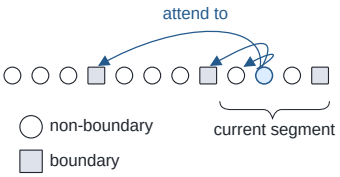
STRUCTURE–RESIDUAL INDUCTIVE BIASES FOR LANGUAGE MODEL PRETRAINING

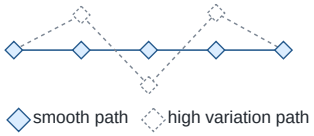
Anonymous authors
Paper under double-blind review

ABSTRACT

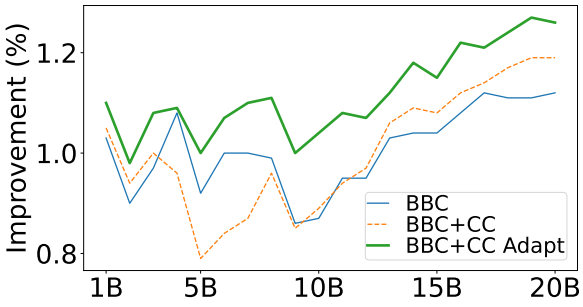
Next-token prediction is relied upon to learn both predictive behavior and internal representations. We ask whether representation structure itself should be an explicit inductive bias. Motivated by the two-part coding view of Kolmogorov complexity, we study a structure–residual principle in which a representation separates structure from realization detail. We instantiate this principle in two complementary ways. Boundary Bottleneck Concept (BBC) routes information from token segments through boundary carriers while preserving local autoregressive computation. Continuous Concept (CC) regularizes representation geometry and induces a spectral decomposition into low-frequency structure and high-frequency residual. We derive coding bounds for both views and show that low CC energy controls the error of multiplicity-corrected BBC-style boundary attention. Empirically, BBC and CC exhibit strikingly different depth preferences: BBC works best in early layers, whereas CC is most effective in later layers. At 300M training tokens, combining BBC and CC improves NLL by 1.44% on GPT-2 Medium. These gains persist with training runs up to 20B tokens. Representation diagnostics show that BBC increases the relative predictive importance of boundary states, while CC reduces spectral residual energy. Together, these results suggest that structure–residual biases can improve language-model pretraining beyond next-token prediction.



(a) Boundary Bottleneck Concept



(b) Continuous Concept



(c) The gains persist through 20B training tokens.

Figure 1: **Structure–residual inductive biases for language model pretraining.** (a) BBC restricts cross-segment communication to boundary carriers while retaining local autoregressive detail. (b) CC encourages smooth positional structure and reduces high-frequency residual variation. (c) The resulting improvements persist over long training horizons.

1 INTRODUCTION

The dominant recipe for improving language models is remarkably simple: scale the model, the data, and the compute, and optimize next-token prediction (NTP) (Kaplan et al., 2020; Hoffmann et al., 2022). Under this paradigm, the training process—primarily gradient-based optimization—is trusted

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.

not only to learn the predictive task, but also to discover how the information required for that task should be internally represented.

We ask a different question: *should representation structure itself be an inductive bias?*

A principled starting point is the two-part coding view of Kolmogorov complexity. An object can be described by first specifying a *structure* Z, and then specifying the *residual* information required to recover the particular realization X:

$$\mathsf{K}(X) \leq \mathsf{K}(Z) + \mathsf{K}(X \mid Z) + O(1).$$

This suggests a general inductive principle for language models: encourage internal representations to discover a favorable balance between structure and residual.

What should such structure look like in a language model? We draw inspiration from two basic functions involved in language modeling.

First, *perception* often involves abstraction. Phrases such as "France's capital" and "the capital of France" differ at the token level but share substantial predictive structure. This suggests compressing information that must be carried across spans into a compact representation. However, a structure-only representation is insufficient: the exact surface realization still matters for autoregressive prediction. Motivated by this structure–residual balance, we introduce *Boundary Bottleneck Concept* (BBC), which routes information from a completed segment to later segments through a boundary carrier while preserving ordinary token-level autoregression within the current segment (Figure 1a). Related approaches include MEGABYTE's combination of global patch modeling and local autoregressive byte decoding (Yu et al., 2023a), and gist tokens that compress prompt information through attention bottlenecks (Mu et al., 2023). BBC instead uses existing segment-end states as carriers.

