ATTENTION MEAN FIELDS PREDICT AVERAGE REPRESENTATION DYNAMICS AND ILLUMINATE EARLY TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

A language model's representation geometry is not predetermined; it evolves as computation proceeds. Each layer reshapes geometry in part through the effects of attention, which updates each token's representation with information drawn from its context. A faithful account of a model's geometry must capture that dynamic process. To that end, we introduce a mean-field analysis of attention. The average attention a head pays from one token to another defines a kernel that carries representations forward through the model and describes how geometry is transformed. Conditioned on a substantial corpus, the kernel predicts, from the input embeddings and frozen weights alone, the trajectory of each token's mean representation. These predictions are accurate across models spanning GPT-2 to Qwen-3-14B. We show that model-oblivious alternatives, including one built from token co-occurrence, are all poorer predictors. We then develop an attention mean field with more fine-grained conditioning, and demonstrate its value. Specifically, we show that early in training, on real text, replacing every attention head's output with its context-conditional mean field leaves the model's loss unchanged. This effect occurs across three scales of Pythia as well as in OLMo-2-7B. As training progresses through the onset of induction, model behavior and mean-field behavior diverge, with representations becoming contextualized and attention becoming correlated with transported values.

1 INTRODUCTION

What happens inside a language model, *on average*, as it processes a large text corpus? This is an important question from a model interpretability standpoint: when studying the behavior of an attention head, a natural baseline can be useful. For example, when a head attends to a particular token and writes a particular vector, it is useful to know how much of that behavior is typical for those token types and how much is specific to the present context.

In this paper we address this need by developing empirical methods for studying the average-case behavior of attention heads. We construct *mean-field approximations* of a head's behavior by averaging the attention of the head over a corpus. Importantly, this is a corpus-averaged statistic of the trained model's behavior, not a statistic of the corpus alone. Its entries summarize how the model routes attention when processing text. The approximation yields a kernel at the *type* level – a characterization of a head's behavior that is parameterized by pairs of types (the entries in the model's token vocabulary). The kernel compresses a head's context-dependent routing into a type-level operator that can be estimated in a single corpus pass, without optimization or fitted parameters.

We show two uses for the resulting approximation. First, it yields a surprisingly accurate, compact description of the evolution of average type representations, allowing drastic simplification of the study of these objects. Considerable work has focused on understanding representation geometry (Karkada et al., 2026; Nava & Wyart, 2026; Engels et al., 2025; Gurnee & Tegmark, 2024; Park et al., 2024), including studies arguing that is important for understanding model behavior and steering (Lu et al., 2026; Wurgaft et al., 2026). A simple description of how geometry evolves over the layers of a model can inform that work.

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.

Second, the mean field yields a highly accurate description of *model behavior in early training*: mean-field attention can replace the model's attention on frequent query types at almost no change in loss. This is followed by a transition where context-specific attention becomes behaviorally important under our intervention, and studying the evolution of the mean field reveals a major reorganization at the same point.

2 Related Work

Low-order corpus statistics and model behavior. The premise that model behavior can reflect low-order corpus statistics predates transformers. Pennington et al. (2014) learn embeddings directly from global co-occurrence counts, and Levy & Goldberg (2014) show that skip-gram with negative sampling implicitly factorizes a shifted PMI matrix. Within the transformer literature, Elhage et al. (2021) show that attention can reflect corpus-average bigram and skip-trigram statistics. This body of work motivates the hypothesis that a low-order corpus-average operator should capture a substantial share of model behavior. Our paper makes this hypothesis precise and then tests it directly, finding it borne out for the layerwise evolution of type centroids and for early training.

Representation geometry. A related line derives representation geometry from those statistics. Karkada et al. (2026) argue, that symmetries in pairwise word co-occurrence determine the circular and continuous geometries observed in word embeddings and language model representations. Nava & Wyart (2026) extend the account to hierarchical geometry, connecting the spectrum of the co-occurrence matrix to coarse-to-fine conceptual organization. These provide insights into phenomena described by Engels et al. (2025), who find circular representations of days and months and give causal evidence that models use them, and by Gurnee & Tegmark (2024), who find linear spatial and temporal representations at multiple scales. Park et al. (2024) and Park et al. (2025) give formal accounts of that geometry, including the role of the inner product and the distinction between a representation and its linear accessibility, and Jiang et al. (2024) derive the emergence of linear semantic structure from a latent-variable model of next-token prediction. Those accounts relate corpus statistics to a *static* geometry, typically in simple models or abstractions of models. The mean field studied here describes how representation geometry is *transformed* layer by layer in real language models, and documents the strong correlation between the corpus-level attention mean-field and co-occurrence statistics.

Corpus statistics in early training. Chang & Bergen (2022) find that early predictions resemble unigram frequencies, then bigram probabilities, and become contextually nuanced only later. Further, Im et al. (2026) show analytically that early stages of learning in transformer models are driven primarily by corpus-average statistics. Our results are consistent, but go further by showing empirically that early in model training, the attention mechanism is *fully replaceable* by its mean field without affecting model performance.

Mean-field theories of attention. The term "mean field" is used in several unrelated senses in the literature. Kim & Suzuki (2024) take a mean-field limit over a distribution of model *parameters* to study optimization dynamics. Our approach has some conceptual similarity with Burger et al. (2025), Castin et al. (2025), Geshkovski et al. (2025) and Rigollet (2025), insofar as we treat tokens as interacting particles driven by a mean-field abstraction of the attention mechanism. However, those papers derive continuum or infinite-token and infinite-depth limits, while our work is explicitly based on finite-sample statistics. To our knowledge, ours is the first mean-field treatment measured directly on and validated against trained language models.

3 Mean Field Analysis of Attention

3.1 Notation

In the attention mechanism, a head at any layer operates on an input$\{\mathbf{x}_i\}_{i=1}^m$, with $\mathbf{x}_i \in \mathbb{R}^D$ the head's *representation* of the ith input token: a running vector that the head reads and updates. The head computes a separate update for each position in the input, so we say that each position defines one *context*, denoted X. The context at position $j \in \{1, \ldots, m\}$ is the prefix $X = \mathbf{x}_{1:j}$ of the input; the attention mechanism uses the representation $\mathbf{x}_j$ as the *query* and the representations $\mathbf{x}_{1:j}$ as the *keys* for that context. The mechanism computes a distribution over the keys, $p_q(\mathbf{x}_i) \geq 0$ with

$\sum_{i=1}^{j} p_q(\mathbf{x}_i) = 1$, and reads a *value* $v(\mathbf{x}_i) \in \mathbb{R}^D$ from each key $\mathbf{x}_i$. The vector the head adds to $\mathbf{x}_j$, the head's *contribution* to $\mathbf{x}_j$, is[1]

$$\mathbf{z}_j \;=\; \sum_{i=1}^{j} p_q(\mathbf{x}_i)\, v(\mathbf{x}_i). \tag{1}$$

Figure 1 illustrates this contribution: the value map defines a vector field over representation space; the head forms an attention-weighted sample of that field that it adds to the query.

Each of the model's input tokens has a *token type*, its entry in the model's *vocabulary* $\mathcal{T}$. $\mathcal{T}$ is the set of all input tokens — such as 'October', or 'was'. Each representation $\mathbf{x}_i$ inherits a token type, that traces back to the type of the model's ith input token. Each occurrence of a given token (e.g., 'October') in the corpus shares that type but has its own representation vector. We write $T(\mathbf{x})$ for the token type of $\mathbf{x}$, and shorten *token type* to *type* where no ambiguity arises.

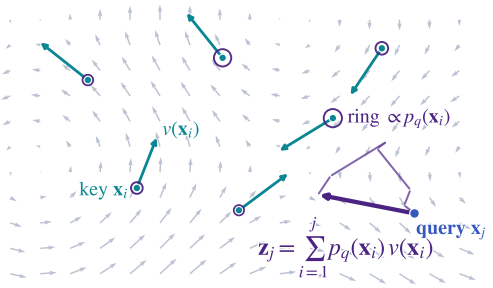

Figure 1: A head's contribution on a single context (Equation 1). The value map assigns a vector to each key; the head takes these vectors, weights them by the query's attention, and delivers their sum to the query as the contribution $\mathbf{z}_j$.

Our analysis will focus primarily on the contribution $\mathbf{z}$ that a given head makes to a token's representation. We consider averages over various conditioning sets; we always denote an average by a bar with the conditioning set in the subscript, e.g., $\overline{\mathbf{z}}_s$ is the average contribution given query type s. The same convention names other averaged quantities, such as $\overline{\mathbf{v}}_{tX}$, the average value over type-t keys in a context X. A corpus-level average carries only type subscripts, while a context-level average also carries the context X. Our mean field estimates of $\overline{\mathbf{z}}$ will always be denoted $\hat{\mathbf{z}}$. Finally, when the particular head (ℓ, h) matters, we note the layer and the head in the superscript, e.g., as $\mathbf{z}^{\ell h}$.

The contribution $\mathbf{z}$ (Equation 1) is what a single head computes on one context. The mean-field approach describes the contribution *on average*, over a set of contexts that share some structure. Different choices of that conditioning set result in mean fields at different resolutions. We develop those in the next two subsections.

3.2 The Corpus Mean Field

A corpus is a collection of inputs (prompts), each of which passes through the model independently. Because many positions across the corpus may share the same type, each type is typically realized by many representations at every layer. At a given layer, the type's *centroid*, $\mathbf{c}_s = \mathbb{E}[\mathbf{x} \mid T(\mathbf{x}) = s]$, is the mean representation over all positions of type s across the inputs of the corpus. Our first mean field is concerned with how each head updates the centroid for type s. This is achieved by considering all contexts whose query type is s, ie, $T(q) = T(\mathbf{x}_j) = s$, and applying (1). Averaging over the corpus, the amount the head adds to the type-s centroid is

$$\overline{\mathbf{z}}_s = \mathbb{E}_s \left[\sum_{i=1}^{j} p_q(\mathbf{x}_i)\, v(\mathbf{x}_i) \right], \tag{2}$$

where $\mathbb{E}_s$ denotes the average over all corpus contexts whose query is type s, ie, $\mathbb{E}_s \overset{\text{def}}{=} \mathbb{E}_{\{X : T(\mathbf{x}_j) = s\}}$.

Attention is per-position, but several positions may share a type, so we can rewrite (2) as a sum over *token types* rather than positions, pooling positions of the same type. For a query of type s and a context X, let $A_{st} = \sum_{i:\, T(\mathbf{x}_i) = t} p_q(\mathbf{x}_i)$, the total attention placed on positions of type t (defined to

[1] The mean-field approach does not depend on the specifics of how $p()$ and $v()$ are realized. As an example, in a typical implementation, tokens are first passed through a layer norm, giving $\tilde{\mathbf{x}} = \mathrm{LN}(\mathbf{x})$, and then $p()$ and $v()$ are computed as $p(\tilde{X}) = \mathrm{Softmax}(\{\tilde{\mathbf{x}}_j^\top W_Q^\top W_K \tilde{\mathbf{x}}_i\}_{i=1}^{j})$ and $v(\tilde{\mathbf{x}}) = W_O W_V \tilde{\mathbf{x}}$. Different models will vary in these details, and we consider models with different implementations of $p()$ and $v()$ in our experiments.

be 0 when a token type is absent), and let

$$\overline{\mathbf{v}}_t = \frac{\sum_{i:\, T(\mathbf{x}_i)=t} p_q(\mathbf{x}_i)\, v(\mathbf{x}_i)}{\sum_{i:\, T(\mathbf{x}_i)=t} p_q(\mathbf{x}_i)}$$

be the attention-weighted mean of the value vectors $v(\mathbf{x}_i)$ over those positions. Then the contribution the head makes to the type-s centroid is, by linearity of expectation,

$$\overline{\mathbf{z}}_s = \mathbb{E}_s\left[\sum_{t\in\mathcal{T}} A_{st}\overline{\mathbf{v}}_t\right] = \sum_{t\in\mathcal{T}} \mathbb{E}_s[A_{st}\overline{\mathbf{v}}_t] = \sum_{t\in\mathcal{T}} \left[\mathbb{E}_s[A_{st}]\,\mathbb{E}_s[\overline{\mathbf{v}}_t] + \mathrm{Cov}_s(A_{st}, \overline{\mathbf{v}}_t)\right]. \tag{3}$$

Mean-field approximation. To construct a mean-field estimate $\hat{\mathbf{z}}_s$ for $\overline{\mathbf{z}}_s$, we introduce two *mean-field assumptions.* We emphasize that we do not assert that these assumptions hold *in general*; rather, they are useful in analyzing the behavior of the model's centroids specifically. Additionally, our subsequent analyses will concern when mean-field assumptions *do not* hold.

Assumption 1. Attention-Value independence. $\mathrm{Cov}_s(A_{st}, \overline{\mathbf{v}}_t) \approx 0$. This assumption posits that attention is approximately uncorrelated with the *value* a head moves. Note, however, that attention may still correlate with the type's *representation*.

Assumption 2. Negligible curvature. $\mathbb{E}_s[\overline{\mathbf{v}}_t] \approx v(\mathbf{c}_t)$. This assumption posits that averaging the value map over a type's occurrences is approximately the same as applying it once at the type's centroid, as it would be if the value map were linear.

Assumption 2 involves two subtleties. First, the sets over which the expectation is taken differ on the two sides of the equation: $\mathbb{E}_s[\overline{\mathbf{v}}_t]$ is the mean type-t value across type-s contexts, whereas $\mathbf{c}_t$ is type t's global centroid, based on all occurrences of type t. Second, typical value maps are nonlinear (e.g., due to the layer norm), so the approximation does not hold exactly. The discrepancy due to nonlinearity is the value map's *Jensen gap.*[2] We evaluate the impact of the value map's Jensen gap, and show that it is small, in Appendix C.

Because of the central role of $\mathbb{E}_s[A_{st}]$, we refer to it as the (corpus) *attention mean field* and denote it by P, with entries $P_{st} = \mathbb{E}_s[A_{st}]$, the mean attention a type-s query places on type t across the corpus. Applying this substitution and the two assumptions to (3) yields the head's *mean-field contribution* $\hat{\mathbf{z}}_s$, an approximation to the true averaged contribution $\overline{\mathbf{z}}_s$:

$$\hat{\mathbf{z}}_s = \sum_{t\in\mathcal{T}} P_{st}\, v(\mathbf{c}_t) \;\approx\; \overline{\mathbf{z}}_s. \tag{4}$$

Figure 2 illustrates the update: the per-context attention of Figure 1 is replaced by the fixed kernel P, applied to the value map at the type centroids.

Equation (4) represents a drastic simplification of the model's processing. Rather than context-specific attention applied to each representation over the entire corpus, the mean-field estimate applies a single constant linear map P_{st} and the value maps of the type centroids $v(\mathbf{c}_t)$ to derive $\hat{\mathbf{z}}_s$. This estimate only requires precomputing P once from a corpus.

Each row of P is an average of probability distributions, so P is row-stochastic: the corpus-conditional mean field is itself a valid attention pattern, one we can use in a replacement model. This is the form the rollout in §4 applies at each head and layer.

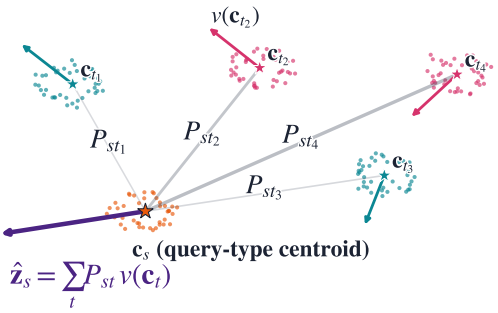

Figure 2: The corpus mean-field update (Equation 4). The mean field applies the value map once at each type's centroid $\mathbf{c}_t$ and combines the results with kernel weights P_{st}.

[2]The Jensen gap is a general phenomenon: for any nonlinear map, it is the difference between applying the map at an average and averaging the map's outputs. The same gap arises later for the MLP (§4).

3.3 Conditional Mean Fields

We now define two more narrowly conditioned mean fields: the (s,t)-conditional mean field, and the context-conditional mean field used in §5. To simplify the exposition here, we take each type to occur at most once per context; Appendix A gives the exact formulas for the general case.

The (s,t)-conditional mean field. Here we condition on a type t being present in a context having a query of type s. Write $\mathbb{E}_{st}$ for the conditional expectation over such contexts and $\Pr_s(t)$ for the probability that t is present in a context having query type s. Define $\boldsymbol{\mu}_t$ to be the corpus average value map of type t acting as a key, ie, $\boldsymbol{\mu}_t = \mathbb{E}[v(\mathbf{x}_i) \mid T(\mathbf{x}_i) = t]$. This is computed directly from the corpus in a fashion similar to the centroid $\mathbf{c}_t$: condition on type t and average over its corpus positions, here averaging the frozen value map's output $v(\mathbf{x})$ rather than the representation $\mathbf{x}$. As in the corpus mean field case, we define the (s,t)-conditional mean field $W_{st} = \mathbb{E}_{st}[A_{st}]$, related to the corpus kernel by $P_{st} = W_{st}\ \Pr_s(t)$.

The context-conditional mean field. Next, we use a similar strategy to make an average case prediction for a particular context X. We assume that X has query type s. We denote the contribution made to query type s when it has context X as $\mathbf{z}_{sX}$, the value map from type t in that context as $\boldsymbol{\mu}_{tX}$, the attention paid by s to t conditioned on the context being X as W_{stX}, and the analog of $\Pr_s(t)$ for context X as $\Pr_X(t)$. Two of these quantities degenerate below because X is a single fixed context: the average W_{stX} is the actual attention A_{stX} the head pays from s to t in X, and $\Pr_X(t)$ is 1 for the types in X and 0 otherwise. This gives an exact per-t decomposition of the contribution, for the single context X:

$$\mathbf{z}_{sX} = \sum_{t \in \mathcal{T}} W_{stX}\, \boldsymbol{\mu}_{tX}\, \Pr_X(t) \;= \sum_{t \in X} A_{stX}\, \boldsymbol{\mu}_{tX} \tag{5}$$

To construct a mean field analog, we introduce two mean-field assumptions.

Assumption 3. Value-context independence. This posits that $\boldsymbol{\mu}_{tX} \approx \boldsymbol{\mu}_t$. That is, the value map of type t in context X is well-approximated by its corpus average. This is an important simplification; the difference between $\boldsymbol{\mu}_{tX}$ and $\boldsymbol{\mu}_t$ can be thought of as the *contextualization* of the value map of t induced by its context X.

