weighted CE averaged over embedding realizations E and U is

$$\mathcal{L}(\mathbf{q}) = \mathbb{E}_{E,U} \left[\sum_{i=1}^{n} p_{g(i)} \ell_i(\mathbf{q}; E, U) \right]. \tag{30}$$

I.2 NUMERICAL OPTIMIZATION

For each $d_{\mathrm{AM}} \in \{100, 105, \ldots, 725\}$, we draw 128 independent pairs of embedding tables (E, U). Using $n_A = n_B = 4096$ queries per group, we estimate $\mathcal{L}(q_A, q_B)$ and minimize it via damped Newton's method jointly over (q_A, q_B):

$$(\widehat{q}_A, \widehat{q}_B) \in \operatorname*{arg\,min}_{q_A, q_B \geq 0} \widehat{\mathcal{L}}(q_A, q_B). \tag{31}$$

I.3 GROUP-MEAN WEIGHTS DO NOT INCREASE THE LOSS

We show that for any weight vector $\mathbf{q}$, replacing every q_i by the mean of its group gives a loss no larger than $\mathcal{L}(\mathbf{q})$. Two properties of Equation (30) are needed.

First, $\mathcal{L}$ is convex in $\mathbf{q}$. The logits $z_{ik} = u_k^\top W e_i = \sum_j q_j \, (u_k^\top u_j)(e_j^\top e_i)$ are linear in $\mathbf{q}$, so each $\ell_i = \ln \sum_k e^{z_{ik}} - z_{ii}$ is a log-sum-exp of linear functions minus a linear function and is convex in $\mathbf{q}$. A nonnegative combination and an expectation of convex functions is convex.

Second, $\mathcal{L}$ is invariant under permutations within a group. Let π be a permutation of $\{1, \ldots, n\}$ that maps A to A and B to B, and let $(\pi\mathbf{q})_i = q_{\pi(i)}$. Relabeling the associations by π leaves the query weights $p_{g(i)}$ unchanged and maps the embedding tables (E, U) to tables with the same distribution, since the embeddings are i.i.d. Hence $\mathcal{L}(\pi\mathbf{q}) = \mathcal{L}(\mathbf{q})$.

Let G be the set of all such permutations. Averaging $\pi\mathbf{q}$ over $\pi \in G$ gives the vector $\bar{\mathbf{q}}$ whose entries are the group means $\bar{q}_A = \frac{1}{n_A} \sum_{i \in A} q_i$ and $\bar{q}_B = \frac{1}{n_B} \sum_{i \in B} q_i$. By convexity and invariance,

$$\mathcal{L}(\bar{\mathbf{q}}) = \mathcal{L}\Big(\tfrac{1}{|G|} \sum_{\pi \in G} \pi\mathbf{q}\Big) \leq \tfrac{1}{|G|} \sum_{\pi \in G} \mathcal{L}(\pi\mathbf{q}) = \mathcal{L}(\mathbf{q}). \tag{32}$$

The minimum of $\mathcal{L}$ over $\mathbf{q} \geq 0$ is therefore attained on the two-parameter family $W(q_A, q_B; E, U)$.

I.4 AMPLITUDE OF THE CROSS-TALK

For a query $i \in B$ and a competing label $k \in A$, the logit margin decomposes as

$$z_{ik} - z_{ii} = \underbrace{-\, q_B(1 - u_k^\top u_i)}_{\text{correct-label advantage}} + \underbrace{\, q_A(1 - u_k^\top u_i) e_k^\top e_i}_{\text{cross-talk from association } k} + \underbrace{\sum_{j \notin \{i,k\}} q_{g(j)} (u_k - u_i)^\top u_j (e_j^\top e_i)}_{\text{cross-talk from other associations}}. \tag{33}$$

We compute the root-mean-square amplitude of each term in Equation (33) for a query $i \in B$ and a competing label $k \in A$, with $n_A = n_B = n/2$. For independent vectors e, e' uniform on the unit sphere in $\mathbb{R}^{d_{\mathrm{AM}}}$, $\mathbb{E}[e^\top e'] = 0$ and $\mathbb{E}[(e^\top e')^2] = 1/d_{\mathrm{AM}}$, and for a fixed vector a, $\mathbb{E}[(a^\top e)^2] = \|a\|^2 / d_{\mathrm{AM}}$.