Second, *generation* requires fine positional information: every token position remains potentially relevant to the next-token distribution. This appears to leave little room for structural compression. Yet positional detail need not imply arbitrary representational complexity. A sequence of hidden states may preserve every token position while still admitting a simple underlying trajectory. Motivated by this observation, we study *Continuous Concept* (CC), which encourages neighboring hidden states to vary smoothly (Figure 1b). We analyze a spectral structure–residual decomposition: low-frequency modes form the structured component, while CC energy bounds the high-frequency residual and its finite-precision index length under a shared coding convention.

On GPT-2 Small, BBC improves evaluation NLL by approximately 0.99% at 300M training tokens, while BBC+CC improves it by 1.30%. On a deeper GPT-2 Medium model, the gains are 1.11% and 1.44%, respectively. BBC is not explained by sparsity alone: same-density sliding-window and random sparse-attention controls do not reproduce its gain, and randomizing the location of BBC carriers significantly degrades performance. Similarly, CC is not explained by either of its two components. Their improvements also persist with longer training, reaching 1.12% for BBC and 1.26% for BBC+CC through our 20B-token runs.

Our empirical study also reveals that the two biases favor different depths. BBC works best near the bottom of the network, while CC is most effective at later layers. Interestingly, the depth asymmetry is consistent with the broader intuition that different depths of a Transformer perform different computational roles: earlier representations can benefit from compact cross-span structure, whereas later representations retain fine token-level access while benefiting from simpler feature geometry.

Our results suggest that scaling NTP is not the only lever for improving language-model training. Explicit biases on *how information is represented* can improve the same predictive objective.

Our contributions are:

- We use the two-part coding principle to analyze structure–residual inductive biases in language-model representations.
- We introduce BBC, which compresses cross-segment communication through boundary carriers, and characterize CC as a spectral structure–residual bias with explicit reconstruction and residual-description bounds.
- We find that BBC and CC favor different depths, outperform matched architectural and regularization controls, and deliver gains that persist across model sizes and long training horizons.

2 THE STRUCTURE–RESIDUAL CODING PRINCIPLE

Two-part coding describes an object through its structure and the detail needed to identify its particular realization (Vereshchagin & Vitanyi, 2004). We write K for prefix Kolmogorov complexity. For finitely described objects,

$$\mathsf{K}(X \mid C) \leq \mathsf{K}(Z \mid C) + \mathsf{K}(X \mid Z, C) + O(1).$$

Here Z describes structure and the conditional term describes the remaining realization detail.

We use this principle to motivate concrete coding bounds. Let Γ denote a coding convention shared by the encoder and decoder. Suppose that, after decoding Z, a residual codebook with $M_\delta(Z)$ entries permits reconstruction of X to accuracy δ. A two-part code then satisfies

$$\ell_\delta^{\mathrm{SR}}(X \mid C, \Gamma) \leq \ell_{\mathrm{str}}(Z \mid C, \Gamma) + \lceil \log_2 M_\delta(Z) \rceil .$$

The structure code is self-delimiting, and the decoded structure and coding convention determine the residual index length. Any codebook or parameter not shared in advance must also be described.

For discrete objects, the residual codebook can enumerate realizations sharing a structure. For continuous representations, a covering codebook gives reconstruction at a chosen precision.

3 BOUNDARY BOTTLENECK CONCEPT

Boundary Bottleneck Concept (BBC) instantiates the structure–residual principle through a granularity hyperparameter K. A sequence is divided into segments of K tokens. At a selected Transformer layer, each completed segment communicates with later segments only through its boundary state, while ordinary autoregressive computation is preserved inside the current segment.

Boundary-limited attention. For notational simplicity, assume that K divides N and index segments by $i = 0, \ldots, N/K - 1$. Empty ranges in the attention mask are interpreted as empty sets.

For fixed segment length K, define $S_i = \big(x_{iK+1}, \ldots, x_{(i+1)K}\big)$. The final position of each segment serves as its *boundary carrier*.

For a query token x_t in segment $s = \left\lfloor \frac{t-1}{K} \right\rfloor$, BBC replaces the ordinary causal attention pattern at the selected layer by

$$\mathcal{A}_{\mathrm{BBC}}(t) = \{K, 2K, \ldots, sK\} \cup \{sK + 1, \ldots, t - 1\}. \tag{1}$$

Once a segment is completed, its K token states are no longer independently accessible to later segments at the BBC layer. The current segment remains autoregressive. Let C_s denote the causal context preceding segment S_s, and define $x_{s,r} = x_{sK+r+1}$. The segment distribution factorizes as $p_\theta(S_s \mid C_s) = \prod_{r=0}^{K-1} p_\theta\left(x_{s,r} \mid C_s, x_{s,<r}\right)$.