Assumption 4. Proportionality of context attention. This assumption posits that for each s, the actual attention values A_{stX} are proportional to W_{st} across the types present in X. The constant of proportionality is then fixed by the requirement that a head's attention over its context sums to one.

Assumption 4 lets our mean-field kernel $\hat{A}_{stX}$ estimate the real attention A_{stX} by renormalizing W_{st} over the types in X (the general case of repeated types is treated in Appendix A):

$$\hat{A}_{stX} = \frac{W_{st}}{\sum_{t' \in X} W_{st'}} \;\approx\; W_{stX} \;=\; A_{stX}. \tag{6}$$

Continuing from (5), we substitute the corpus average $\boldsymbol{\mu}_t$ for $\boldsymbol{\mu}_{tX}$ (Assumption 3) and $\hat{A}_{stX}$ for the attention A_{stX} (Equation 6) to derive our *context-conditional* mean field estimator:

$$\hat{\mathbf{z}}_{sX} = \sum_{t \in X} \hat{A}_{stX}\, \boldsymbol{\mu}_t \;\approx\; \mathbf{z}_{sX}. \tag{7}$$

Additional analysis justifying the use of $\hat{A}$ as the context conditional mean field is in Appendix B.

4 Mean Field Prediction of Representation Evolution

As a first application of mean-field attention analysis, we use the mean field to predict how a type's centroid evolves across the layers of a model, iterating the update of Equation (4) through the network from the frozen weights alone.

4.1 Method: Open-Loop Rollout

To construct the model's mean-field replacement, we run a purely open-loop computation, analogous to a dynamical system, shown in Algorithm 1. The process starts with input embeddings (the layer-0

centroids), and iterates the mean-field update layer by layer: at each layer the predicted centroids are passed to the next, where the process repeats. Tracking these type centroids across layers is how we observe the evolution of a type's representation geometry through the network.

Algorithm 1: Open-loop mean-field rollout

input : layer-0 centroid representations $\{\hat{\mathbf{c}}_s^0\}$ (input embeddings); mean fields $P^{\ell h}$; frozen weights

output: predicted centroid representations $\{\hat{\mathbf{c}}_s^\ell\}$ for each type s and layer ℓ

for *each layer* $\ell = 0$ **to** $L-1$ **do**
  for *each type* $s \in \mathcal{T}$ **do**
    for *each head* h **do**
      $\hat{\mathbf{z}}_s^{\ell h} \leftarrow \sum_{t\in\mathcal{T}} P_{st}^{\ell h}\, v(\hat{\mathbf{c}}_t^\ell)$; `// per-head contribution, (4)`
    $\mathbf{m}_s^\ell \leftarrow \hat{\mathbf{c}}_s^\ell + \sum_h \hat{\mathbf{z}}_s^{\ell h}$; `// add attention contributions`
    $\hat{\mathbf{c}}_s^{\ell+1} \leftarrow \mathbf{m}_s^\ell + \mathrm{MLP}^\ell(\mathrm{LN}(\mathbf{m}_s^\ell))$; `// apply MLP to the average`

The rollout is a self-contained computation of the network's average-case behavior for each type, built only from the mean-field kernel and the frozen weights, with no forward pass on any text. Analogous to the mean-field substitution for attention (4), we apply the MLP only once, at the post-attention average. This is again a drastic simplification of what the MLP does, as it avoids averaging many executions of the MLP over each specific type-s context. The centroid trajectory that the rollout produces is a prediction; we demonstrate its accuracy in the remainder of this section.

4.2 Basic Results and Ablations

We now evaluate the performance of mean-field prediction of type centroids as obtained via Algorithm 1. Our evaluation spans five models: GPT-2 Small (Radford et al., 2019), Gemma-2 2B (Gemma Team, 2024), Llama-3.1-8B (Grattafiori et al., 2024), Llama-3.2-3B (Meta AI, 2024), and Qwen-3-14B (Yang et al., 2025). The mean field P and the type centroids are estimated as defined in §3.2, on the $N = 1000$ most frequent types, from 1.5–2M tokens of OpenWebText (Gokaslan et al., 2019); details are in Appendix D. These types cover 64–68% of corpus positions and 67–72% of non-BOS attention, nearly independent of vocabulary size (Appendix E). Held-out evaluation uses a document-interleaved split: alternating documents are assigned to two disjoint halves of the corpus, the kernel and all corpus statistics are estimated on one half, and the true centroids used for scoring are measured on the other (GPT-2 and Gemma-2; the Qwen-3 and Llama results use a single split, as noted where they appear). In Appendix F we show that our results here generalize to FineWeb, WikiText, and pre-1919 books, including across corpora when the training and testing corpora differ.

We assess prediction quality with four metrics: centered cosine similarity, RSA, relative Euclidean error, and norm ratio (all defined in Appendix G). Each is computed per type and then averaged. Relative error is computed as a ratio of means (mean Euclidean error over mean distance to the entire type set's centroid) so that types lying near the centroid do not arbitrarily inflate it. All are computed after mean-centering the representation cloud. Centering is necessary because the representations are strongly anisotropic: a few outlier dimensions dominate similarity measures while contributing little to behavior (Timkey & van Schijndel, 2021; Ethayarajh, 2019). Appendices H and I provide details of anisotropy and analyze the effects of 'rogue' dimensions and attention sinks in detail.

Mean field predictions of type centroids prove highly accurate. Figure 3 shows cosine similarity for GPT-2, Gemma-2 and Qwen-3; the other three metrics are consistent (Appendix G). The Llama models are the exception, treated in Appendix J (§4.3 explains why). In GPT-2, centered cosine exceeds 0.9 through the middle layers and stays above 0.75 in the last; in Qwen-3 and Gemma-2 it stays well above 0.75 through most of the network. Given the high dimension of the representation space, these reflect very close alignment with the true centroids.

To assess the importance of the mean-field kernel, we compare against a `context-free` baseline (each type propagated through the network in isolation) and against alternatives that each simplify one part of the kernel: `cooc` replaces attention with a corpus co-occurrence matrix; `P0only` and `Pavg` remove the kernel's variation across depth (reusing the layer-0 kernel, or the depth-averaged kernel). The mean field outperforms every alternative (Figure 3): co-occurrence statistics do not

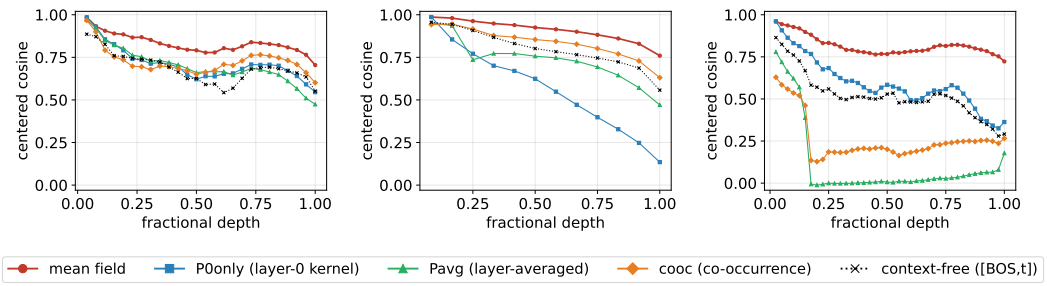

Figure 3: Mean-field predictions are accurate and superior to alternative kernels. Average per-type centered cosine distance. Left: Gemma-2, Middle: GPT-2, Right: Qwen-3-14B.

suffice, the kernel's variation across depth carries important content, and accurate prediction requires type-level distinctions among both queries and keys, increasingly so with depth. Full details are in Appendix G.

4.3 Understanding Approximation Errors

Algorithm 1 incorporates two approximations at every layer: the mean-field prediction of each head's contribution $\hat{\mathbf{z}}_s^{\ell h}$, and the application of the MLP once at the resulting centroid. Only the first is a property of mean-field attention; the second is a property of the rollout as a way of propagating centroids. The prediction is good at every layer of every model: across all five models, cross-layer mean centered cosine lies between 0.78 and 0.87, and the norm ratio stays near one (cross-layer means 0.91 to 1.24), so the mean field does well at recovering both the direction and the scale of what each head writes.

A per-layer decomposition shows that the main source of rollout inaccuracy is not mean-field prediction, but rather is introduced by the MLPs (Appendix G.3): the dominant per-layer source of error is the MLP evaluated at the centroid. The significance of the gap is nuanced, as we explain in Appendix K. The same decomposition explains performance on the two Llama models, whose rollouts predict centroid evolution distinctly less well than the others (full details in Appendix J). The mean-field *attention* prediction for Llama is as accurate as for the other models; the error is in the MLP-at-centroid step, whose Jensen gap is larger because many of Llama's centroids, especially in early layers, sit where the MLP is more sharply curved.

5 The Mean Field as a Training Waypoint

We next show that mean-field analysis provides insight into a model's training dynamics. Specifically we find that early in training, the model's computation *reduces to* its mean field, and then deviates as contextual computation develops. The resulting dynamics reveal which weight changes drive early-stage learning. We demonstrate this behavior on three scales of Pythia (Biderman et al., 2023) across 19 checkpoints spanning each model's training run. We then demonstrate the same behavior in OLMo-2-7B (OLMo Team, 2024), a second architecture trained on different data, confirming the effect is not specific to Pythia.

5.1 Replacing Attention with its Mean Field

To give a mean-field approximation of the attention the model computes for each context X we use the context-conditional mean field of §3.3, Equation (7), substituting $\hat{\mathbf{z}}_{sX}^{\ell h}$ for each $\mathbf{z}_{sX}^{\ell h}$ throughout the forward pass and leaving the rest of the model's computation unchanged.

We compute the ingredients of (7) ($\hat{A}_{st}^{\ell h}$ and $\boldsymbol{\mu}_t^{\ell}$) from the same corpus as §4.2 and run the model on a disjoint $100{,}000$-token sample. The substitution covers the top $N = 1000$ types (65% of Pythia's positions, Appendix E; remaining query types keep the model's own computation); Appendix L shows the effects are unchanged at $N = 2000$ and 4000 (72% and 80% coverage).

We report two diagnostics: ΔL and D_{MF}. Write $p(\cdot \mid X)$ for the model's next-token distribution on context X, and $p_{\mathrm{MF}}(\cdot \mid X)$ for the distribution under the substitution above. Writing y for the token

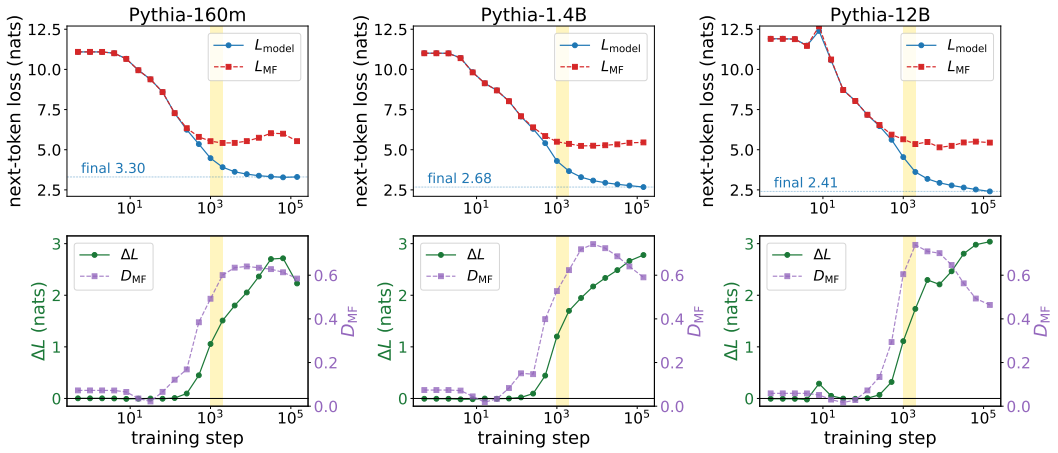

Figure 4: Pythia models side by side (160m, 1.4B, 12B). **Top:** the model's loss (blue) and the iterated mean field's loss (red) across training (log scale); the yellow band marks the induction-formation window. **Bottom:** the gap ΔL (green, left axis) and the deviation D_{MF} (purple, right axis).

that actually follows X,

$$L_{\text{model}} = -\log p(y \mid X), \qquad L_{\text{MF}} = -\log p_{\text{MF}}(y \mid X), \qquad \Delta L = L_{\text{MF}} - L_{\text{model}}. \tag{8}$$

The paired loss gap ΔL is the loss the model avoids by doing something the mean field does not. We consider this a measure of how much of the model's prediction is contextually driven.

The *mean deviation* at a sampled position (s, X), $D_{sX} = \text{mean}_{\ell,h}\left(1 - \cos(\mathbf{z}_{sX}^{\ell h}, \hat{\mathbf{z}}_{sX}^{\ell h})\right)$, measures how far the per-head contributions move from their corpus-averaged counterparts (note that centering is not needed here). We report $D_{\text{MF}} = \text{mean}_{s,X}\ D_{sX}$, its average over the sampled positions.

5.2 Models Reduce to Their Mean Field, Then Depart from It

Figure 4 plots the experimental results for the three scales of Pythia. The top row of each panel plots model loss against mean field loss; the bottom row plots ΔL and D_{MF}. The yellow band marks the induction window (roughly 2–5 billion tokens), where induction heads are reported to form in Pythia (Tigges et al., 2024). The substitution's effect divides into three phases around this window.

Before the induction window, the model reduces to its mean field: ΔL stays within a few millinats of zero and D_{MF} near its floor, so the substitution costs essentially nothing. This is not for lack of learning: the losses themselves fall by more than three nats over the same stretch. What the model learns here depends only on which tokens are present, not their order, exactly what the mean field captures (§5.3 gives the mechanistic reason).

During the window, the two diverge: from around step 128 through the induction band, ΔL and D_{MF} both rise sharply. The rise occurs at the same token count in all three models despite a 75× difference in parameters, so the transition is set by data seen rather than model size.

After the window, the mean field can no longer stand in for the model: the mean-field loss stops tracking the model's (top panels), while D_{MF} plateaus (160m) or eases from its peak (1.4B and 12B). Imposing the mean field now harms performance, as the model learns to rely on something the mean field does not supply, namely contextual processing.

The gap widens with scale: larger models reach lower true losses while their mean-field losses stay similar, so the gains from scale are context-dependent.

The effect replicates in OLMo-2. To ask whether the effect is a property of the Pythia suite or of language-model training generally, we repeated the mean field substitution on OLMo-2-7B (OLMo Team, 2024), an independent architecture trained on a different corpus (Figure 5). Early in training, OLMo behavior is explained by its mean field to the same degree as Pythia, so the effect is not unique to that suite. As training proceeds the model loss keeps falling while the mean-field loss flattens into a narrow band. The final gap ΔL is nearly the same across all three models despite

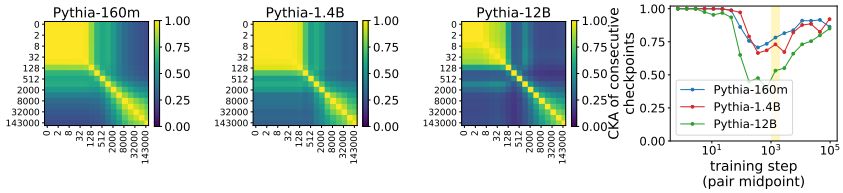

Figure 6: Changes in the mean-field kernel, measured as similarity between checkpoints: 1.0 is no change. **Left three panels:** kernel similarity between every pair of checkpoints averaged over heads. **Right:** adjacent-checkpoint kernel similarity.

an 8× difference in model size and different training data. The mean-field ceiling appears to be established by corpus statistics, not the model; it stays fixed while the model continues to improve.

5.3 Kernel Dynamics Across Training

We now consider the evolution of the kernel itself over time. Figure 6 shows how much the (s, t)-conditional mean field W changes across training, measured using linear CKA, averaged over heads. The left three panels give the kernel similarity between every pair of checkpoints; the right panel gives the similarity between adjacent checkpoints, depicting the rate at which W changes. Up through step 128 the kernel barely moves (adjacent-checkpoint similarity 1.0), even as the loss falls steeply over the same checkpoints (Figure 4): *learning in this regime is in the values, MLPs, and embeddings, not the attention pattern.* It then reorganizes at the same point the departure measures begin to rise, adjacent-checkpoint similarity dropping to its minimum through the induction band before recovering.

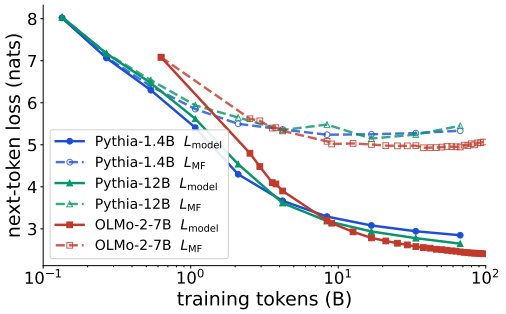

Figure 5: Pythia-1.4B, Pythia-12B and OLMo-2-7B on a common axis of training tokens in billions. Solid lines are L_{model}; dashed lines are L_{MF}, the loss when each head is replaced by its per-context mean field.

When the mean field holds in practice. The measurements in this section support the following conclusion: early in training a head's corpus-mean attention to a type is fixed by how often the type occurs, independent of the values those occurrences carry. Attention and value are therefore uncorrelated across types, which is precisely Assumption 1, attention-value independence for the corpus mean field. The assumption thus holds closely early in training as an empirical fact about Pythia (not a simplification) and stops holding once attention heads acquire contextual sensitivity.

Conclusions

Mean-field analysis is one half of a larger picture. When it is accurate, the mean field gives a compressed description of average representation dynamics; when it is inaccurate, the structured deviation from it identifies computation specific to the particular context, such as syntactic and positional processing, and in-context learning. Model geometry can then be read as a corpus-average component plus a context-supplied one: global structure plus local variation. In this paper we have developed the average-case side of the story; we believe that mean field deviation and its context-specific computation are an important direction for further study.

To summarize, we make three main contributions. First, we develop the conceptual framework needed to precisely describe the average-case attention of an attention head. Second, we show that, using this operator, one can make accurate predictions of average representation evolution through the layers of a model. Finally, we show that in early training stages, the behavior of both Pythia and OLMo are remarkably well-explained by mean-field substitution. We conclude that mean-field analysis offers a useful new tool for understanding representation geometry.