The correct-label advantage $-q_B(1 - u_k^\top u_i)$ has mean square $q_B^2 \, \mathbb{E}[(1 - u_k^\top u_i)^2] = q_B^2(1 + 1/d_{\mathrm{AM}})$, so its RMS amplitude is q_B up to a factor $\sqrt{1 + 1/d_{\mathrm{AM}}}$.

The cross-talk from association k is $q_A(1 - u_k^\top u_i)\, e_k^\top e_i$. Since E and U are independent, its mean square is $q_A^2(1 + 1/d_{\mathrm{AM}})/d_{\mathrm{AM}}$.

The cross-talk from another association $j \notin \{i, k\}$ is $q_{g(j)} \, (u_k - u_i)^\top u_j \, (e_j^\top e_i)$. Conditioning on u_k, u_i and using $\mathbb{E}\|u_k - u_i\|^2 = 2$, its mean square is $q_{g(j)}^2 \cdot \frac{2}{d_{\mathrm{AM}}} \cdot \frac{1}{d_{\mathrm{AM}}} = 2q_{g(j)}^2 / d_{\mathrm{AM}}^2$. Terms with different j are uncorrelated, because $\mathbb{E}[(e_j^\top e_i)(e_{j'}^\top e_i) \mid e_i] = e_i^\top \, \mathbb{E}[e_j] \, \mathbb{E}[e_{j'}]^\top e_i = 0$ for $j \neq j'$. Summing over the $n_A - 1$ terms in A and $n_B - 1$ terms in B gives a total mean square of approximately $2(n_A q_A^2 + n_B q_B^2)/d_{\mathrm{AM}}^2 = n(q_A^2 + q_B^2)/d_{\mathrm{AM}}^2$.

The two cross-talk contributions are also uncorrelated, so the total cross-talk has mean square $q_A^2(1 + 1/d_{\mathrm{AM}})/d_{\mathrm{AM}} + n(q_A^2 + q_B^2)/d_{\mathrm{AM}}^2$. For $n \gg d_{\mathrm{AM}}$ the second term dominates, and dividing by the RMS of the correct-label advantage gives

$$\frac{\mathrm{RMS(cross\text{-}talk)}}{\mathrm{RMS(correct\text{-}label\ advantage)}} \approx \frac{\sqrt{n}}{d_{\mathrm{AM}}} \, \frac{\sqrt{q_A^2 + q_B^2}}{q_B} = \frac{\sqrt{n}}{d_{\mathrm{AM}}} \sqrt{1 + \left(\frac{q_A}{q_B}\right)^2}. \tag{34}$$

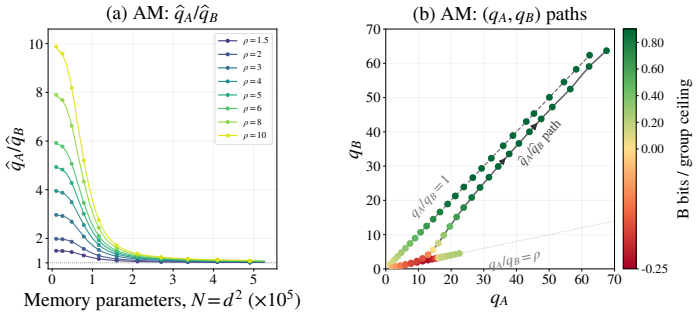


Figure 15: Optimal memory weights across dimensions for $\rho = 5$. Each point minimizes the frequency-weighted CE at one dimension, either freely (middle path) or subject to $q_A/q_B = 5$ (bottom) or $q_A/q_B = 1$ (top). Colors show group-B information normalized by its ceiling.

The cross-talk has zero mean, so when this ratio is of order one the margin $z_{ik} - z_{ii}$ is positive for a non-negligible fraction of the n_A competing labels, and each such label contributes $e^{z_{ik}-z_{ii}} > 1$ to the loss in Equation (29).