All other layers retain the standard causal mask. Since other layers and the residual stream can retain additional prefix information, this layer-local restriction does not imply that the full model conditions only on boundary carriers.

Structure and realization residual. Let $a = \phi(S)$ be a discrete structure assigned to a K-token segment. Its realization class is $\mathcal{S}_K(a) = \{S' \in \mathcal{V}^K : \phi(S') = a\}$, where $\mathcal{V}$ is the vocabulary. The segment distribution induces a structure distribution: $p_\theta(a \mid C_s) = \sum_{S' \in \mathcal{S}_K(a)} p_\theta(S' \mid C_s)$.

For an observed segment S with $\phi(S) = a$ and positive probability,

$$-\log p_\theta(S \mid C_s) = -\log p_\theta(a \mid C_s) - \log p_\theta(S \mid a, C_s). \tag{2}$$

Thus exact segment prediction specifies both the structure and its realization. In particular, $-\log p_\theta(a \mid C_s) \leq -\log p_\theta(S \mid C_s)$. The realization class also gives a coding bound. Assume that ϕ is computable and that the finite vocabulary, segment length, and structure map are supplied to the decoder as Γ_{BBC}. The class can then be enumerated, and a realization identified by its index:

$$\mathsf{K}(S \mid C_s, \Gamma_{\mathrm{BBC}}) \leq \mathsf{K}(a \mid C_s, \Gamma_{\mathrm{BBC}}) + \lceil \log_2 |\mathcal{S}_K(a)| \rceil + O(1).$$

Since $\mathcal{S}_K(a) \subseteq \mathcal{V}^K$, the logarithm of the realization-class size satisfies

$$\log_2 |\mathcal{S}_K(a)| \leq K \log_2 |\mathcal{V}|. \tag{3}$$

Proposition 1 (BBC routing and realization detail)**.** *BBC restricts direct attention access to a completed K-token segment to its boundary carrier at the selected layer, while retaining local autoregressive attention. The segment distribution admits the decomposition in Equation 2. Under the stated coding convention, a nonempty realization class requires at most $\lceil \log_2 |\mathcal{S}_K(a)| \rceil$ index bits, bounded by $\lceil K \log_2 |\mathcal{V}| \rceil$ using Equation 3.*

Proof. The routing statement follows from Equation 1. Because $a = \phi(S)$ is determined by S, $p_\theta(S \mid C_s) = p_\theta(a \mid C_s) p_\theta(S \mid a, C_s)$. Taking negative logarithms gives Equation 2. The counting bound follows from $|\mathcal{S}_K(a)| \le |\mathcal{V}|^K$. □

For fixed sequence length, increasing K reduces the number of boundary carriers and increases the worst-case realization index bound. This counting bound does not specify which information the learned carrier retains.

The probability identities hold for any segment distribution; BBC contributes the architectural routing restriction. Its structure–residual interpretation is therefore a compact cross-segment carrier together with retained local autoregressive computation. The boundary need not equal $\phi(S)$ or be predictively sufficient by construction.

4 Continuous Concept

Continuous Concept (CC) is an optimization bias on the geometry of the hidden-state trajectory. Let $N \ge 2$, and let $H \in \mathbb{R}^{N \times d}$ have row i equal to $h_i^\top$. The states are taken at a selected layer. CC penalizes adjacent variation:

$$\mathcal{L}_{\mathrm{CC}} = \frac{1}{N-1} \sum_{i=1}^{N-1} \|h_{i+1} - h_i\|_2^2.$$

The objective is $\mathcal{L} = \mathcal{L}_{\mathrm{NTP}} + \beta \mathcal{L}_{\mathrm{CC}}$, where $\beta \ge 0$ controls the regularization strength.

We analyze the trajectory as low-frequency structure plus a high-frequency residual. The CC objective controls an upper bound on the residual magnitude.