AI USE STATEMENT

The authors used various versions of Anthropic's Claude and OpenAI's ChatGPT models in this work. Under the authors' guidance, Claude wrote the experimental code, ran numerous experiments, drafted preliminary analyses of experimental results, and suggested follow-on experiments. In many cases, author-validated findings from those experiments appear in this manuscript. In close collaboration with the authors, Claude and ChatGPT reviewed and copy-edited portions of the manuscript for correctness, readability and concision. The research questions, experimental designs, and the interpretations reported here are the authors'. The authors take full responsibility for the content of this paper.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purber, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, 2023.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Quantizable transformers: Removing outliers by helping attention heads do nothing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Shaked Brody, Uri Alon, and Eran Yahav. On the expressivity role of LayerNorm in transformers' attention. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023.

Martin Burger, Samira Kabri, Yury Korolev, Tim Roith, and Lukas Weigand. Analysis of mean-field models arising from self-attention dynamics in transformer architectures with layer normalization, 2025. URL https://arxiv.org/abs/2501.03096.

Valérie Castin, Pierre Ablin, José Antonio Carrillo, and Gabriel Peyré. A unified perspective on the dynamics of deep transformers, 2025. URL https://arxiv.org/abs/2501.18322.

Tyler A. Chang and Benjamin K. Bergen. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16, 2022. doi: 10.1162/tacl_a_00444. URL https://doi.org/10.1162/tacl_a_00444.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Joshua Engels, Eric J. Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=d63a4AM4hb.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 55–65, 2019.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations (ICLR)*, 2019a.

Xiang Gao, Meera Sitharam, and Adrian E. Roitberg. Bounds on the Jensen gap, and implications for mean-concentrated distributions. *The Australian Journal of Mathematical Analysis and Applications*, 16(2), 2019b.

Gemma Team. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Borjan Geshkovski, Cyril Letrouit, Yury Polyanskiy, and Philippe Rigollet. A mathematical perspective on transformers. *Bulletin of the American Mathematical Society*, 62(3):427–479, 2025.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

Wes Gurnee and Max Tegmark. Language models represent space and time. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=jE8xbmvFin.

Martin Idel. A review of matrix scaling and Sinkhorn's normal form for matrices and positive maps. *arXiv preprint arXiv:1609.06349*, 2016.

Shawn Im, Changdae Oh, Zhen Fang, and Sharon Li. How do transformers learn to associate tokens: Gradient leading terms bring mechanistic interpretability. In *International Conference on Learning Representations (ICLR)*, 2026.

Yibo Jiang, Goutham Rajendran, Pradeep Kumar Ravikumar, Bryon Aragam, and Victor Veitch. On the origins of linear representations in large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 21879–21911. PMLR, 2024. URL https://proceedings.mlr.press/v235/jiang24d.html.

Dhruva Karkada, Daniel J. Korchinski, Andres Nava, Matthieu Wyart, and Yasaman Bahri. Symmetry in language statistics shapes the geometry of model representations, 2026. URL https://arxiv.org/abs/2602.15029.

Juno Kim and Taiji Suzuki. Transformers learn nonlinear features in context: Nonconvex mean-field dynamics on the attention landscape. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 24527–24561. PMLR, 2024. URL https://proceedings.mlr.press/v235/kim24af.html.

Olga Kovaleva, Saurabh Kulshreshtha, Anna Rogers, and Anna Rumshisky. BERT busters: Outlier dimensions that disrupt transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, volume 27, pp. 2177–2185, 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/hash/b78666971ceae55a8e87efb7cbfd9ad4-Abstract.html.

Jennifer Meng Lu, Ruochen Zhang, Isabelle Lee, David Alvarez-Melis, Ellie Pavlick, and Naomi Saphra. Adversarial concept search: Predicting compositional errors from feature geometry, 2026. URL https://arxiv.org/abs/2606.13934.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017.

Meta AI. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024. Accessed 2026-08-03.

Jiaqi Mu, Suma Bhat, and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations (ICLR)*, 2018.

Neel Nanda and Joseph Bloom. Transformerlens. https://github.com/TransformerLensOrg/TransformerLens, 2022.

Andres Nava and Matthieu Wyart. Hierarchical concept geometry in language models emerges from word co-occurrence, 2026. URL https://arxiv.org/abs/2605.23821.

Gary W. Oehlert. A note on the delta method. *The American Statistician*, 46(1):27–29, 1992.

OLMo Team. 2 OLMo 2 Furious, 2024.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39643–39666. PMLR, 2024. URL https://proceedings.mlr.press/v235/park24c.html.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=bVTM2QKYuA.

Guilherme Penedo, Hynek Kydlícek, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks*, 2024.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, Doha, Qatar, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162/.

Giovanni Puccetti, Anna Rogers, Aleksandr Drozd, and Felice Dell'Orletta. Outlier dimensions that disrupt transformers are driven by frequency. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Technical Report*, 2019.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations (ICLR)*, 2020.

Philippe Rigollet. The mean-field dynamics of transformers, 2025. URL https://arxiv.org/abs/2512.01868.

Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

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

Curt Tigges, Michael Hanna, Qinan Yu, and Stella Biderman. LLM circuit analyses are consistent across training and scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

William Timkey and Marten van Schijndel. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4527–4546, 2021.

Daniel Wurgaft, Can Rager, Matthew Kowal, Vasudev Shyam, Sheridan Feucht, Usha Bhalla, Tal Haklay, Eric Bigelow, Raphael Sarfati, Thomas McGrath, Owen Lewis, Jack Merullo, Noah Goodman, Thomas Fel, Atticus Geiger, and Ekdeep Singh Lubana. Manifold steering reveals the shared geometry of neural network representation and behavior, 2026. URL https://arxiv.org/abs/2605.05115.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

A Conditional Mean Fields with Repeated Types

Section 3.3 defined the s,t-conditional and context-conditional mean fields under the simplifying assumption that a type occurs at most once in a context. This appendix removes that assumption and provides the full derivations. derivation is unchanged: the same two mean fields are reached by the same two steps, but one additional assumption is required, governing how the attention a query places on a type grows with the number of copies of that type present. Setting every count to zero or one recovers Section 3.3 exactly.

Notation for repeated types. Recall from Section 3.2 that $n_t(X)$ is the number of positions of type t available to the query in X, and write n_t for the corresponding random variable when X is drawn from the corpus. We write $\Pr_s(n_t = n)$ for the probability that a random context with query s contains exactly n copies of t, and $\mathbb{E}_{stn}$ for the conditional expectation over such contexts. The expected number of type-t positions in a type-s query's context is

$$\overline{n}_{st} \;=\; \mathbb{E}_s[n_t] \;=\; \sum_{n\geq 1} n \Pr_s(n_t = n).$$

When a type can occur at most once, $n_t \in \{0,1\}$, so $\overline{n}_{st} = \Pr_s(t)$ and $\mathbb{E}_{st1} = \mathbb{E}_{st}$; every expression below then reduces to its counterpart in Section 3.3.

The s,t-conditional mean field. Start from the exact contribution to the centroid (3):

$$\overline{\mathbf{z}}_s = \sum_{t\in\mathcal{T}} \mathbb{E}_s[A_{st}\overline{\mathbf{v}}_t]$$

We now condition on the *number* of copies of t in the context rather than merely on the presence of t. Partitioning on the value of n_t,

$$\mathbb{E}_s[A_{st}\overline{\mathbf{v}}_t] = \sum_{n\geq 0} \mathbb{E}_{stn}[A_{st}\overline{\mathbf{v}}_t] \Pr_s(n_t = n) \;=\; \sum_{n\geq 1} \mathbb{E}_{stn}[A_{st}\overline{\mathbf{v}}_t] \Pr_s(n_t = n),$$

where the $n = 0$ term drops out because $A_{st} = 0$ when t is absent from the context, exactly as before.

Denote $\mathbb{E}_{stn}[A_{st}\overline{\mathbf{v}}_t]$ as $\overline{\mathbf{z}}_{stn}$. This is the average contribution to the s token centroid by token t when exactly n copies of t are in the context: the influence measure $\overline{\mathbf{z}}_{st}$ of Section 3.3, resolved by multiplicity. Then

$$\overline{\mathbf{z}}_s = \sum_{t\in\mathcal{T}} \sum_{n\geq 1} \overline{\mathbf{z}}_{stn} \Pr_s(n_t = n)$$

which is again an exact decomposition of the corpus contribution, now refined by multiplicity. We rewrite $\overline{\mathbf{z}}_{stn}$ as

$$\overline{\mathbf{z}}_{stn} = \mathbb{E}_{stn}[A_{st}]\,\mathbb{E}_{stn}[\overline{\mathbf{v}}_t] + \mathrm{Cov}_{stn}(A_{st}, \overline{\mathbf{v}}_t). \tag{9}$$

Now we introduce two mean-field assumptions: first, the (s,t)-conditional analog of Assumption 1, and second, that the average value mapped by a key of type t is independent of s:

Assumption 5. Attention-value independence, (s,t)-conditional . This assumption posits that $\mathrm{Cov}_{st}(A_{st}, \overline{\mathbf{v}}_t) \approx 0$, now on the finer conditioning set of contexts having a query of type s and type t among the keys.

Assumption 6. Value-query independence. This assumption posits that $\mathbb{E}_{st}[\overline{\mathbf{v}}_t] \approx \boldsymbol{\mu}_t$, i.e., conditioning on the query type s has negligible effect on the average value transmitted by type t keys. Extending to repeated types, the average value map of token t is further assumed to be independent of how many copies of t are present: $\mathbb{E}_{stn}[\overline{\mathbf{v}}_t] \approx \boldsymbol{\mu}_t$ for every s and every n.

The quantity $\mathbb{E}_{stn}[\overline{\mathbf{v}}_t]$ is an average over type-s query contexts, and $\overline{\mathbf{v}}_t$ is itself attention-weighted within the context, so it differs from $\boldsymbol{\mu}_t$ by the degree to which type t's representation is contextualized in the company of s; Assumption 6 treats that difference as negligible. Applying Assumptions 5 and 6 to (9) gives

$$\overline{\mathbf{z}}_{stn} \approx \mathbb{E}_{stn}[A_{st}]\,\boldsymbol{\mu}_t. \tag{10}$$

That leaves $\mathbb{E}_{stn}[A_{st}]$, where repeated types enter: it is a separate quantity for each multiplicity n, whereas Section 3.3 had the single number $W_{st} = \mathbb{E}_{st}[A_{st}]$. Here, we recast W_{st} as a rate *per key* rather than per context,

$$W_{st} \;:=\; \frac{\mathbb{E}_s[A_{st}]}{\mathbb{E}_s[n_t]} \;=\; \frac{P_{st}}{\overline{n}_{st}},$$

the total attention a type-s query places on type t, divided by the expected number of type-t keys that mass was spread over. This is a definition rather than an assumption, and it is directly measurable: numerator and denominator are each a single corpus pass. When a type occurs at most once it reduces to $W_{st} = P_{st}/\Pr_s(t) = \mathbb{E}_{st}[A_{st}]$, the quantity of Section 3.3. Our final assumption relates how attention mass grows with multiplicity:

Assumption 7. Homogeneity of attention. $\mathbb{E}_{stn}[A_{st}] = n\,W_{st}$. The attention a type-$s$ query places on type t is proportional to the number of copies of t present, at a per-key rate that does not otherwise depend on the context.

Assumption 7 is purely a linearity claim: averaging it back over n returns $\sum_n n\,W_{st}\,\Pr_s(n_t = n) = W_{st}\,\overline{n}_{st} = P_{st}$, which holds identically by the definition of W_{st}. What is being assumed is therefore not the overall level of the attention but its *distribution across multiplicities*: that a context holding five copies of t draws five times the mass of one holding a single copy, rather than, say, saturating. Applying Assumption 7 and summing over n,

$$\overline{\mathbf{z}}_s \approx \hat{\mathbf{z}}_s = \sum_{t\in\mathcal{T}} W_{st}\,\boldsymbol{\mu}_t \sum_{n\geq 1} n\,\Pr_s(n_t = n) \;=\; \sum_{t\in\mathcal{T}} W_{st}\,\boldsymbol{\mu}_t\,\overline{n}_{st},$$

the repeated-type counterpart of the s,t-conditional mean field of Section 3.3. Here, the presence probability $\Pr_s(t)$ is replaced by the expected count $\overline{n}_{st}$, and nothing else changes.

The context-conditional mean field. We now repeat the second construction of Section 3.3 for a particular context X with query s, again allowing a type to occur any number of times in X. As before, W_{stX} denotes the attention paid by s to t in context X, and $\boldsymbol{\mu}_{tX}$ the average value map of token t in that context.

Note that W_{stX} is the *total* attention the query places on type t, summed over all $n_t(X)$ of its positions in the prefix; it is A_{st} of (3) evaluated on this particular context. Likewise $\boldsymbol{\mu}_{tX}$ is the attention-weighted mean of the value vectors at those same $n_t(X)$ positions, which is $\overline{\mathbf{v}}_t$ evaluated on this context. Both therefore already account for every copy of t in X.

Conditioning on X fixes every count, so $\Pr_X(t)$ is again 1 for all token types present in X (a set denoted $\mathrm{supp}(X)$), and 0 otherwise, and

$$\mathbf{z}_{sX} = \sum_{t\in\mathcal{T}} W_{stX}\,\boldsymbol{\mu}_{tX}\,\Pr_X(t) \;=\; \sum_{t\in\mathrm{supp}(X)} W_{stX}\,\boldsymbol{\mu}_{tX}.$$

Next, as before, we make the **assumption** that $\boldsymbol{\mu}_{tX} = \boldsymbol{\mu}_t$, again noting that the difference between them is the *contextualization* of t induced by its context X, leaving

$$\hat{\mathbf{z}}_{sX} = \sum_{t\in\mathrm{supp}(X)} W_{stX}\,\boldsymbol{\mu}_t.$$

And as before, since X is a fixed context, W_{stX} is simply the attention A_{stX} that the head pays from s to t in this context. To construct a mean-field analog we form an **approximation** to A_{stX} using renormalization, now weighting each type by the number of copies it contributes:

$$W_{stX} \approx \hat{A}_{stX} = \frac{n_t(X)\,W_{st}}{\sum_{t'\in\mathrm{supp}(X)} n_{t'}(X)\,W_{st'}}, \tag{11}$$

the numerator being the mass that Assumption 7 predicts for a context holding $n_t(X)$ copies of t. This yields the *context-conditional* mean field approximation for contexts with repeated types,

$$\hat{\mathbf{z}}_{sX} = \sum_{t\in\mathrm{supp}(X)} \hat{A}_{stX}\,\boldsymbol{\mu}_t,$$

which reduces to the expression of Section 3.3 when every $n_t(X)$ equals 1. Note that the multiplicities do not cancel from the renormalization: they reweight the types relative to one another, so the prediction depends on X through the full multiset of types it contains and not merely through $\mathrm{supp}(X)$.

B Analysis of the Context-conditional Mean Field

B.1 The Kernel $\hat{A}$ as a Mean Field

The context-conditional mean field underlies §5. Its kernel renormalizes the corpus kernel over the types a context actually contains (11):

$$\hat{A}_{stX} \;=\; \frac{n_t(X)\, W_{st}}{N_X}, \qquad N_X \;=\; \sum_{t' \in \mathrm{supp}(X)} n_{t'}(X)\, W_{st'}, \tag{12}$$

where A_{stX} is the *total* attention the head pays from a query of type s to all $n_t(X)$ positions of type t, and W_{st} defined as $P_{st}/\bar{n}_{st}$ (Appendix A) with $P_{st} = \mathbb{E}_s[A_{st}]$ and $\bar{n}_{st} = \mathbb{E}_s[n_t(X)]$.

Renormalization is a natural way to turn a corpus-average propensity into something that sums to one over a particular context, but it introduces a nonlinearity into the definition of $\hat{A}$. Nonetheless, the choice of $\hat{A}$ can be justified if $\hat{A}$ is an *unbiased estimator*, that is, if

$$\mathbb{E}_{st}\Big[A_{stX} - \hat{A}_{stX}\Big] = 0. \tag{13}$$

Here $\hat{A}_{stX}$ varies with the context through its token counts, while $\mathbb{E}_{st}[A_{stX}]$ is fixed, so (13) asks the random quantity to match the constant in expectation.

The remainder of this section shows the following:

1. Equation (13) does *not* hold in general; we show why and we show how to estimate its error.
2. Equation (13) *does* hold approximately in practice for the data we study in this paper.
3. It is possible to define an *unbiased* replacement for W. Specifically, we show how to construct a matrix U, such that if we use U in place of W in the definition of $\hat{A}$, then (13) holds by construction. When we replace W with its unbiased version U in our experiments, *our key results do not change.*

In the remainder of this appendix we typically omit the context X and abbreviate the per-context quantities A_{stX}, $\hat{A}_{stX}$ and $n_t(X)$ as A_{st}, $\hat{A}_{st}$ and n_t. We restore the explicit X (as in A_{stX}, $n_t(X)$, N_X) at the steps where the dependence on the particular context is the point. The expectations $\mathbb{E}_s$ and $\mathbb{E}_{st}$ average over the contexts X, and the corpus quantities P_{st}, W_{st}, $\bar{n}_{st}$ and π_{st} carry no context argument.