I.5 Robustness to optimizer and batch size

We assess whether the allocation pattern changes with the optimizer or batch size. We compare projected SGD without momentum and projected Adam with $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\epsilon = 10^{-8}$, using $n_A = n_B = 4096$, $\rho = 5$, and batch sizes $128, 512, 2048, 8192$. Only the two shared coefficients (q_A, q_B) are optimized. Each fit uses four fixed, independently sampled full pairs of spherical embedding tables (E, U), and every epoch visits all 8192 queries in each table. Group-A and group-B queries receive weights $2\rho/(\rho+1)$ and $2/(\rho+1)$, respectively, yielding the objective $(\rho\ell_A + \ell_B)/(\rho+1)$. All associations remain in memory, and every softmax includes all candidate outputs.

Each setting is trained for 120 epochs, with initial learning rates $0.5d_{\mathrm{AM}}^2/n$ for SGD and $0.05d_{\mathrm{AM}}^2/n$ for Adam, where $n = 8192$, followed by cosine decay to 5% of the initial rate. Comparisons share embedding tables and shuffle seeds. Final coefficients are evaluated on all queries in four fresh embedding-table pairs, and results average three independent repetitions. Equal epochs match full-data passes rather than optimizer-update counts; a batch of 8192 covers one embedding table rather than all four tables simultaneously.

Both optimizers and all four batch sizes exhibit low-frequency suppression followed by recovery. The minimum normalized group-B information is approximately -0.179 near $d_{\mathrm{AM}} = 224$. The maximum difference across batch sizes is 0.0028 for SGD and 0.0081 for Adam. Thus, in this tested setting, the interference pattern is not specific to one optimizer or batch size. This conclusion concerns the two-coefficient outer-product family and the stated training budget, rather than unrestricted training of the full matrix W.

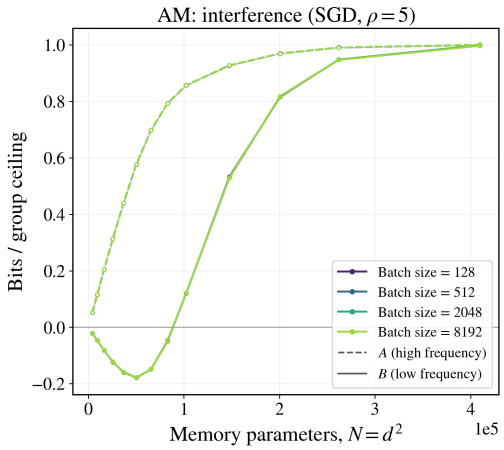


Figure 16: AM interference under SGD with $n_A = n_B = 4096$ and $\rho = 5$. Colors indicate query batch size; dashed and solid curves show groups A and B. Every epoch uses all queries in each of four fixed full embedding tables, with all associations retained in memory and all outputs included in the softmax. Each setting is trained for 120 epochs with the same dimension-dependent learning-rate schedule. Final coefficients are evaluated using every query on four independent full embedding tables. Curves average three independent repetitions, with shading showing one standard deviation. The vertical axis is $1 - \mathrm{CE}_g / \ln 8192$.

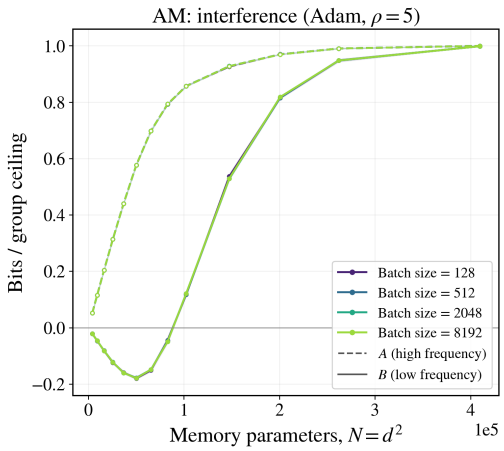


Figure 17: AM interference under projected Adam with $n_A = n_B = 4096$ and $\rho = 5$. Colors indicate query batch size; dashed and solid curves show groups A and B. Every epoch uses all queries in each of four fixed full embedding tables, with all associations retained in memory and all outputs included in the softmax. Each setting is trained for 120 epochs with the same dimension-dependent learning-rate schedule. Final coefficients are evaluated using every query on four independent full embedding tables. Curves average three independent repetitions, with shading showing one standard deviation. The vertical axis is $1 - \mathrm{CE}_g / \ln 8192$.