Spectral structure and residual. Let L be the unit-weight path Laplacian. The unnormalized CC energy is $E(H) = \sum_{i=1}^{N-1} \|h_{i+1} - h_i\|_2^2 = \mathrm{Tr}(H^\top L H) = (N-1)\mathcal{L}_{\mathrm{CC}}$. Write $L = Q \Lambda Q^\top$, $0 = \lambda_1 < \lambda_2 \le \cdots \le \lambda_N$, with orthonormal eigenvectors. These form a graph-frequency basis over token positions (Shuman et al., 2013). This is a vector-valued path instance of the graph-smoothness objective underlying Laplacian Eigenmaps (Belkin & Niyogi, 2003).

For $1 \le m < N$, define $Q_m = [q_1, \ldots, q_m]$, $H_m = Q_m Q_m^\top H$. H_m is the low-frequency structure, and $R_m = H - H_m$ is the spectral residual.

Proposition 2 (Spectral residual bound)**.** *For* $1 \le m < N$, $\|R_m\|_F^2 \le \frac{E(H)}{\lambda_{m+1}}$.

Proof. Let $\widehat{H} = Q^\top H$. Orthogonality gives $\|R_m\|_F^2 = \sum_{r>m} \|\widehat{H}_r\|_2^2$, $E(H) = \sum_r \lambda_r \|\widehat{H}_r\|_2^2$. Therefore $\|R_m\|_F^2 \le \frac{1}{\lambda_{m+1}} \sum_{r>m} \lambda_r \|\widehat{H}_r\|_2^2 \le \frac{E(H)}{\lambda_{m+1}}$. □

For a chosen cutoff, the approximation is accurate when $E(H)/\lambda_{m+1}$ is small.

Residual codebook size. Fix the dimensions, cutoff, and an energy budget $E_\star \ge 0$. For trajectories with $E(H) \le E_\star$, the residual lies in a subspace of dimension $D = (N-m)d$ and in a ball of radius $r_\star = \sqrt{\frac{E_\star}{\lambda_{m+1}}}$. A D-dimensional Euclidean ball of radius $r_\star$ admits a cover by at most

$$\left(1 + \frac{2r_\star}{\eta}\right)^D \tag{4}$$

balls of radius $\eta > 0$ (Vershynin, 2018).

Proposition 3 (Spectral residual index length)**.** *For a shared covering codebook satisfying Equation 4, the bit length of an index identifying a residual approximation with Frobenius error at most η satisfies*

$$L_\eta(R_m \mid \Gamma) \le \left\lceil (N-m)d \log_2 \left(1 + \frac{2}{\eta}\sqrt{\frac{E_\star}{\lambda_{m+1}}}\right)\right\rceil.$$

Proof. Choose a covering center within distance η of the residual and transmit its index; Equation 4 then gives the bound. □

The coding convention Γ includes the dimensions, basis, cutoff, energy budget, precision, and codebook; any unshared components incur additional description cost. Allocating reconstruction error $\delta/2$ to both structure and residual gives $\|H - (\widetilde{H}_m + \widetilde{R}_m)\|_F \le \delta$, and hence

$$\ell_\delta^{\mathrm{SR}}(H \mid \Gamma) \le \ell_{\delta/2}^{\mathrm{str}}(H_m \mid \Gamma) + \left\lceil (N-m)d \log_2 \left(1 + \frac{4r_\star}{\delta}\right)\right\rceil.$$

Thus, at fixed dimensions, cutoff, and precision, a smaller energy budget reduces the residual description bound, while the structure retains its own description cost.

5 From spectral smoothness to boundary representation

The structure–residual coding viewpoint motivates both interventions. We now give a quantitative connection: low within-segment path energy bounds the error of multiplicity-corrected boundary attention. The reverse direction requires additional conditions; see Appendix B.

Let I_j contain the positions of a completed K-token segment, with boundary $b_j = \max I_j$, and define $E_j = \sum_{\substack{t \in I_j \\ t < b_j}} \|h_{t+1} - h_t\|_2^2$.