B.2 Analysis of Bias in $\hat{A}$

First, we give some insight into the nature of the bias in $\hat{A}$. Because $A_{stX} = 0$ and $n_t(X) = 0$ together whenever t is absent, writing $\pi_{st} = \Pr_s(t \in \mathrm{supp}(X))$ gives $\mathbb{E}_s[A_{st}] = \pi_{st}\mathbb{E}_{st}[A_{st}]$ and $\mathbb{E}_s[n_t] = \pi_{st}\mathbb{E}_{st}[n_t]$. The presence probability therefore cancels from the definition of W:

$$W_{st} \;=\; \frac{P_{st}}{\bar{n}_{st}} \;=\; \frac{\mathbb{E}_s[A_{st}]}{\mathbb{E}_s[n_t]} \;=\; \frac{\mathbb{E}_{st}[A_{st}]}{\mathbb{E}_{st}[n_t]} \tag{14}$$

which means that

$$\mathbb{E}_{st}[A_{st}] \;=\; W_{st}\, \mathbb{E}_{st}[n_t]. \tag{15}$$

From (12), $\mathbb{E}_{st}[\hat{A}_{st}] = W_{st}\, \mathbb{E}_{st}[n_t/N_X]$. Subtracting from (15),

$$\mathbb{E}_{st}\Big[A_{st} - \hat{A}_{st}\Big] \;=\; W_{st}\Big(\mathbb{E}_{st}[n_t] \;-\; \mathbb{E}_{st}[n_t/N_X]\Big). \tag{16}$$

Writing $1/N_X = 1 + (1 - N_X)/N_X$ in (16),

$$\mathbb{E}_{st}[n_t/N_X] - \mathbb{E}_{st}[n_t] \;=\; -\,\mathbb{E}_{st}\left[\frac{n_t\,(N_X - 1)}{N_X}\right], \tag{17}$$

More precisely, the deficit splits into two terms of opposite sign,

$$\mathbb{E}_{st}[n_t/N_X] - \mathbb{E}_{st}[n_t] \;=\; \mathbb{E}_{st}[n_t]\big(\mathbb{E}_{st}[1/N_X] - 1\big) \;+\; \mathrm{Cov}_{st}\big(n_t,\, 1/N_X\big). \tag{18}$$

The first is a Jensen term: $1/N_X$ is convex, so $\mathbb{E}_s[1/N_X] > 1/\mathbb{E}_s[N_X] \approx 1$, pushing toward *over*-prediction. For example, in GPT-2 measured $\mathbb{E}_s[1/N_X]$ is 1.36 while $\mathbb{E}_s[N_X]$ is 0.99. The second

is negative: both $n_t(X)$ and N_X grow with the length of the context, so n_t and $1/N_X$ are anticorrelated, pushing toward *under*-prediction. For every bucket whose multiplicity grows with length (all tracked types, and the "other" bucket), the covariance term is larger, and $\hat{A}_{st}$ under-predicts. The one exception is the position-0 bucket, whose multiplicity is identically 1: its covariance term vanishes exactly, leaving the pure Jensen term, and it is over-predicted.

Both A and $\hat{A}$ are distributions over the same context: $\sum_t A_{stX} = \sum_t \hat{A}_{stX} = 1$, hence

$$\sum_t \mathbb{E}_s\left[A_{st} - \hat{A}_{st}\right] = 0 \tag{19}$$

exactly, for every s. It's useful to define the per-cell *relative* bias

$$r_{st} = 1 - \mathbb{E}_{st}[\hat{A}_{st}]/\mathbb{E}_{st}[A_{st}]. \tag{20}$$

This establishes a connection with P. The identity (19) then reads $\sum_t P_{st}\, r_{st} = 0$, since $\mathbb{E}_s[A_{st} - \hat{A}_{st}] = \pi_{st}\mathbb{E}_{st}[A_{st}]\, r_{st} = P_{st}\, r_{st}$. In other words, the relative biases cancel when each is weighted by the average attention P_{st} its pair carries.

The bias is thus a redistribution of attention mass among key types, not a net distortion: because it sums to zero under the same P-weighting that governs the mean-field prediction, it cannot accumulate in the aggregate quantities the paper reports. §B.3 shows the per-pair bias is also small empirically, and §B.4 removes it entirely with a calibrated estimator.

B.3 The Measured Bias of $\hat{A}$ is Small

To assess the empirical bias of $\hat{A}_{st}$, we proceed as follows. For each corpus position whose query type is tracked, and each head, we compute N_X over the inclusive causal prefix and accumulate per (head, type) the conditional means $\mathbb{E}_{st}[n_t]$ and $\mathbb{E}_{st}[n_t/N_X]$.[3] The quantity $\mathbb{E}_{st}[n_t/N_X]/\mathbb{E}_{st}[n_t] = \mathbb{E}_{st}[\hat{A}_{st}]/\mathbb{E}_{st}[A_{st}]$ by (16); a ratio 1 is unbiased. The results are in Table 1. The table shows that the unweighted cell average is 0.946 and 0.933, a 5–7% under-prediction for the typical pair, with over 95% of pairs within 20% of unbiased in both models. $\mathbb{E}_s[N_X] \approx 1$ is a consistency check on P and $\bar{n}$.

model	$\mathbb{E}_s[N_X]$	$\mathbb{E}_{st}[n_t]$	$\mathbb{E}_{st}[n_t/N_X]/\mathbb{E}_{st}[n_t]$	cells within 20% of 1.0
GPT-2	0.988	1.468	**0.946**	95.5%
Gemma-2-2B	0.995	1.466	**0.933**	96.8%

Table 1: Observed bias of (head,type) pairs. "Cells within 20%" counts well-sampled (head, type) pairs whose ratio lies in $[0.8, 1.2]$: 129k cells for GPT-2, 187k for Gemma.

Empirically, in GPT-2 we observe that correlation between n_t and $(N_X - 1)/N_X$ is quite similar for different token types t. Binning types into quartiles by presence frequency (the number of scored query positions whose causal prefix contained the type), with the top quartile split at the 90th percentile, every bin's mean ratio lies in $[0.93, 0.95]$, with ≈11% of cells above 1 in each. The compensation that (19) requires therefore sits in the untracked bucket: the position-0 bucket has $n = 1$ by definition (hence no covariance term by (18)), ratio $\mathbb{E}[1/N_X] = 1.36$, and ≈40% of the attention mass. Thus, $\hat{A}$ has one over-predicted sink balancing a uniformly under-predicted field. The same structure holds in Gemma-2-2B: tracked types uniform at 0.93–0.94 across every frequency quartile, and the position-0 sink over-predicted at 1.22 on 40% of the mass. A separate run comparing $\hat{A}$ against the model's measured attention reproduces the deficit, confirming that (16) is the whole story.

This small bias has negligible effect on our results. We demonstrate this as follows. In the next section we show how to construct a truly unbiased estimator of $\hat{A}$. We then show that, using this unbiased estimator, the key results in the paper are unchanged.

[3] Corpus: 389 chunks of 512 OpenWebText tokens (∼200k).

Algorithm 2: Unbiased calibration of W: the estimator U (run per head; heads are independent)

$U^{(0)} \leftarrow W = P/\bar{n}$; // the paper's estimator: a good warm start
for $k = 0, 1, 2, \ldots$ **do**
  $Q_{st} \leftarrow 0$ for all (s,t); $m_s \leftarrow 0$ for all s ; // corpus accumulators
  foreach *chunk, each query position p with tracked type s* **do**
    $n \leftarrow \big(n_t(X)\big)_t$; // multiplicities of the causal prefix
    $a_t \leftarrow n_t U^{(k)}_{st}$ for all t ; // numerators of (12)
    $N \leftarrow \sum_t a_t$; // the normalizer N_X
    $Q_{st} \leftarrow Q_{st} + a_t/N$ for all t ; // accumulate $\hat{A}_{stX}(U^{(k)})$ into row s only
    $m_s \leftarrow m_s + 1$;
  foreach *query type s* **do** // normalize each row by its own occurrence count
    $Q_{st} \leftarrow Q_{st}/m_s$ for all t ; // now $Q_{st} = \mathbb{E}_s[\hat{A}_{st}(U^{(k)})]$
  if $\max_{s,t} |Q_{st} - P_{st}| < \varepsilon$ **then stop**;
  foreach *pair (s,t)* **do** // one multiplicative correction per entry
    $U^{(k+1)}_{st} \leftarrow U^{(k)}_{st} \cdot \Big(\dfrac{P_{st}}{\max(Q_{st},\, \delta)}\Big)^{\alpha}$;

B.4 Using an Unbiased Estimator of $\hat{A}$ Does Not Change Key Results

The bias in (16) exists because W is obtained in closed form, as the ratio $P/\bar{n}$ of two *marginal* means, while it is *used* inside a nonlinear per-context renormalization.

Constructing an unbiased estimator of $\hat{A}$. Instead of using W in the definition of $\hat{A}$, we may instead find a matrix U that makes (13) true by construction. Multiplying (13) by π_{st} to remove the conditioning, the requirement is

$$\mathbb{E}_s\Big[\hat{A}_{st}(U)\Big] \;=\; \mathbb{E}_s[A_{st}] \;=\; P_{st} \qquad \text{for every } (s,t), \tag{21}$$

i.e. *the kernel must reproduce the measured attention marginals when run through its own renormalization.* Any U satisfying (21) gives $\mathbb{E}_{st}[A_{st} - \hat{A}_{st}] = 0$ exactly, for every pair, by construction.

We note that, for each query type s, achieving (21) is a marginal-matching problem. To see this, observe that for each context X containing s, (12) makes $\hat{A}_{s\cdot X}(U)$ a probability vector over the key types: type t is offered $n_t(X)$ times, each occurrence carries weight U_{st}, and the vector is the resulting share. Averaging those per-context vectors over the corpus gives $Q_{s\cdot} = \mathbb{E}_s[\hat{A}_{s\cdot}(U)]$, again a probability vector over key types — the model's *predicted type marginal* for query type s. The measured $P_{s\cdot}$ is the corresponding *observed type marginal*. So (21) asks for the weights U whose induced marginal, after per-context normalization and averaging, reproduces the observed one.

Problems of this marginal-matching form are typically solved by Sinkhorn-type iterations (Sinkhorn & Knopp, 1967; Idel, 2016), i.e., iterative proportional fitting (IPF). IPF is appropriate because per-context normalization $\sum_t \hat{A}_{stX} = 1$ is one constraint family, satisfied automatically by construction; the target $Q_{s\cdot} = P_{s\cdot}$ is the other. We have exactly one free parameter W_{st} per constraint, so the system is square. The update is correspondingly multiplicative: if the current kernel puts too little mass on (s,t), scale U_{st} up in proportion to the shortfall and let the renormalization redistribute the rest. Applying IPF to this problem is shown as Algorithm 2.

Algorithm. Applying IPF to this problem is shown as Algorithm 2. Applying a damping exponent $\alpha < 1$ in Algorithm 2 is the standard remedy if the residual were to oscillate rather than decrease. Empirically it is not needed here — with $\alpha = 1$ the residual converges in three iterations with no oscillation, which is the behavior expected of a contraction.

Using an unbiased estimator of $\hat{A}$ does not change key results. At a fixed point of Algorithm 2, $Q = P$ so (21) holds and the bias (16) is zero for every (s,t). Unbiasedness of $\hat{A}(U)$ is then true

by construction. Consequently, we can recompute any body result that uses the context-conditional kernel with a calibrated U in place of $W = P/\bar{n}$, holding everything else fixed.

We first build a calibrated U at 2M tokens. GPT-2 was calibrated at its published weights and Pythia-160m at training step 1000, each against its own P and $\bar{n}$. Table 2 shows the convergence properties of Algorithm 2, noting that good convergence was achieved in 5 iterations for both cases. We note that the ratio $U/W^{(0)}$ over the 67.6M fitted GPT-2 cells has a 5–95% range of $[0.76,\ 1.29]$ and a median of 1.005.

model	residual	iter 0	1	2	3	4
GPT-2	median $\lvert Q-P\rvert/P$	0.1178	0.0103	0.0014	0.0002	**0.00003**
	max $\lvert Q-P\rvert$	0.346	0.127	0.090	0.071	**0.057**
Pythia-160m (step 1000)	median $\lvert Q-P\rvert/P$	0.1261	0.0145	0.0028	0.0006	**0.0001**
	max $\lvert Q-P\rvert$	0.217	0.107	0.081	0.064	**0.052**

Table 2: Calibration at 2M tokens (3914 chunks of 512), five iterations, $\alpha = 1$. Iteration 0 is the residual of the present estimator $W = P/\bar{n}$.

Recomputing the training-dynamics result of §5 confirms it is unaffected. We chose the data point for Pythia-160m at step 1000 as this lies just at the point where mean-field model performance begins to deviate from true model performance, and applied the calibrated U for that model. At that point, L_{MF} is unchanged to three significant figures (invisible on the plot), and ΔL and D_{MF} are also unchanged to three significant figures.

The paper's other results (§4) use the kernel P, so are not affected by bias in $\hat{A}$. We thus conclude that not only is the bias of $\hat{A}$ small, as shown here, but that small bias does not affect any of the key results in the paper.

C Impact of the Value-Map Mean-Field Approximation

This appendix asks what is lost by evaluating the value map at a type's centroid rather than averaging it over the type's occurrences. We find that little is lost: occurrences of a type sit close enough to their centroid that the two agree.

The argument has two parts. The error is exactly the Jensen gap of the attention block's input normalizer, so it is governed by the within-type covariance; and that covariance is small enough, at every layer of both models, that $\bar{\mathbf{v}}_t \approx v(\mathbf{c}_t)$.

The corpus mean field (Section 3.2) replaces the per-occurrence value $v(\mathbf{x}_t)$ by the value of the type centroid $v(\mathbf{c}_t)$, i.e. it assumes $\bar{\mathbf{v}}_t = \mathbb{E}[v(\mathbf{x}_t)] \approx v(\mathbf{c}_t)$. Because the per-head value map $v(\mathbf{x}) = \ln_1(\mathbf{x})\, W_V + \mathbf{b}_V$ has the input normalizer $\ln_1$ (LayerNorm in GPT-2, RMSNorm in Gemma-2) as its only nonlinearity (and W_V is linear, so it commutes with the expectation), the error is exactly the Jensen gap of $\ln_1$ over the type-t occurrence cloud,

$$\bar{\mathbf{v}}_t - v(\mathbf{c}_t) = \big(\mathbb{E}[\ln_1(\mathbf{x}_t)] - \ln_1(\mathbf{c}_t)\big)\, W_V.$$

A second-order (delta-method) expansion (Oehlert, 1992; Gao et al., 2019b) gives $\mathbb{E}[\ln_1(\mathbf{x}_t)] - \ln_1(\mathbf{c}_t) \approx \frac{1}{2}\operatorname{tr}\big(\nabla^2 \ln_1(\mathbf{c}_t)\operatorname{Cov}(\mathbf{x}_t)\big)$, where the LayerNorm Jacobian and Hessian follow from $\ln_1(\mathbf{x}) = \sqrt{d}\, P\mathbf{x}/\|P\mathbf{x}\|$ with the mean-centering projection $P = I - \frac{1}{d}\mathbf{1}\mathbf{1}^\top$ (Ba et al., 2016; Brody et al., 2023) (RMSNorm omits P, $\ln_1(\mathbf{x}) = \sqrt{d}\,\mathbf{x}/\|\mathbf{x}\|$, but the argument is unchanged). The gap is therefore governed by the *within-type covariance* $\operatorname{Cov}(\mathbf{x}_t)$: if occurrences of a type cluster tightly around their centroid, the value-map mean field is exact to second order. We verify this clustering directly.

Within-type clustering is the Euclidean counterpart of the contextual self-similarity of Ethayarajh (2019), who studied mean pairwise cosine across a word's occurrences, which they found decreases with depth.

Metrics. Write $\{\mathbf{x}_{t,i}\}_{i=1}^{n_t}$ for the residual-stream representations of the n_t occurrences of type t at a given depth, with type centroid $\mathbf{c}_t = \frac{1}{n_t}\sum_i \mathbf{x}_{t,i}$ and global centroid $\bar{\mathbf{c}} = \frac{1}{\sum_t n_t}\sum_{t,i}\mathbf{x}_{t,i}$. We

report three quantities per layer. *(i) Relative RMS deviation:* $r_t = \sqrt{\frac{1}{n_t}\sum_i \|\mathbf{x}_{t,i} - \mathbf{c}_t\|^2} \,/\, \|\mathbf{c}_t\|$, the root-mean-square spread of a type's occurrences about its centroid, relative to the centroid magnitude. *(ii) Variance explained by type, η^2:* a one-way variance decomposition (the intraclass correlation). With total variance $V_{\text{tot}} = \frac{1}{\sum n_t}\sum_{t,i} \|\mathbf{x}_{t,i} - \bar{\mathbf{c}}\|^2$ and pooled within-type variance $V_{\text{w}} = \frac{1}{\sum n_t}\sum_{t,i} \|\mathbf{x}_{t,i} - \mathbf{c}_t\|^2$,

$$\eta^2 = 1 - V_{\text{w}}/V_{\text{tot}},$$

the fraction of variance attributable to type identity ($\eta^2 \to 1$ means occurrences collapse onto their centroids; $\eta^2 \to 0$ means type identity carries no information). *(iii) Value-map Jensen gap:* the relative magnitude $\|\mathbb{E}[\ln_1 \mathbf{x}_t] - \ln_1 \mathbf{c}_t\| / \|\ln_1 \mathbf{c}_t\|$ and the direction $1 - \cos(\mathbb{E}[\ln_1 \mathbf{x}_t],\, \ln_1 \mathbf{c}_t)$ — the actual size and orientation of the gap that the second-order argument above bounds. All quantities are accumulated in a single pass over 2M OpenWebText tokens, per type and per residual depth $E[\ell]$ (the input to each block's attention).

Rogue-dimension control. A handful of coordinates — "rogue dimensions" — carry a disproportionate share of the residual variance. Timkey & van Schijndel (2021) showed that for GPT-2's final layer a single dimension accounts for $\sim$76% of the expected cosine between two random tokens. A variance-weighted statistic, including the η^2 above (which sums variance across coordinates), is therefore dominated by whether those few dimensions separate types, and is nearly blind to the structure in the remaining $\sim$760. We accordingly also report a per-dimension-standardized version: writing σ_d^2 and w_d for the total and pooled-within variance of coordinate d, we average the *per-coordinate* ratio,

$$\eta_z^2 = \tfrac{1}{d}\textstyle\sum_d \left(1 - w_d/\sigma_d^2\right),$$

which weights every dimension equally — equivalent to z-scoring each coordinate to unit variance before the decomposition — so that no single rogue dimension dominates. (The raw $\eta^2 = 1 - \sum_d w_d / \sum_d \sigma_d^2$ is its variance-weighted counterpart.) This is the standardization control recommended by Timkey & van Schijndel (2021).

Results. Figure 7 reports the results for both models. In GPT-2 the rogue-dimension-robust measure shows type identity explaining $\eta_z^2 = 0.90$ of the variance at the model input: occurrences are very tightly clustered around their type centroid relative to the between-type spread. The raw η^2 *understates* this at 0.40 — a few rogue dimensions obscure the within-type structure, as documented by Timkey & van Schijndel (2021). Both measures decline with depth (to $\eta_z^2 \approx 0.25$ by layer 11), the geometric form of the increasing context-specificity of Ethayarajh (2019).

Gemma-2 behaves similarly: $\eta_z^2 = 0.85$ at the first block input falling to $\approx$0.15 in the deep layers. There are two aspects specific to Gemma-2. First, its input residual $E[0]$ is *exactly* degenerate ($\eta^2 = 1$, zero within-type spread): RoPE injects position inside the attention computation rather than adding it to the residual, so a type's input representation is identical across occurrences, whereas GPT-2's additive positional embedding already spreads a type's occurrences at the input. Second, for Gemma the z-scored η_z^2 sits slightly *below* the raw η^2 rather than above it (e.g. 0.32 vs 0.52 at $E[4]$): its high-variance coordinates are themselves type-discriminative, so equalizing dimensions removes information rather than noise — the opposite of GPT-2 — though the clustering stays strong under both measures.