Lemma 1 (Boundary approximation under low path energy)**.**

$$\max_{i \in I_j} \|h_i - h_{b_j}\|_2 \le \sqrt{(K-1)E_j}.$$

Proof. For $r = b_j - i$, $h_{b_j} - h_i = \sum_{t=i}^{b_j-1}(h_{t+1} - h_t)$. Hence $\|h_{b_j} - h_i\|_2^2 \le r \sum_{t=i}^{b_j-1} \|h_{t+1} - h_t\|_2^2 \le (K-1)E_j$. Equal, aligned increments attain the bound. □

Consider one attention head (Vaswani et al., 2017) with fixed query q, shared linear key and value maps, and no attention dropout: $s_i = \frac{q^\top W_K h_i}{\sqrt{d_k}}$, $v_i = W_V h_i$. Partition all allowed key positions into groups G_j. Completed segments form groups of size K; retained local-prefix positions remain singleton groups. Write $\kappa_j = |G_j|$ and let b_j be the boundary or singleton representative.

Define $g_j = \log \sum_{i \in G_j} e^{s_i}$, $u_j = \sum_{i \in G_j} \frac{e^{s_i}}{\sum_{r \in G_j} e^{s_r}} v_i$. Regrouping the numerator and denominator gives $o = \mathrm{Attn}_{\mathrm{dense}}(q) = \sum_j a_j u_j$, $a = \mathrm{softmax}(g)$. The representative approximation is $\widetilde{g}_j = s_{b_j} + \log \kappa_j$, $\widetilde{u}_j = v_{b_j}$, with $\widetilde{o} = \sum_j \widetilde{a}_j \widetilde{u}_j$ and $\widetilde{a} = \mathrm{softmax}(\widetilde{g})$. As in proportional attention (Bolya et al., 2023), $\log \kappa_j$ preserves the mass of identical entries. Singleton groups are unchanged.

Let $c_K = \frac{\|q\|_2 \|W_K\|_{\mathrm{op}}}{\sqrt{d_k}}$, $c_V = \|W_V\|_{\mathrm{op}}$, we have $|g_j - \widetilde{g}_j| \le c_K\sqrt{(K-1)E_j}$, $\|u_j - \widetilde{u}_j\|_2 \le c_V\sqrt{(K-1)E_j}$ for each completed segments. Indeed, each token score differs from its boundary score by at most $c_K\sqrt{(K-1)E_j}$; bounding the sum of their exponentials gives the first inequality. The second follows because u_j is a convex combination of values individually close to v_{b_j}.

Proposition 4 (Multiplicity-corrected boundary attention)**.** *Let $\Delta_s = \max_j |g_j - \widetilde{g}_j|$, $\Delta_v = \max_j \|u_j - \widetilde{u}_j\|_2$. If $\|\widetilde{u}_j\|_2 \le V_{\max}$ for every group, including local-prefix singletons, then*

$$\|o - \widetilde{o}\|_2 \le \Delta_v + 2V_{\max} \tanh(\Delta_s).$$

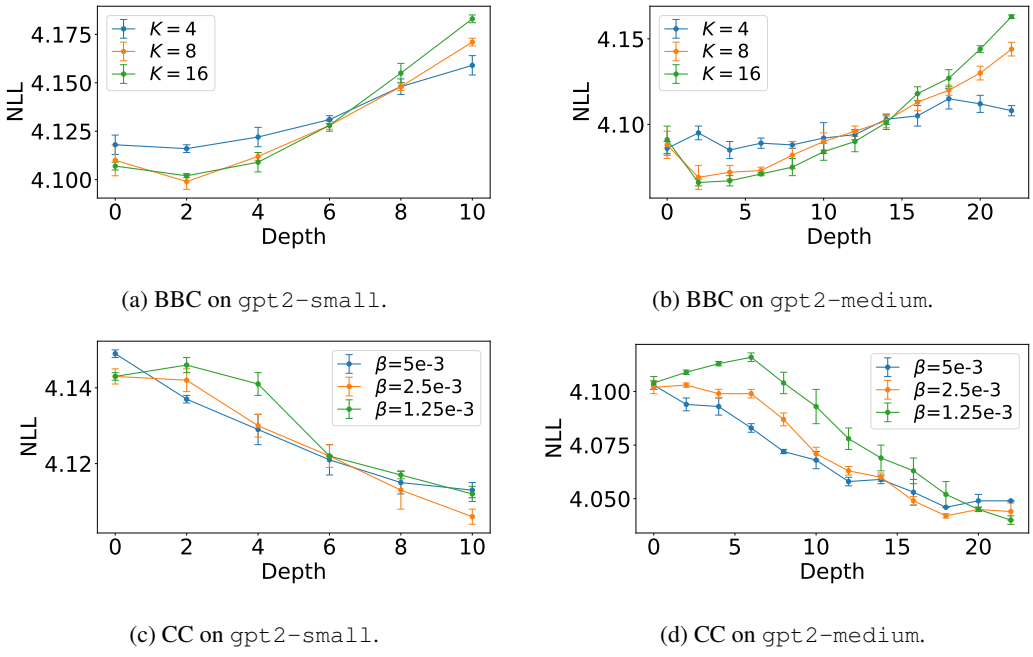
Proof. The score bound implies $e^{-2\Delta_s} \le a_j/\widetilde{a}_j \le e^{2\Delta_s}$. Consequently, $|a_j - \widetilde{a}_j| \le \tanh(\Delta_s)(a_j + \widetilde{a}_j)$, so $\|a - \widetilde{a}\|_1 \le 2\tanh(\Delta_s)$. Therefore $\|o - \widetilde{o}\|_2 \le \sum_j a_j \|u_j - \widetilde{u}_j\|_2 + \sum_j |a_j - \widetilde{a}_j| \, \|\widetilde{u}_j\|_2 \le \Delta_v + 2V_{\max}\tanh(\Delta_s)$. □

For fixed K and uniformly bounded sensitivity and value constants, the error vanishes as the maximum segment energy vanishes. This is a same-layer result for multiplicity-corrected attention, not directly for the uncorrected BBC mask. The bound assumes that h_i enters the stated key and value maps; intervening normalization must be included in the sensitivity constants, and position-dependent maps or logit biases require additional terms.

6 Experiments

Setup. We pre-train `gpt2-small` and `gpt2-medium` models (Radford et al., 2019) on a randomly sampled 1B-token subset of `FineWeb-Edu` (Penedo et al., 2024). We evaluate on a 1M-token holdout set sampled from the same dataset and report holdout NLL. The primary runs use a cosine learning-rate schedule over 1B tokens with a 5M-token linear warmup. To reduce compute, hyperparameter sweeps and ablations are evaluated after 100M or 300M training tokens while retaining the same 1B-token learning-rate decay schedule. Unless otherwise noted, we use three seeds, $\{82, 23, 37\}$, and report mean and standard deviation. For the 20B-token runs, we enlarge the holdout set to 10M tokens and use a single seed (82) to reduce computational cost.

We first tune the vanilla baseline by sweeping the learning rate $\eta \in \{3\times10^{-4}, 6\times10^{-4}, 10^{-3}, 1.5\times10^{-3}\}$. We then fix η and separately sweep the BBC segment length $K \in \{4, 8, 16, 32, 64\}$ and the CC regularization strength $\beta \in \{0.005, 0.0025, 0.00125, 0.000625\}$. We do not retune η for BBC or CC, avoiding method-specific learning-rate tuning. We use $\eta = 10^{-3}$ for `gpt2-small` and $\eta = 6\times10^{-4}$ for `gpt2-medium`. Code: `https://anonymous.4open.science/r/iclr26885`.



(a) BBC on `gpt2-small`.

(b) BBC on `gpt2-medium`.

(c) CC on `gpt2-small`.

(d) CC on `gpt2-medium`.

Figure 2: **BBC and CC exhibit opposite depth preferences.** BBC is most effective at lower layers, while CC becomes more effective toward higher layers, consistently across both model sizes.

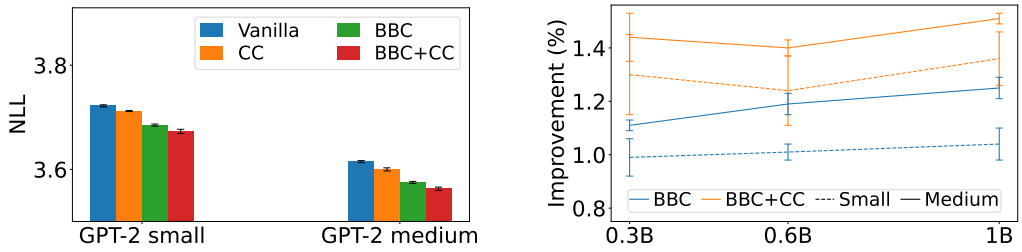
BBC and CC are depth-dependent. Our first finding is a clear depth dependence for both methods. We sweep the intervention layer over the even-numbered Transformer layers for both `gpt2-small` and `gpt2-medium`, using $K \in \{4, 8, 16\}$ for BBC and $\beta \in \{0.00125, 0.0025, 0.005\}$ for CC (Figure 2 and also Table 2, 3 in Appendix A.1). The same qualitative pattern appears across both

model sizes: BBC is most effective when applied at lower layers, whereas CC becomes more effective toward higher layers. This depth asymmetry was not built into either method. Interestingly, it is consistent with the motivating intuition that BBC is perception-inspired while CC is generation-inspired, suggesting that different Transformer depths may favor different representational functions.