The impact of this tight clustering for the mean field is the same in both models: the value-map Jensen gap is negligible at every layer (right-hand side of the figure). In terms of cosine distance, in GPT-2 (using LayerNorm), $1 - \cos(\mathbb{E}[\ln_1 \mathbf{x}], \ln_1 \mathbf{c}_t) < 3 \times 10^{-5}$, and in Gemma-2 (using RMSNorm), $< 3 \times 10^{-4}$. (These are invisible on the plots.) Thus, in both models, $\bar{\mathbf{v}}_t$ and $v(\mathbf{c}_t)$ are essentially parallel, and have a residual magnitude gap of order 10–28%. The clustering is thus tight enough that replacing $\bar{\mathbf{v}}_t$ by $v(\mathbf{c}_t)$ is an excellent approximation, particularly in direction — the empirical condition behind the corpus mean field.

D Mean-Field Kernel Details

The mean-field rollout tracks the N most frequently occurring types, $\mathcal{T}_N \subset \mathcal{T}$. So that each query's attention remains normalized, the kernel for each head carries two key columns beyond the N tracked types: a **BOS/position-0** column and an **other** column that aggregates every type not in

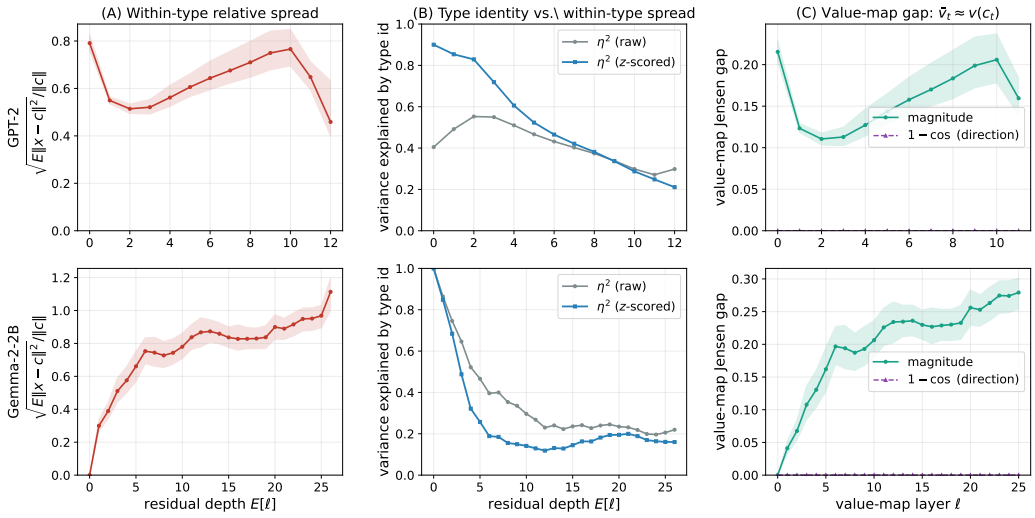

Figure 7: Same-type token representations and their type centroids (2M OpenWebText tokens, top-1000 types). **Top row:** GPT-2; **bottom row:** Gemma-2-2B. **(A)** relative RMS deviation of occurrences around their type centroid, $\sqrt{\mathbb{E}\|\mathbf{x}-\mathbf{c}_t\|^2}/\|\mathbf{c}_t\|$ (median; IQR band over types). **(B)** fraction of variance explained by type identity, η^2: raw (gray) and per-dimension z-scored (blue). **(C)** the value-map Jensen gap $\|\mathbb{E}[\ln_1 \mathbf{x}] - \ln_1 \mathbf{c}_t\|/\|\ln_1 \mathbf{c}_t\|$ (green) and its direction $1-\cos$ (purple).

$\mathcal{T}_N$. Thus $P^{\ell h}$ is an $N \times (N{+}2)$ row-stochastic matrix: for a query of type s,

$$\sum_{t\in\mathcal{T}_N} P^{\ell h}_{st} \;+\; P^{\ell h}_{s\bullet} \;+\; P^{\ell h}_{s\circ} \;=\; 1,$$

where $P^{\ell h}_{s\bullet}$ and $P^{\ell h}_{s\circ}$ are the corpus-averaged attention masses a type-s query places on position 0 and on positions holding non-tracked types, respectively, defined exactly as the tracked masses $P^{\ell h}_{st} = \mathbb{E}_{\mathrm{occ}}[A_{st}]$. The per-head contribution of (4) (with $v(\cdot)$ the head's value map) is then

$$\mathbf{z}^{\ell h}_s = \sum_{t\in\mathcal{T}_N} P^{\ell h}_{st}\, v(\hat{\mathbf{c}}^{\ell}_t) \;+\; P^{\ell h}_{s\bullet}\, v(\mathbf{b}^{\ell}) \;+\; P^{\ell h}_{s\circ}\, v(\bar{\mathbf{c}}^{\ell}).$$

BOS/position-0 column. Position 0 of every input holds the BOS token, and under the causal mask it attends only to itself, so its representation has no occurrence-to-occurrence variation. $\mathbf{b}^{\ell}$ is that layer-ℓ representation: a deterministic function of the model weights, obtained from a single forward pass on the one-token input [BOS] (equivalently, the corpus mean of position-0 representations). It is therefore not a measured contextual quantity. Position 0 is also where the models place a large share of their attention, the attention sink of Appendix E, which is why this column is kept separate rather than folded into "other". Every input is prepended with the model's BOS token — or, for models with no dedicated BOS such as GPT-2, its surrogate `<|endoftext|>` — following the TransformerLens convention. In models with a true BOS the token can be frequent enough to enter $\mathcal{T}_N$ itself (Gemma-2's `<bos>` does), and it must then be excluded as a *query* type from the P and W matrices: it occurs only at position 0, where the causal prefix contains nothing else to attend to, so its kernel row is empty and its renormalized attention undefined.

Other column. The non-tracked types are collapsed into a single column whose value we set to $\bar{\mathbf{c}}^{\ell} = \frac{1}{N}\sum_{t\in\mathcal{T}_N} \hat{\mathbf{c}}^{\ell}_t$, the running mean of the predicted *tracked* centroids at layer ℓ. This is internal to the rollout and requires no measurement. Consequently the rollout consults no measured residual at any layer: the tracked centroids $\hat{\mathbf{c}}^{\ell}_t$ evolve from the input embeddings, the BOS value $\mathbf{b}^{\ell}$ is a weight-derived constant, and the "other" value $\bar{\mathbf{c}}^{\ell}$ is the running centroid mean.

Held-out evaluation. All comparisons in Section 4.2 are out-of-sample. We split OpenWebText into two disjoint halves by document parity (even-indexed documents versus odd), 5M tokens each, and estimate every corpus quantity the model uses — the mean-field attention kernel $P^{\ell h}$, the co-occurrence statistics, and the per-type value and centroid statistics — on the *train* half, while mea-

suring the true per-layer centroids $\mathbf{c}_t^\ell$ that the rollout is scored against on the disjoint *test* half. The rollout therefore never sees the evaluation centroids during estimation.

Estimation of W. Because $W_{st} = P_{st}/\bar{n}_{st}$ is a ratio of estimated quantities, its entries are noise-dominated — and can be arbitrarily large — wherever the pair (s, t) is rarely observed. We therefore restrict W to its well-estimated support: entries whose co-occurrence count C_{st} (the number of s-occurrences with at least one t in the causal prefix) falls below 20 are set to zero, and $\hat{A}_{stX}$ renormalises over the remaining support. The mask discards 53% of (s, t) pairs but only 3% of attention mass (Gemma-2: 41% and 5%), and all context-conditional results are computed under it.

E Coverage of the Tracked Type Set

Every construction in the body of this paper tracks the $N = 1000$ most frequent token types, with everything else collapsed into a position-0 bucket and an "other" bucket (Appendix D). How much of the corpus that set covers depends on what is counted: about two thirds of corpus positions carry a tracked type, while the share of attention mass landing on tracked types is far lower, 18–42%, almost all of the difference attributable to the attention sink at position 0. The choice of N itself matters less than either figure suggests: Appendix L repeats the training-waypoint substitution at $N = 2000$ and $N = 4000$ and finds the phase structure unchanged.

Position coverage. Since each position defines one context (Section 3.1), corpus positions whose type is tracked equate to contexts for which the mean field has a tracked query type. The fraction of positions covered is the relevant measure wherever the substitution is applied only at tracked query types. The remaining positions keep the model's own computation. Measured over 1M OpenWebText tokens with each model's own tokenizer and its own top-1000 set (Table 3):

Table 3: Position coverage: the fraction of corpus positions (1M OpenWebText tokens) whose type is among the model's top-1000 set.

model	vocabulary	position coverage
GPT-2 small	50,257	0.663
Pythia	50,277	0.655
Gemma-2-2B	256,000	0.684
Llama-3.2-3B / 3.1-8B	128,256	0.643
Qwen-3-14B	151,669	0.660

Coverage of positions sits at 64–68% across a fivefold spread in vocabulary size. 1000 types out of Gemma-2's 256,000 cover as much text as 1000 out of GPT-2's 50,257. Token frequency is Zipfian, so the head of the distribution is the same text however finely the tail is divided. The two Llamas share a tokenizer, and so share a set and a coverage.

Figure 8 shows why. The rank–frequency curves of the five tokenizers, measured on the same 2M tokens of OpenWebText, lie almost on top of one another over the tracked range, and the rank-1000 cutoff falls at nearly the same frequency in every model (0.96–1.10×10^{-4}). They separate only beyond rank $\sim 10^4$, where the larger vocabularies keep resolving types the smaller ones have merged, which is the region the tracked set excludes. Taking the top 1000 types therefore selects the same text in each model, not just the same number of types.

Attention-mass coverage. The second quantity is how much of the attention a head places lands on tracked types, as opposed to the position-0 bucket or the "other" bucket. This is the row sum $\sum_{t \in \mathcal{T}_N} P_{st}^{\ell h}$, the share of query type s's attention that reaches tracked key types, averaged over layers, heads and tracked query types (Table 4).

†The Llama-3.1-8B row is measured on the reduced-vocabulary extraction of Appendix J, which excludes 67 structural types that are both frequent and heavily attended, so its tracked share is understated. Measured on that same reduced basis the Llama-3.2-3B figure falls from 0.231 to 0.182, against the 8B's 0.181, so on a like-for-like footing the two models agree, and the 8B's full-vocabulary value should be read as ≈0.23. The final column inherits the same understatement, so the 8B's 0.523 is not comparable to the full-vocabulary rows above it.

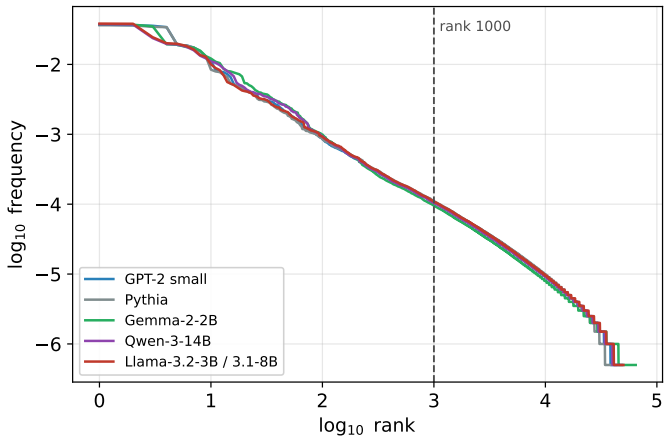

Figure 8: Rank–frequency curves for the five tokenizers on 2M tokens of OpenWebText, log–log, one curve per tokenizer; the two Llamas share a tokenizer and so a curve. Dashed line: the rank-1000 cutoff used throughout. The curves are nearly indistinguishable up to the cutoff and for a decade beyond it, diverging only in the far tail where vocabulary size decides how finely rare text is split.

Table 4: Attention-mass coverage: average per-query attention landing on tracked types, the position-0 sink, and the "other" bucket. The final column is the tracked share of non-sink attention, computed from the row entries; it is the quantity comparable to the position coverage of Table 3.

model	tracked types	position 0	other	tracked / (tracked + other)
GPT-2 small	0.419	0.399	0.182	0.697
Gemma-2-2B	0.417	0.405	0.178	0.701
Qwen-3-14B	0.370	0.488	0.142	0.723
Llama-3.2-3B	0.231	0.654	0.115	0.668
Llama-3.1-8B†	0.181	0.654	0.165	0.523

The largest single destination in every model is the sink: position 0 takes 40% of all attention in GPT-2 and Gemma-2. It takes 49% in Qwen-3 and 65% in both Llamas, consistent with the attention-sink literature (Xiao et al., 2024; Gu et al., 2025). The sink also accounts for nearly all of the gap between the two tables. Raw attention-mass coverage sits far below position coverage, 18–42% against 64–68%, but the raw number charges the tracked set with sink mass that no choice of tracked types could capture. The comparable quantity is the tracked share of *non-sink* attention, the final column of Table 4, labeled tracked/(tracked+other), which is 67–72% for every model measured on its full vocabulary. So when heads attend to token content at all, they place attention on tracked types at slightly above the tracked set's share of positions, and the shortfall in the raw number belongs to the sink.

What the untracked mass costs. Attention that leaves the tracked set still enters the prediction, through the two bucket values $v(\mathbf{b}^{\ell})$ and $v(\bar{\mathbf{c}}^{\ell})$ of Appendix D, and thus the kernel remains row-stochastic. However, some detail is lost. Attention to untracked types is answered by one shared vector rather than by a value per type, so a model with a lower tracked share has more of its heads' behavior compressed into that single average.

F Mean-Field Predictions Across Corpora

As described in the body, mean field analysis concerns the average behavior of a model over a given corpus. The results in the body of the paper use OpenWebText as the reference corpus. Here we describe how mean-field prediction of centroids varies across four different corpora. We demonstrate that, across the corpora we study, differences between predictions are very small, and that centroid predictions generalize out of distribution (i.e., cross-corpora).

F.1 Corpora Studied

Models. We evaluate corpus differences using the same models as studied in the paper body: GPT-2 small and Gemma-2 2B. Statistics are re-extracted from scratch for each model, and each model's shared vocabulary is re-derived as *its* 1000 most frequent OpenWebText types.

We choose corpora to reflect a diversity of ordinary English prose:

OpenWebText (OWT) (Gokaslan et al., 2019). The reference corpus, used in the paper body. This corpus consists of web pages linked from Reddit, collected to reproduce GPT-2's training data (Radford et al., 2019).

FineWeb (Penedo et al., 2024). This is a recent, heavily filtered crawl of the general web. We use the 10-billion-token sample. Like OWT, this corpus is general web text, but collected separately with different filtering.

Wikipedia (WikiText-103) (Merity et al., 2017). This consists of 'featured' and 'good' Wikipedia articles, in raw form. This corpus is stylistically different from web text in that it is more formal and more encyclopedic in subject matter.

Pre-1919 books (PG-19) (Rae et al., 2020). This set consists entirely of books published before 1919 from Project Gutenberg. As such, it has a literary style, uses older vocabulary and constructions, and has no web artifacts.

All corpora are prepared identically: text is tokenized with the model's tokenizer and packed into 512-token windows (windows may span document boundaries), and statistics are accumulated over 2 million tokens per (corpus, half) sample using TransformerLens (Nanda & Bloom, 2022). Throughout, the vocabulary is fixed to the 1000 most frequent types of the OWT reference sample, so that all results are over the same set of types.

F.2 Mean Field Predictions Are Stable Across Corpora and Generalize Out of Distribution

For each corpus X we estimate the mean-field kernel and roll out predicted centroids on X's statistics half. For each corpus Y we measure true centroids on Y's evaluation half. For each pair (X, Y) we initialize the rollout from X's layer-0 centroids, roll it out with X's kernel through the model's weights, and score the predicted centroids at every layer against Y's true centroids. With four corpora this gives a 4×4 matrix of runs, which we ran in full for both models.

Figure 10 shows centroid prediction accuracy using the same open-loop prediction as Section 4.2. We summarize those figures in Figure 9, where the diagonal cells ($X = Y$) show the average cosine similarity of mean-field prediction of centroids for that corpus (average over curve values in Figure 10). The off-diagonal cells assess generalization: whether statistics measured on one kind of text predict behavior on another.

Table 5 shows bootstrapped 95% confidence intervals for Figure 9, and additionally the context-free baselines. The context-free baseline runs each type through the model alone, as the two-token input $[\text{BOS}, t]$, and scores the resulting per-layer residual against the true centroids exactly as a prediction is scored. Using no corpus statistics beyond the layer-0 initialization, it is the floor that any corpus-derived kernel must clear. The context-free forward does not depend on a corpus, so a single set of residuals serves every column and only the evaluation centroids change.

We make three observations.

1. Mean-field centroid prediction replicates well on each corpus. Diagonal cells are all at or near the model's published OpenWebText level, including for pre-1919 books. **Thus the conclusions of Section 4.2 are not specific to the OWT corpus used.**
2. **OOD generalization is strong** (off-diagonal cells in Figure 9). Training the mean-field on one corpus and predicting centroid evolution on a different corpus yields accuracy generally close to the same-corpus evaluation. The worst mismatched cell clears the context-free baseline in every column and in both models. Statistics from century-old books still carry usable information about how the model treats modern web text.

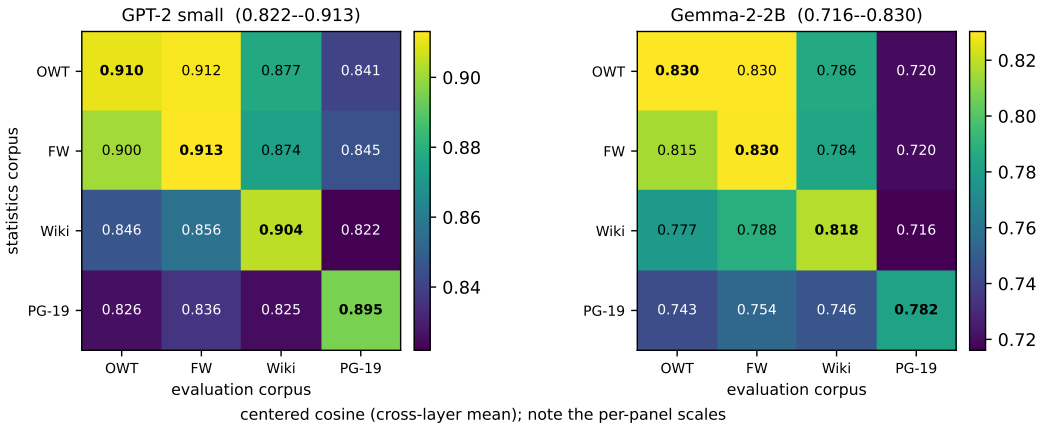

Figure 9: Average cosine similarities of centroid predictions. Rows: corpus supplying the mean-field statistics. Columns: corpus supplying the evaluation centroids. Cell values are the cross-layer mean centered cosine; diagonal (matched) cells in bold.

Table 5: Values from Figure 9 showing confidence intervals as well as context-free baseline

model	statistics	eval OWT	eval FineWeb	eval Wikipedia	eval PG-19
GPT-2	OWT	**0.910** [.907,.913]	0.912 [.910,.915]	0.877 [.873,.881]	0.841 [.835,.846]
	FineWeb	0.900 [.897,.904]	**0.913** [.911,.916]	0.874 [.870,.878]	0.845 [.839,.850]
	Wikipedia	0.846 [.838,.855]	0.856 [.848,.865]	**0.904** [.901,.907]	0.822 [.812,.831]
	PG-19	0.826 [.818,.833]	0.836 [.828,.842]	0.825 [.817,.832]	**0.895** [.889,.900]
	context-free	*0.799*	*0.807*	*0.781*	*0.745*
Gemma-2	OWT	**0.830** [.826,.834]	0.830 [.826,.834]	0.786 [.781,.791]	0.720 [.713,.726]
	FineWeb	0.815 [.810,.820]	**0.830** [.826,.833]	0.784 [.779,.789]	0.720 [.712,.727]
	Wikipedia	0.777 [.767,.786]	0.788 [.779,.797]	**0.818** [.814,.822]	0.716 [.705,.728]
	PG-19	0.743 [.734,.752]	0.754 [.745,.762]	0.746 [.738,.754]	**0.782** [.775,.788]
	context-free	*0.686*	*0.699*	*0.681*	*0.607*

3. OOD generalization degrades according to expected dissimilarity of corpora, with generalization between OWT and FineWeb best, and pre-1919 novels least. All within-column differences are statistically distinguishable

G Full Mean Field Results

G.1 Evaluation Metrics

We compare the mean-field predictions $\hat{\mathbf{c}}_t^\ell$ of Algorithm 1 with the true centroids $\mathbf{c}_t^\ell$ measured on the held-out corpus half, over the evaluation set $\mathcal{T}$ of the 1000 most frequent types (experimental settings in §4.2). Because the representation cloud is strongly anisotropic — it sits far from the origin relative to its own size (Appendix H) — every metric is computed after mean-centering each cloud by its own mean: writing $\bar{\mathbf{c}} = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathbf{c}_t^\ell$ and $\bar{\hat{\mathbf{c}}}$ analogously for the predictions, we score the centered quantities $\tilde{\mathbf{c}}_t = \mathbf{c}_t^\ell - \bar{\mathbf{c}}$ and $\tilde{\hat{\mathbf{c}}}_t = \hat{\mathbf{c}}_t^\ell - \bar{\hat{\mathbf{c}}}$. Without centering, the shared offset would dominate every similarity. The four metrics we use are:

- *Centered cosine similarity* — the mean over types of the per-type cosine, $\frac{1}{|\mathcal{T}|}\sum_t \cos\big(\tilde{\hat{\mathbf{c}}}_t,\ \tilde{\mathbf{c}}_t\big)$: per-type directional agreement, aggregated by a simple mean.
- *Representation similarity analysis (RSA)* — we compute the pairwise cosine similarities among the centered true centroids $\{\tilde{\mathbf{c}}_t\}$, the same among the centered predictions $\{\tilde{\hat{\mathbf{c}}}_t\}$, and report the Pearson correlation between the two lists of off-diagonal entries. RSA scores

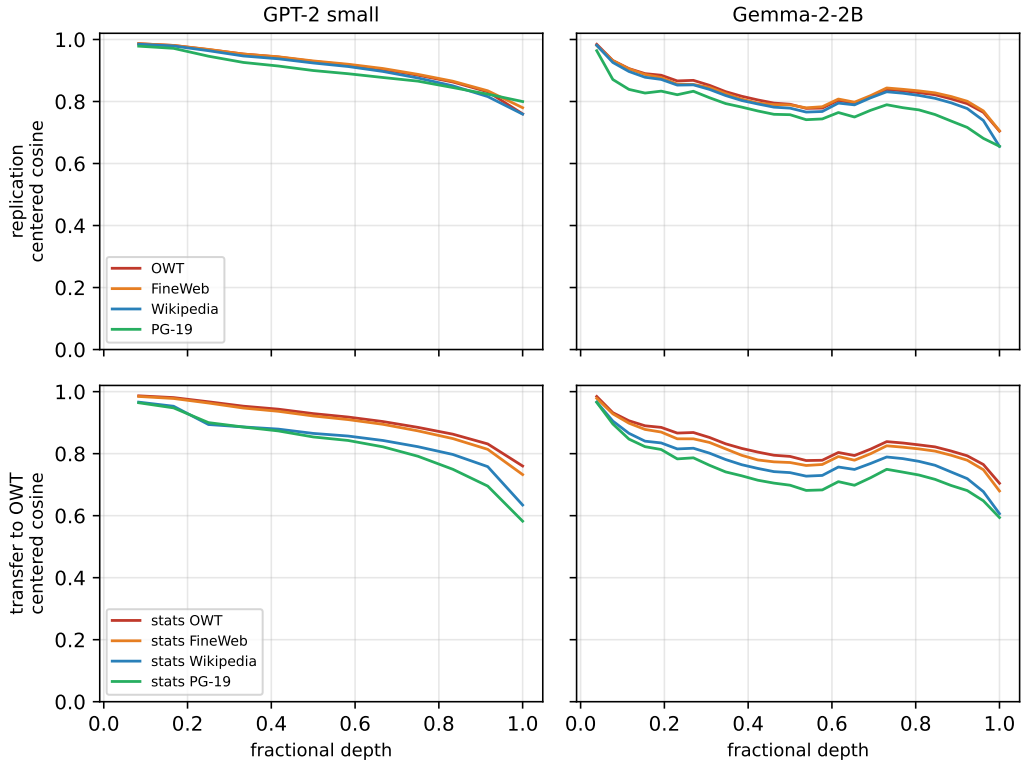

Figure 10: Per-layer rollout quality by fractional depth, one column per model. Top row: replication (statistics and evaluation on the same corpus), one curve per corpus. Bottom row: generalization (statistics from each corpus, evaluated against OpenWebText centroids).

whether the predicted cloud has the right *shape* — which types lie near which — independently of any global rotation of the cloud.

- *Centered relative error* — the ratio of means $\sum_t \|\tilde{\hat{\mathbf{c}}}_t - \tilde{\mathbf{c}}_t\| \,/\, \sum_t \|\tilde{\mathbf{c}}_t\|$. A ratio of means rather than a mean of per-type ratios, so types lying near the cloud center do not arbitrarily inflate the metric; this is the one metric jointly sensitive to direction and magnitude.
- *Centered norm ratio* — $\sum_t \|\tilde{\hat{\mathbf{c}}}_t\| \,/\, \sum_t \|\tilde{\mathbf{c}}_t\|$, the predicted cloud's mean radius relative to the true cloud's, isolating magnitude: 1 is ideal, values above 1 indicate overshoot.

§4.2 reports only the first of these four metrics: Figure 3 in §4.2 shows centered cosine for the kernel-source substitutions. We now provide full details on those substitutions and extend our results to all four metrics, and across Gemma-2, GPT-2 and Qwen-3-14B.

G.2 Kernel Ablations

The results comparing the mean-field kernel against the alternative kernels for all four metrics are depicted in Figure 11. Our first substitute is `context-free`: a baseline that propagates each type through the network in isolation (i.e., running the model on just [BOS, $\mathbf{c}_t$]). The mean field improves on it at every layer, and increasingly so with depth. Second, because the mean field is a corpus-average object, a natural question is whether we need to compute attention at all: can co-occurrence statistics suffice? In the `cooc` alternative, the type-to-type distribution is taken from a symmetric ± 5-word co-occurrence matrix over the corpus, normalized to be row-stochastic, and used to distribute attention across the centroids, corrected for the fraction of attention allocated to the top 1000 tokens.[4] The mean field outperforms `cooc` as well. Finally, we test whether the kernel's

[4]We normalize $\tilde{C}_{st} = C_{st} / \sum_{t'} C_{st'}$, form the row sums $T_s = \sum_{t=1}^{1000} P_{st}$, and use $T_s \tilde{C}_{st}$ as the attention substitute.

depth structure can be simplified. `P0only` reuses the *layer-0* kernel at every layer (head by head); `Pavg` replaces it with the kernel averaged over layers. Both are distinctly worse than the measured mean field, so the kernel's variation across depth carries important content for prediction.

We next ask whether the specific structure of the kernel P is necessary, or whether a structureless version could do as well (Figure 12). We study three cases. `meanq` (query-blind): every query row of P is replaced by the average row, so every type reads the same mixture of key types, contributing an identical vector to every type. `uniform` (key-blind): each query's total budget $\sum_t P_{s,t}$ is preserved but spread evenly over key types. `rowshuffle` (negative control): the query rows are permuted randomly, so each type receives another type's attention profile; this distinguishes "the graph structure of P is uninformative" (shuffling would then match `meanq`) from "P's graph carries type-specific information" (shuffling should then do *worse*). None of the three matches the measured kernel, so accurate prediction requires type-level distinctions among both queries and keys, increasingly so with depth. Table 6 gives the cross-layer summary for all of these alternatives.

Table 6: Cross-layer fidelity of the mean-field rollout, the context-free baseline, `cooc`, and mean-field ablations: mean (standard error over layers) of the centered cosine and centered relative error (lower is better).

	GPT-2		Gemma-2		Qwen-3-14B	
	cosine	rel. error	cosine	rel. error	cosine	rel. error
Mean field	**0.907** (0.019)	**0.434** (0.051)	**0.830** (0.011)	**0.593** (0.026)	**0.816** (0.009)	**0.640** (0.023)
Context-free	0.797 (0.033)	0.643 (0.063)	0.686 (0.018)	0.809 (0.024)	0.521 (0.021)	1.183 (0.036)
`cooc`	0.834 (0.026)	0.567 (0.042)	0.724 (0.015)	0.738 (0.027)	0.258 (0.020)	*1.429** (0.067)
`meanq`	0.786 (0.038)	0.615 (0.050)	0.682 (0.017)	0.786 (0.028)	0.430 (0.014)	0.949 (0.004)
`uniform`	0.776 (0.038)	0.630 (0.054)	0.666 (0.018)	0.809 (0.030)	*0.061* (0.027)	*96.1** (12.9)
`rowshuffle`	0.672 (0.034)	0.754 (0.036)	0.616 (0.021)	0.894 (0.035)	0.245 (0.009)	1.042 (0.013)
`P0only`	0.561 (0.074)	1.124 (0.153)	0.712 (0.019)	0.747 (0.034)	0.591 (0.024)	1.091 (0.055)
`Pavg`	0.732 (0.039)	0.830 (0.069)	0.702 (0.022)	0.740 (0.031)	*0.114* (0.036)	*24.4** (3.0)

***Note on the Qwen-3-14B ablations.** Qwen's `cooc`, `uniform` and `Pavg` entries show large relative errors which are produced by a numerical instability of that model's rollout concentrated at layer 0 and not a reflection of kernel quality; Appendix I gives the full accounting.

G.3 Understanding Approximation Errors

Algorithm 1 incorporates two approximations at every layer: the mean-field prediction of each head's contribution $\hat{\mathbf{z}}_s^{\ell h}$, and the application of the MLP once at the resulting centroid. Only the first is a property of mean-field attention; the second is a property of the rollout as a way of propagating centroids.

Accordingly, we first ask: how good is the mean-field approximation of attention? We measure this by comparing $\overline{\mathbf{z}}$ with its approximation $\hat{\mathbf{z}}$. Figure 13 does this teacher-forced: starting from the *true* centroids at each layer, it compares the predicted attention-block output $\sum_h \hat{\mathbf{z}}_s^{\ell h}$ against the same quantity measured over the corpus, $\sum_h \overline{\mathbf{z}}_s^{\ell h}$, averaging across all types s.[5] The prediction is good at every layer of every model: across all five models, cross-layer mean centered cosine lies between 0.78 and 0.87, and the norm ratio stays near one (cross-layer means 0.91 to 1.24), so the mean field does well at recovering both the direction and the scale of what each head writes.

We next decompose the per-layer error using three increasingly approximate estimators of the layer-$\ell{+}1$ centroid, measured for GPT-2, Gemma-2, and Qwen-3-14B; Llama is discussed below. All three are applied to the true layer-ℓ centroids and end with the MLP applied once at the resulting centroid; each makes one more approximation in the attention block than the last. Estimator (a) isolates the MLP gap by using exact attention for the rest of the prediction ($\mathbf{z}_s$, per (3)), so that applying the MLP at the centroid is its only approximation.[6] Estimator (b) adds the further approximation of taking the per-type value at the centroid, i.e., replacing $\mathbb{E}_s[\overline{\mathbf{v}}_t]$ with $v(\mathbf{c}_t)$ (still keeping the attention-covariance term); and estimator (c) applies the full mean field of (4), which additionally drops the covariance term.

[5] Bootstrap CIs over types are less than $\pm$ 0.01 and so not shown.

[6] For Gemma-2, this also includes the post-attention norm at the centroid.

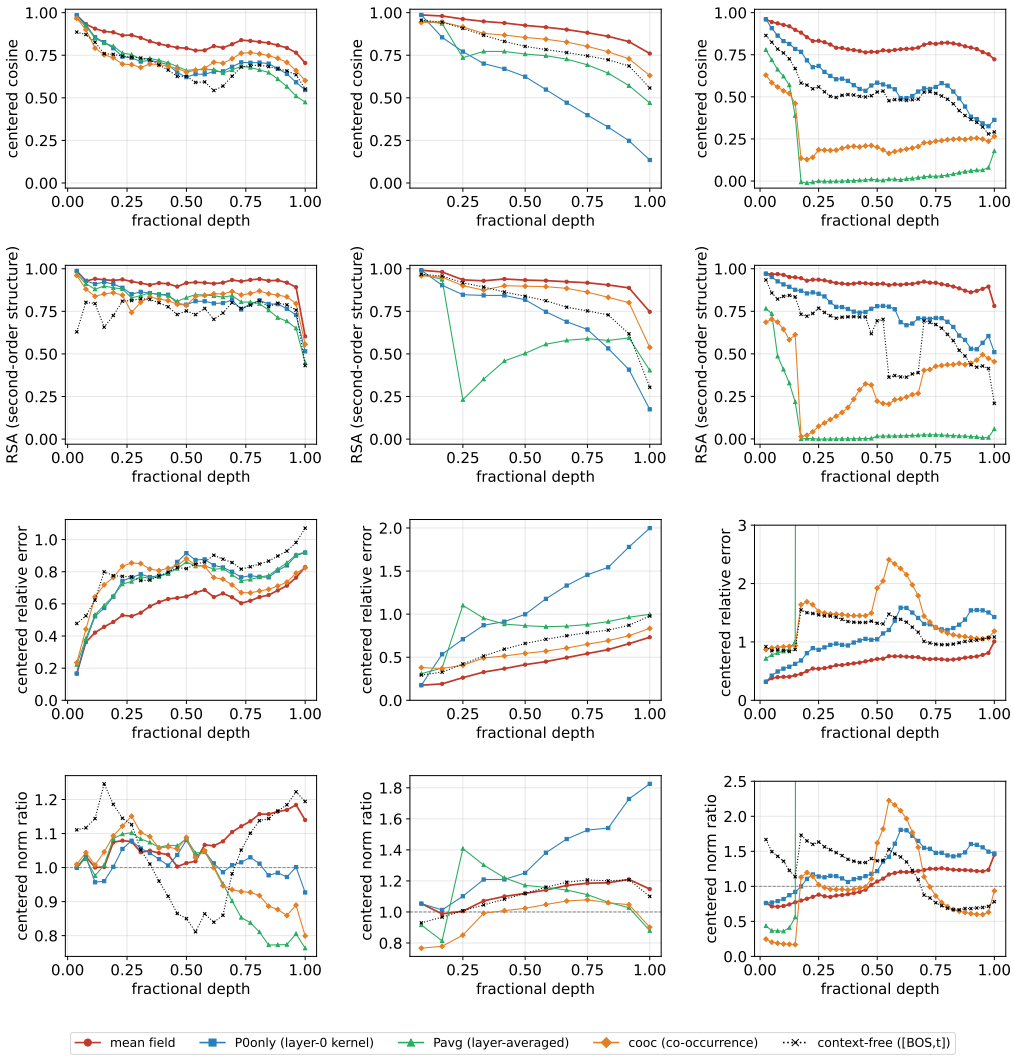

Figure 11: Mean-field predictions are accurate and superior to alternative kernels (kernel-source substitutions). Left: Gemma-2; middle: GPT-2; right: Qwen-3-14B. Rows (top to bottom): centered cosine similarity, representation similarity analysis (RSA), centered relative error, and centered norm ratio. For Qwen the `uniform` and `Pavg` substitutions diverge in magnitude; their relative-error and norm-ratio curves leave the plotted range.

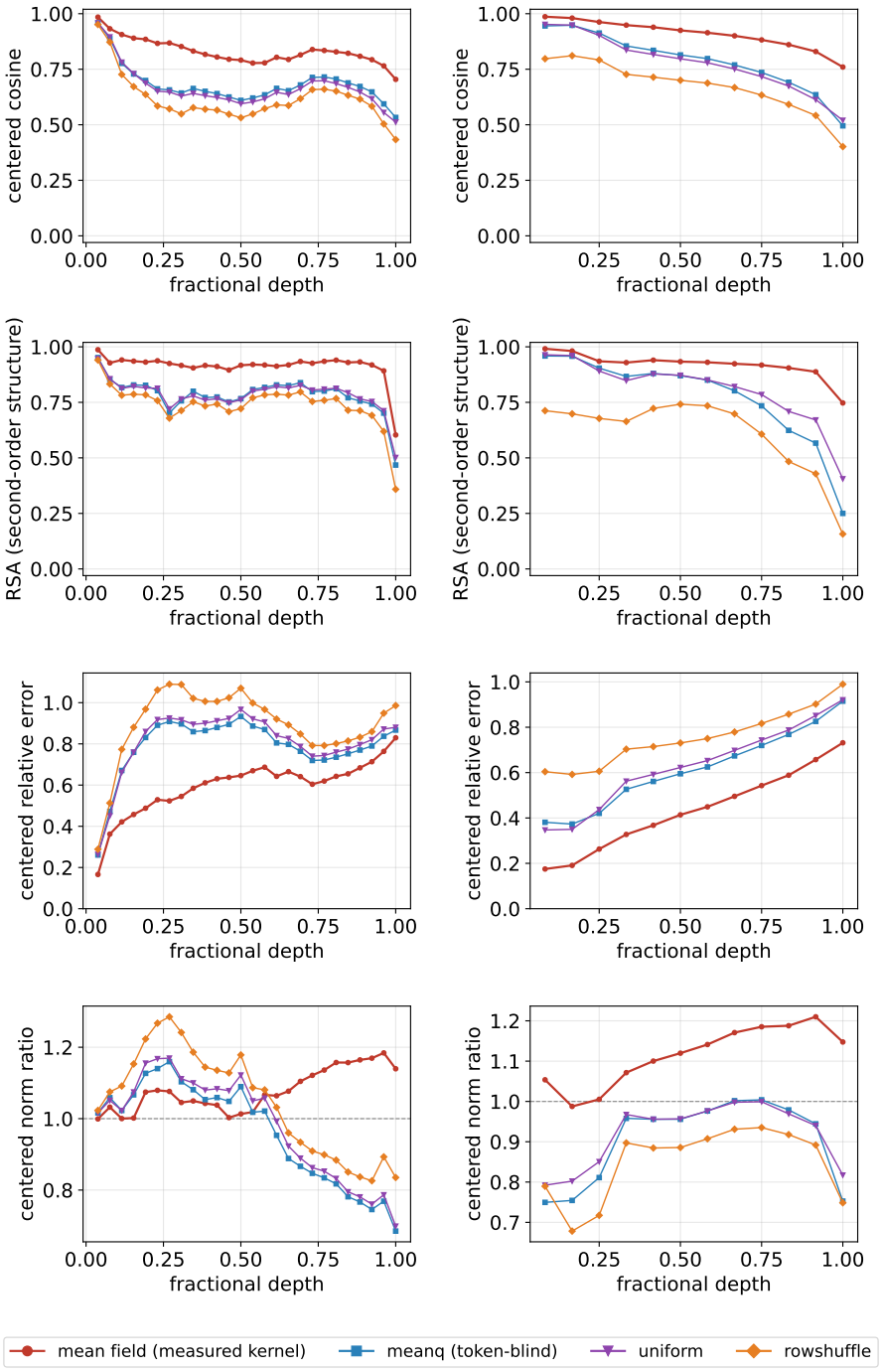

Figure 12: Mean-field predictions are more accurate than the attention-recipe ablations. Left: Gemma-2, Right: GPT-2. Rows (top to bottom): centered cosine similarity, representation similarity analysis (RSA), centered relative error, and centered norm ratio.

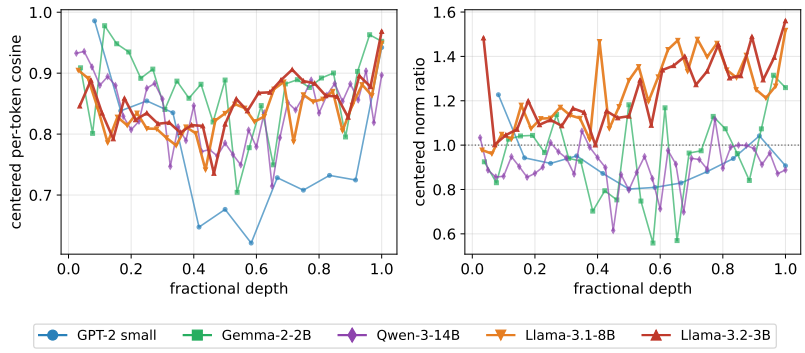

Figure 13: Teacher-forced prediction of the head contribution $\bar{\mathbf{z}}$, by fractional depth. Left: centered cosine; right: centered norm ratio, with the dotted line marking zero error.

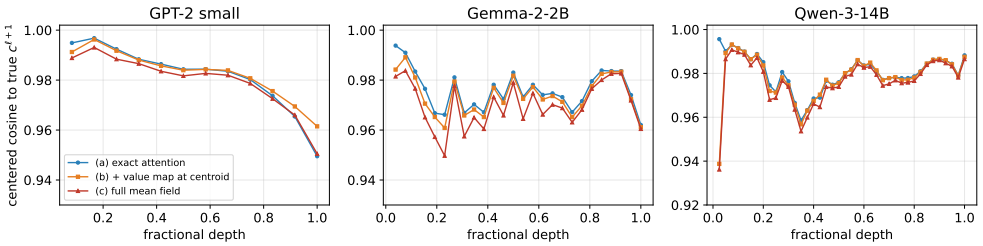

Figure 14: Per-layer accounting of mean-field approximation error. Starting from the true centroids at layer ℓ, three nested predictions of the layer-$\ell{+}1$ centroid are compared to the true value (centered cosine): (a) exact attention with the MLP applied at the centroid; (b) additionally taking the value map at the centroid; (c) the full mean field of (4). Left: GPT-2; middle: Gemma-2; right: Qwen-3-14B.

The decomposition confirms that MLP-introduced error is the main source of rollout inaccuracy (Figure 14): the dominant per-layer source of error is the MLP evaluated at the centroid. Estimator (a) (exact attention at the centroid rather than per occurrence) accounts for almost all of the per-layer error, falling short of perfect prediction by approximately $0.02 - 0.03$ across all models in centered cosine similarity on average. The two further approximations then add little: taking the value map at the centroid ((a)$\rightarrow$(b)) and dropping the covariance term ((b)$\rightarrow$(c)) each cost at most $\sim$0.005 per layer. The final error of the full open-loop rollout (Figure 11) therefore does not arise from any single step being inaccurate, but from the accumulation of these small per-layer errors across depth, with the MLP-at-centroid approximation the leading per-step contributor. Measured on its own, the MLP gap is less significant than this may suggest, as we explain in Appendix K.

The same decomposition explains the two Llama models, whose rollouts predict centroid evolution distinctly less well than the others (full details in Appendix J). The mean-field *attention* prediction for Llama is as accurate as for the other models (Figure 13); the error is in the MLP-at-centroid step, whose Jensen gap is larger because many of Llama's centroids, especially in early layers, sit where the MLP is more sharply curved. Almost all of the Llama-3.2-3B rollout error traces to poor MLP approximation at layer 1. Because this is a property of the MLP rather than of the mean-field attention, Llama's centroid rollout requires a Jensen-gap correction to be accurate.

H Anisotropy of Representations

As noted in § 4.2, in both models the cloud of type centroids sits far from the origin relative to its own size. In particular, the all-type centroid $\bar{\mathbf{c}}$ grows in magnitude from 35 to 379 across GPT-2's layers and from 25 to 424 across Gemma-2's, while the mean Euclidean distance of a type centroid to $\bar{\mathbf{c}}$ stays a fraction of that magnitude — around $0.8\,\|\bar{\mathbf{c}}\|$ in GPT-2's early and middle layers and falling to $0.3\,\|\bar{\mathbf{c}}\|$ at the last layer; in Gemma-2, it starts above 1.0 but settles around 0.6–$0.7\,\|\bar{\mathbf{c}}\|$

at deeper layers. Table 7 reports, per layer for both models, the magnitude of the all-type centroid $\bar{\mathbf{c}} = \frac{1}{N}\sum_t \mathbf{c}_t$, the mean Euclidean distance $\|\mathbf{c}_t - \bar{\mathbf{c}}\|$ of a type centroid from the cloud center, and their ratio mean dist/$\|\bar{\mathbf{c}}\|$. The table shows that representations occupy a compact region offset from the origin, the motivation for mean-centering before computing cosine similarities.

Table 7: Anisotropy of centroid representations, per layer: the magnitude of the all-type centroid $\|\bar{\mathbf{c}}\|$, the mean Euclidean distance of a type centroid from the cloud center, and their ratio. Gemma-2's layers appear in two columns.

GPT-2				Gemma-2							
ℓ	$\|\bar{\mathbf{c}}\|$	dist.	ratio	ℓ	$\|\bar{\mathbf{c}}\|$	dist.	ratio	ℓ	$\|\bar{\mathbf{c}}\|$	dist.	ratio
0	2.4	2.7	1.12	0	25.5	78.3	3.07	14	114.5	72.3	0.63
1	34.8	26.6	0.76	1	33.8	66.8	1.97	15	120.7	77.9	0.65
2	39.5	31.8	0.81	2	50.6	61.8	1.22	16	132.8	90.5	0.68
3	44.9	35.2	0.78	3	43.7	60.2	1.38	17	154.4	101.4	0.66
4	47.8	38.3	0.80	4	49.9	55.3	1.11	18	157.0	110.0	0.70
5	50.5	40.8	0.81	5	47.0	56.1	1.19	19	177.7	127.1	0.72
6	54.1	44.1	0.81	6	50.2	54.8	1.09	20	185.3	139.0	0.75
7	58.9	48.1	0.82	7	51.4	55.8	1.09	21	213.9	153.5	0.72
8	67.0	53.7	0.80	8	64.7	56.8	0.88	22	248.8	174.0	0.70
9	77.4	59.9	0.77	9	68.8	58.1	0.84	23	279.4	187.1	0.67
10	99.1	67.9	0.69	10	78.6	60.5	0.77	24	314.6	203.1	0.65
11	166.3	78.6	0.47	11	82.8	62.0	0.75	25	361.4	222.1	0.61
12	379.1	114.8	0.30	12	96.2	64.7	0.67	26	424.3	280.6	0.66
				13	94.5	65.5	0.69				

I Impact of Rogue Dimensions on Results

In every model, a handful of the D dimensions of the representation $\mathbf{x}_i \in \mathbb{R}^D$ carry magnitudes far above the rest, and through the block's normalization a small error in one perturbs every dimension on the next rollout step. This appendix examines the effect of those "rogue" dimensions. Setting them to their corpus value (which we call 'pinning') is a small, principled correction that modestly improves most models (§I.1). Note, though, that the results in the body all use the unmodified (unpinned) representations, and the effects of the correction are examined here in the appendix only. As an aside, we note that these rogue dimensions also seem to play a role in the instability results for Llama-3.2-3B that we analyze in Appendix J.

I.1 Rogue-Dimension Pinning: a Modest Correction

The phenomenon. Every transformer we examine carries a handful of dimensions one to two orders of magnitude above the median: referred to as "rogue dimensions" of GPT-2 and BERT (Timkey & van Schijndel, 2021; Kovaleva et al., 2021; Puccetti et al., 2022), "outlier features" that emerge with scale (Dettmers et al., 2022), and "massive activations" documented across the LLaMA family and other modern models (Sun et al., 2024). The same dimensions underlie the representation anisotropy of Gao et al. (2019a) and Mu et al. (2018). Because each transformed block normalizes its input by a scale computed over all D dimensions (root-mean-square for the RMSNorm models, standard deviation for GPT-2's LayerNorm), a rogue dimension dominates that scale, so a small relative error in its predicted value perturbs every other dimension on the following step. Pinning these dimensions to their corpus value removes that coupling.

Why removing them is justified. These dimensions are approximately *context-invariant*, taking nearly the same value whatever the input token, and act as an implicit bias or attention-sink mechanism rather than as carriers of token values (Sun et al., 2024). These dimensions have a direct link to attention sinks (the tendency of heads to deposit excess probability mass on a few uninformative positions) (Xiao et al., 2024; Gu et al., 2025). Bondarenko et al. (2023) show that this "do-nothing" routing of attention is what produces the outlier activations. Holding these dimensions at their corpus value during the rollout thus mainly removes a nuisance term, rather than a signal the mean field is asked to reproduce.

Table 8: Number of dimensions pinned during the rollout by model, and against model width D.

model	D	pinned dimensions
GPT-2	768	10
Gemma-2-2B	2304	25
Llama-3.2-3B	3072	10
Llama-3.1-8B	4096	14
Qwen-3-14B	5120	53

Table 9: Qwen-3-14B rollout under each kernel, unpinned versus pinned: cross-layer mean centered cosine (standard error over the 40 layers) and mean centered relative error.

	centered cosine		relative error	
kernel	unpinned	pinned	unpinned	pinned
Mean field	0.816 (0.009)	0.826 (0.009)	0.640	0.721
`cooc`	0.258 (0.020)	0.498 (0.008)	1.429	0.954
`Pavg`	0.114 (0.036)	0.622 (0.009)	24.42	0.892
`rowshuffle`	0.245 (0.009)	0.456 (0.006)	1.042	0.969
`meanq`	0.430 (0.014)	0.503 (0.007)	0.949	0.928
`P0only`	0.591 (0.024)	0.628 (0.021)	1.091	1.027
`uniform`	0.061 (0.027)	0.089 (0.029)	96.11	31.91

Identifying the set. We pin a dimension if at some layer its per-dimension RMS over the token centroids exceeds $15\times$ that layer's median, a threshold set by the wide magnitude gap separating these dimensions from the bulk; it selects at or below 1% of D in each model (Table 8). The specific indices are in the code release.

Effect of pinning. Pinning is a refinement that produces a modest improvement in accuracy of rollout prediction. Pinning leaves the teacher-forced prediction and the early rollout layers unchanged, and it improves the final-layer rollout for the models whose rollout error runs through these dimensions: for GPT-2 the centered cosine rises $0.76 \to 0.84$ (relative error $0.73 \to 0.57$), for Gemma-2 $0.70 \to 0.82$ ($0.83 \to 0.56$), and for Qwen-3-14B $0.72 \to 0.82$. For the two Llama models it is neutral to slightly negative (Llama-3.1-8B $0.72 \to 0.71$, Llama-3.2-3B $0.60 \to 0.58$), because their rollout error is not dominated by these dimensions, but by the mechanism in Appendix J.

I.2 Rogue Dimensions and the Qwen-3-14B Ablations

Qwen-3-14B is the one model where pinning changes the *interpretation* of a result rather than just a quantitative improvement. The mean-field rollout of Qwen-3-14B is barely affected by pinning. However, three substitute kernels whose rollouts drifted far off course (`Pavg`, `cooc`, and `rowshuffle`) as shown in Figure 11 (right-hand column), are dramatically improved (and the remaining four improve marginally) (Table 9). The rogue dimensions are a conditional amplifier: with the mean-field kernel the predicted norm never leaves the stable regime, whereas some substitute kernels drive the trajectory off course enough for the mispredicted rogue values, amplified through the shared RMSNorm scale to produce a discrete blow-up. In those cases, pinning removes the blowup. Rogue dimensions therefore do not penalize every kernel uniformly; they engage only once a prediction has already drifted, which is why the mean-field kernel escapes their effect.

The sensitivity of Qwen-3-14B to rogue dimensions is almost entirely a property of a single layer. Applying a substitute kernel to a chosen subset of layers and the mean-field kernel everywhere else localizes the damage (Table 10): substituting at layer 0 alone incurs 0.279 of the 0.327 all-layers total (an 85% share), while substituting at all thirty-nine *other* layers costs only 0.065 (a 20% share). The two shares sum to 105%, so the decomposition is nearly clean, and layer 0 carries more than four times the damage of the other thirty-nine combined. The same measurement on GPT-2 spreads the damage across depth instead, as one expects of a rollout whose error accumulates layer by layer. Qwen's apparent hypersensitivity to kernel substitution is therefore an early-layer property rather than depth compounding, in the same family as the layer-1 amplification that destabilizes Llama-3.2-3B's rollout (Appendix J), though by a different mechanism.

Table 10: Localizing the damage from a kernel substitution in Qwen-3-14B, pinned. The substitute (`cooc`) is applied only to the listed layers, with the measured kernel used everywhere else; entries are cross-layer mean centered cosine over the 40 layers and the shortfall against the measured kernel.

layers given the substitute kernel	centered cosine	damage
none (measured kernel)	0.826 (0.009)	—
all 40 layers	0.498 (0.008)	0.327
layer 0 only	0.547 (0.008)	0.279
all layers except 0	0.761 (0.011)	0.065

Pinning is not uniformly beneficial: the mean field's own relative error *worsens* slightly under it ($0.640 \rightarrow 0.721$) even as its cosine improves, and `uniform` stays broken when pinned, so the amplifier is not the only thing defeating that kernel.

J The Llama Models: Accurate Per-Layer Prediction, Failed Rollout

We extended the mean-field validation of Section 4.2 to Llama-3.2-3B and Llama-3.1-8B (both RMSNorm, RoPE, grouped-query attention; statistics from a 2M-token OpenWebText pass over the top-1000 Llama token types). We find that per-layer accuracy matches the other models (Figure 13). However, the open-loop rollout does not, and exposes a failure mode specific to Llama. We work out this dependency here, through a sequence of targeted experiments. While our experiments serve to *eliminate* a number of natural hypotheses, we ultimately lack a crisp explanation. We report the eliminations along with the final characterization, since several of the eliminated explanations could have been reasonable hypotheses *a priori*.

Summary of results. First, we affirm that the mean-field yields reasonably accurate predictions layer-by-layer; it is only when errors accumulate through the open-loop rollout that predictions diverge. We show using teacher-forced predictions that the mean field predicts each layer's centroid update for both Llamas about as well as for the other models: cross-layer mean centered cosine 0.858 (± 0.017) for Llama-3.2-3B and 0.837 (± 0.019) for Llama-3.1-8B, with the familiar mid-stack dip (minima 0.69 and 0.62) and recovery.

Figures 15–16 give the Llama counterparts of Figures 3 and 12, and Figure 17 the counterpart of Figure 14. Iterated open-loop, Llama-3.1-8B behaves approximately like GPT-2, Gemma-2, and Qwen-3: cross-layer mean rollout cosine 0.719 (± 0.019) with cosine similarity at the final-layer of 0.72. Notably, Llama-3.2-3B instead *diverges*: the predicted centered magnitude inflates to $4.9\times$ the true value, the rollout cosine collapses to 0.30 by layer 2 and averages 0.499 (± 0.028), and the centered relative error exceeds 1 from mid-stack on.

The teacher-forced prediction is not severely affected (Figure 17). These per-layer measurements remain the important mean-field fidelity measure for this model. The rest of this section concerns why the *iteration* fails.

The divergence is a single-step amplification in the layer-1 MLP. Tracking the rolled-out and true trajectories together, the error entering the layer-1 step is amplified $17\times$ in that one step and thereafter merely persists ($\sim 1.05\times$ per later step). We note three properties of the error. (i) It is a *curvature* effect and not a large derivative. Rescaling the incoming error to $\alpha\delta$ and re-running the block, the amplification grows nonlinearly with the amount that we scale the error. So the block's derivative at the true centroid is benign in the direction the error actually points — at small amplitude the real error and a random one are amplified almost equally ($1.6\times$ versus $1.2\times$). What matters is that when the error is large enough, 36% of the centroid norm, it leaves the region over which the MLP is affine. (ii) It is *in the MLP*: decomposing the step, the attention update expands the error by $1.06\times$; the subsequent MLP takes it to $17\times$. (iii) It is *model-specific*: the same measurement gives $\sim 1\times$ at every layer for GPT-2, Gemma-2, Qwen-3-14B, and Llama-3.1-8B. Further, the rogue-dimension pinning of Appendix I does not affect the error (final rollout cosine $0.602 \rightarrow 0.584$ for the 3B, $0.720 \rightarrow 0.708$ for the 8B): the amplified error is spread broadly across coordinates, with the largest massive-activation channel contributing only 1.2% of the error.

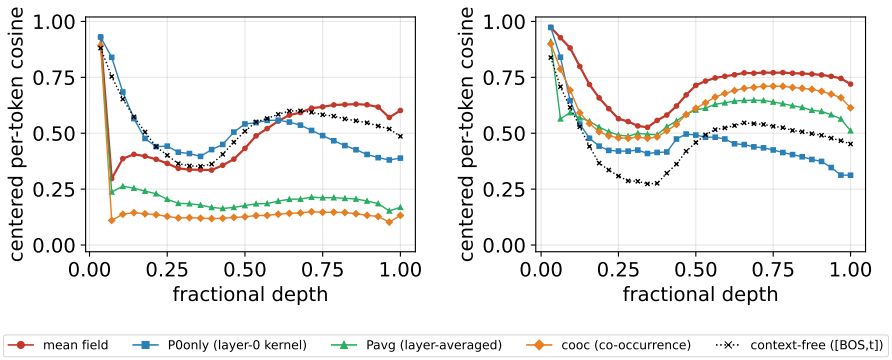

Figure 15: Kernel-source substitutions (as Figure 3) for the Llama models. Left: Llama-3.2-3B; right: Llama-3.1-8B. Centered per-token cosine of the open-loop rollout by fractional depth.

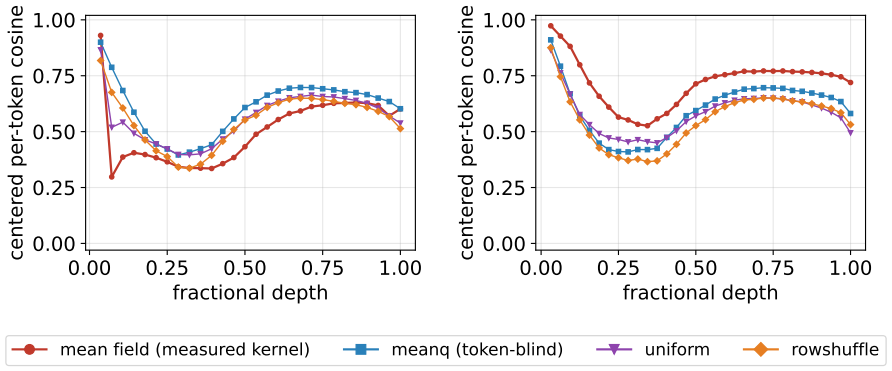

Figure 16: Attention-recipe ablations (as Figure 12) for the Llama models. Left: Llama-3.2-3B; right: Llama-3.1-8B.

Further investigation identifies that only a few gates (approximately 25) in the first MLP are responsible for the great majority of the added error. Those few gates account for the amplification outright. Because the MLP output is exactly linear in the post-activation, the response decomposes exactly over the MLP's 8192 gates; restricting it to the top 25 recovers 94.7% of the *nonlinear excess*, while an equal number of randomly chosen gates recovers 0.1%.

The amplified error concentrates on closed-class tokens. The per-token amplification is heavy-tailed: the median token's error grows 12×, the 99th percentile 143×, the maximum 323×, and the ten most-amplified tokens carry 53% of the total output-error energy. Of the fifty most-amplified, forty-three are newlines, punctuation, or clause-boundary tokens such as parentheses or "The.". These are all 'closed-class' tokens: high-frequency function words and structural markers rather than content words. Many start at early layers with tiny errors; the extreme case is token `').\n\n'` whose centroid error norm is 0.49 going into transformer block 1 and error 159 coming out of that block.

Exclusion experiment: structural tokens are not the cause. The concentration on boundary tokens suggests an obvious hypothesis: the structural tokens themselves drive the failure. We tested it by rerunning the entire pipeline — token selection, corpus statistics, and rollout, on a fresh 1M-token extraction — with every newline, whitespace, and pure-punctuation token excluded from the tracked vocabulary (67 of the original top-1000; capitalized words that initialized a clause were retained deliberately, as a probe). The divergence is unchanged: peak magnitude ratio 5.2 versus 4.9, mean rollout cosine 0.426 versus 0.499, teacher-forced accuracy unchanged (0.852). Token amplification simply *migrates* to remaining closed-class tokens: conjunctions (`but`, `and`, `or`), auxiliaries (`is`, `are`, `has`, `was`), negation stems (`wasn`, `didn`, `doesn`), determiners (`the`, `a`), and sub-word prefixes (`p`, `de`) — only one of the top twenty is a capitalized clause-initial word

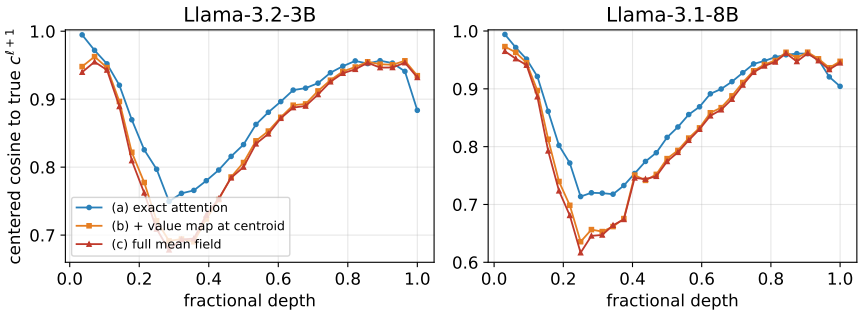

Figure 17: Per-layer (teacher-forced) error decomposition (as Figure 14) for the Llama models: (a) exact attention with the MLP at the centroid; (b) additionally the value map at the centroid; (c) the full mean field. Left: Llama-3.2-3B; right: Llama-3.1-8B. Note the lower vertical scale than Figure 14.

— with the same concentration (top ten $= 54\%$ of error energy). The same rerun for Llama-3.1-8B likewise changes nothing material (mean rollout cosine 0.615 versus 0.719, peak magnitude ratio 1.45 versus 1.70, teacher-forced unchanged at 0.827).

Attention-sink accounts, tested. The coincidence of boundary tokens, massive-activation channels, and layer 1 (where sink structure first forms) suggests the attention sink (Xiao et al., 2024; Sun et al., 2024) as a common cause. However, three versions of this idea all give insufficient explanations.

The first version is that boundary tokens develop large errors because they are local attention sinks. In fact, boundary tokens genuinely are *local* sink targets — they receive 1.4–$1.7\times$ more attention per occurrence than content tokens. However, the queries that feed them are not the queries that feed BOS, so boundary tokens are not standing in for the sink. Further, across tokens, how strong a "magnet" a token is does not predict its error amplification.

The second version is that boundary tokens develop large errors because they carry the massive activations. However, we find that boundary tokens' median magnitude in the massive-activation dimensions is 0.71, vs 0.73 for content tokens. So they are not unusual in their massive activation dimensions.

The third version is that boundary tokens are a mixture of tokens with high values of the massive activation dimension and lower values of the massive activation dimension. This would cause the MLP at the centroid to make strongly incorrect predictions. However, we find that the massive-activation dimension values are unimodel across boundary tokens. What survives is only a graded boundary signal in the massive activation dimension (1.2–$1.8\times$ higher for phrase- and sentence-initial or -final occurrences), too weak and too continuous to account for the amplification.

Capitalization pairs: position is not the driver. As a final positional test, we compared mean-field deviations across all 77 capitalized/lowercase pairs in the vocabulary (`The`/ `the`, `In`/ `in`, ...), capitalization being a near-perfect marker of sentence-initial position. There is no systematic effect: the paired median detonation ratio is 1.00 (Wilcoxon $p = 0.37$), and teacher-forced error is actually slightly *lower* for the capitalized forms ($0.81\times$, $p < 10^{-4}$). The tails are large in both directions — `The` detonates $3.6\times$ its lowercase twin and paragraph-initial `The` $7.1\times$, but `and` detonates $50\times$ more than `And` — so the well-known `The` case is atypical rather than representative.

What remains. The Llama-3.2-3B divergence is fully characterized: it is confined to a single step in the layer-1 MLP, specific to this model, and heavy-tailed over closed-class tokens, with its output concentrated in the massive-activation dimensions. The natural explanations are eliminated: it is not the structural tokens (exclusion experiment), not rogue-dimension coupling (pinning inert), not sink-magnet status, not a sink-writing gate, and not syntactic position (capitalization pairs). What is left is a curvature effect in this one model's layer-1 MLP, and it is now accounted for: the layer-0 mean-field error of high-frequency, closed-class tokens is concentrated enough to drive a handful

Table 11: MLP mean-field approximation, per layer: mean-centered cosine and relative error (ratio of means) between the occurrence-mean MLP contribution for type s and the MLP applied to the mean intermediate token $\mathbf{m}_s$, averaged over types. Gemma-2's layers continue in the right-hand block.

GPT-2			Gemma-2					
ℓ	cos	rel. err	ℓ	cos	rel. err	ℓ	cos	rel. err
0	0.995	0.333	0	0.983	0.190	13	0.806	0.866
1	0.966	0.360	1	0.952	0.411	14	0.856	0.636
2	0.828	0.717	2	0.947	0.381	15	0.894	0.591
3	0.742	0.865	3	0.892	0.606	16	0.887	0.613
4	0.650	1.073	4	0.877	0.662	17	0.913	0.555
5	0.650	1.118	5	0.815	0.970	18	0.894	0.643
6	0.686	1.045	6	0.860	0.609	19	0.868	0.668
7	0.752	0.903	7	0.791	1.025	20	0.880	0.595
8	0.778	0.856	8	0.778	0.917	21	0.885	0.563
9	0.748	0.859	9	0.765	0.965	22	0.861	0.637
10	0.773	0.783	10	0.766	0.898	23	0.860	0.609
11	0.697	0.909	11	0.726	0.917	24	0.871	0.656
			12	0.795	0.754	25	0.829	0.756

of SwiGLU gates across their nonlinearity, and those gates carry essentially all of the resulting amplification. This is the MLP-at-centroid (Jensen) approximation, identified in Section 4.2 as the leading per-step error source in every model, meeting an error large enough to leave the affine region. Why this model, and this layer, place their gates so close to the knee where the others do not, we leave as an open question.

K Impact of the MLP on Mean-Field Approximation

The last step of Algorithm 1 applies the layer's MLP once, to the mean intermediate representation $\mathbf{m}_s$ ($\mathrm{MLP}(\mathrm{LN}(\mathbf{m}_s))$), in place of averaging the MLP over the individual intermediate tokens of type s ($\mathbb{E}_{\mathrm{occ}}[\mathrm{MLP}(\mathrm{LN}(\mathbf{m}))]$). To quantify the resulting error, we accumulate both quantities per type and layer over a corpus and compare them. Table 11 reports, per layer, the mean-centered cosine and relative error (ratio of means) between the two quantities, averaged over types. For both GPT-2 and Gemma-2, the behavior follows a similar pattern. The approximation is near-exact at the first layers (cosine ≈ 0.99 for both models), weakest in the middle of the network (cosine ≈ 0.65 for GPT-2 at layers 4–5, ≈ 0.73 for Gemma-2 at layer 11), and improved somewhat at later layers. Overall, the centered cosine averaged over layers is 0.77 (GPT-2) and 0.86 (Gemma-2).

These numbers are what make the MLP step the dominant term in the error decomposition of §4.3, yet a cosine of 0.65 (the bark) sounds far worse than the per-layer shortfall of 0.02–0.03 (the bite) reported there. Two things account for the difference. The cosines above measure the MLP contribution against itself, whereas the decomposition measures the predicted centroid, which carries the attention output alongside the MLP contribution, so a given disagreement in the MLP term costs less in the centroid than it does on its own. And most of the disagreement is a shared offset in *magnitude* rather than an error in *direction*.

L Sensitivity to the Top-N Truncation

To assess the sensitivity to truncation to the top-N types, we re-ran the substitution of §5 for Pythia-160M at $N = 2000$ and $N = 4000$ types, which raised position coverage, the quantity of Appendix E, from 65% to 72% and 80%, respectively. That substitution resolves the tracked types explicitly and collapses everything else into the position-0 and "other" buckets of Appendix D, each of which carries the measured attention mass but a single shared value vector; queries outside the tracked set keep the model's unmodified computation.

Controlling estimation quality. The kernel entry $W_{st} = P_{st}/\bar{n}_{st}$ is estimated from a corpus pass, and its support is the co-occurrence count C_{st}. Enlarging N enlarges the pair set quadratically

while leaving the corpus fixed, so support per pair falls sharply. Matching $N = 1000$'s support at larger N requires roughly $4\times$ the corpus at $N = 2000$ and 15–20$\times$ at $N = 4000$, by three separate criteria (pair coverage, mass coverage, and the worst-supported corner). We therefore repeated every extraction at **40M tokens**, twenty times the original pass and enough to cover the worst case, using the same corpus for all three arms so that corpus size is not itself a variable.

At matched estimation quality, the truncation has little effect. Over steps 0–64 the three vocabularies give ΔL within 0.001 nats of one another (Figure 18). When each arm's ΔL curve is normalized to its own final value, the three agree in shape to within 0.01 across a fourfold change in N. The phase structure detailed in §5 is unchanged.

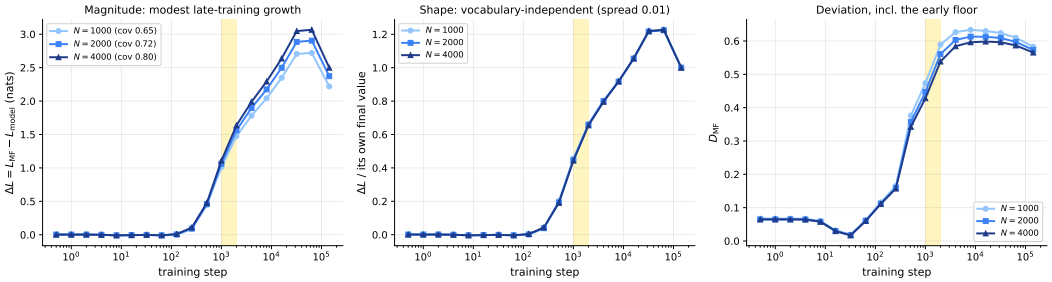

Figure 18: The waypoint experiment at $N = 1000$, 2000 and 4000 keys, all with 40M-token statistics. **Left:** ΔL; the arms separate only late in training, by an amount tracking substitution coverage. **Center:** ΔL normalized to each arm's own final value, whose shape agrees to within 0.01. **Right:** D_{MF}, including the early minimum, which is within 0.003 across the three. The yellow band marks the induction window as in Figure 4.

The full trajectories. Figure 19 redraws the three arms in the layout of Figure 4, We note, first, that L_{model} is the same curve in all three columns, agreeing to 6×10^{-7} nats at every checkpoint: the substitution vocabulary is a property of the null model, not of the network, so the model's own loss is a fixed reference and only L_{MF} moves. This is a check that the three arms differ in nothing but the intended variable. Second, every landmark of the trajectory falls at the same checkpoint irrespective of N. L_{MF} stays within 0.01 nats of L_{model} through step 64 and detaches between steps 128 and 256; it reaches its minimum at step 2000, rises to a late maximum at step 32000, and falls back by step 143000.

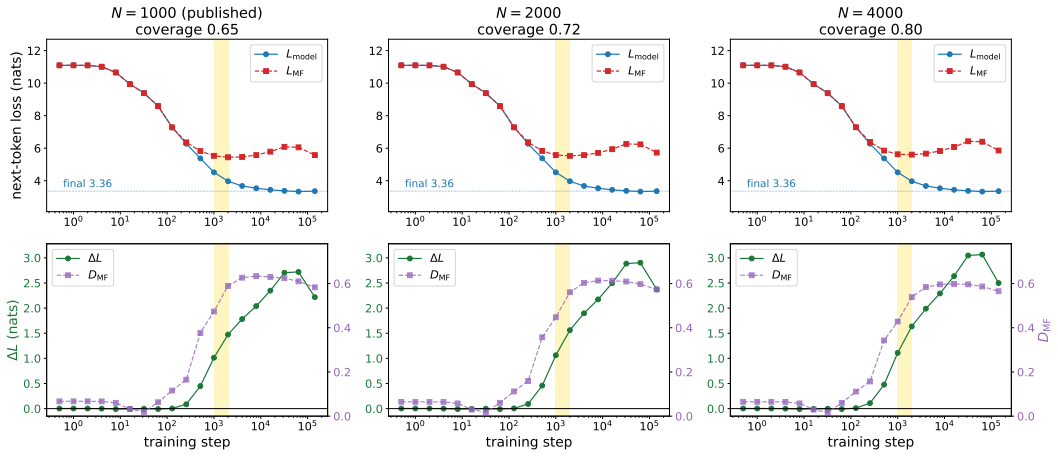

Figure 19: The waypoint experiment at $N = 1000$, 2000 and 4000 keys (columns), 40M-token statistics throughout, drawn in the layout of Figure 4 so the two may be compared panel for panel. **Top:** next-token loss of the model (L_{model}, solid) and of the context-conditional mean field (L_{MF}, dashed); the dotted line marks the model's final loss, and each column is titled with its substitution coverage. **Bottom:** $\Delta L = L_{\text{MF}} - L_{\text{model}}$ (left axis) with the mean per-head deviation D_{MF} (right axis). The band marks the induction window, as in Figure 4. L_{model} is common to the three columns by construction; the shape of the remaining curves is likewise unchanged, and only the magnitude of ΔL grows with coverage.