In the main BBC/CC comparisons, BBC is applied at layer 2, while CC is applied at layer 10 for `gpt2-small` and layer 22 for `gpt2-medium`.

300M significance tests and 1B persistence. We find that the best K and β selected at 100M tokens retain their gains through 1B training tokens. For `gpt2-small`, we use $K = 8$ and $\beta = 0.0025$. For `gpt2-medium`, we use $K = 16$ and $\beta = 0.00125$.

At 300M tokens, the ordering is consistent across both model sizes: CC significantly outperforms vanilla, BBC significantly outperforms CC, and BBC+CC significantly outperforms BBC (Figure 3a and also Table 4, 5 in Appendix A.2 for p-values).



(a) NLL at 300M training tokens.

(b) BBC and BBC+CC improvement through 1B training tokens.

Figure 3: **Statistically significant gains at 300M tokens persist through 1B tokens.** (a) CC, BBC, and BBC+CC progressively improve over the vanilla baseline at 300M tokens. (b) BBC continues to outperforms vanilla as training proceeds, while BBC+CC maintains an additional margin over BBC.

The improvement persists as training continues. As shown in Figure 3b, from 300M to 1B tokens, BBC shows a steady upward trend in relative improvement over vanilla for both `gpt2-small` and `gpt2-medium`, while BBC+CC continues to maintain a statistically significant margin over BBC.

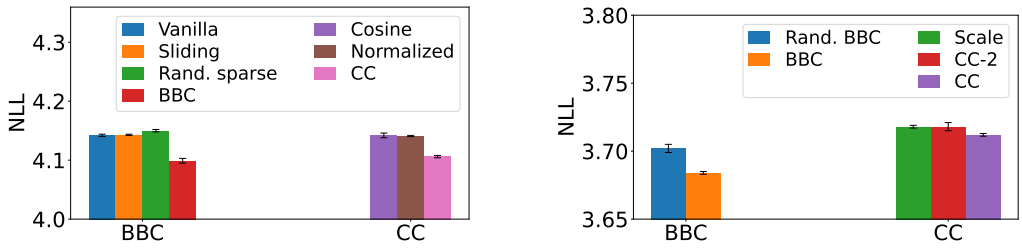
BBC is not merely sparse attention. BBC is a sparse attention pattern. To test whether its gains arise simply from reduced attention density, we compare BBC against same-density sliding-window and random sparse attention, applying all sparse patterns at layer 2. At 100M tokens, BBC significantly outperforms both baselines, and neither of them improves over the vanilla baseline (Figure 4a).

We further compare against *Random BBC*, in which the carrier positions are selected at random while preserving the same per-token attention budget. At 300M tokens, BBC significantly outperforms Random BBC (Figure 4b), indicating that the gain depends on structured carrier placement rather than the attention budget alone. Table 6 in Appendix A.3 reports the paired t-test p-values.

CC requires both scale and roughness. The CC objective has a natural tendency to reduce hidden-state scale (see Appendix C.1 for a formal discussion). To test whether its gains arise primarily from this scaling effect, we compare against a *scale* regularizer, $\mathcal{L}_{\text{scale}} = \frac{1}{N}\sum_{i=1}^{N}\|h_i\|_2^2$, and a *normalized roughness* regularizer, $\mathcal{L}_{\text{norm}} = \frac{1}{N-1}\sum_{i=1}^{N-1}\frac{\|h_{i+1}-h_i\|_2^2}{\|h_{i+1}\|_2^2+\|h_i\|_2^2+\varepsilon}$. The normalized roughness term is complementary to scale in the CC objective.

CC is based on the path-graph Laplacian, whose Dirichlet energy is the weakest penalty in the finite-range pairwise family considered here (see Appendix C.2). We also include CC-2 $\mathcal{L}_{\text{CC-2}} = \frac{1}{N-2}\sum_{i=1}^{N-2}\|2h_{i+2} - h_{i+1} - h_i\|_2^2$, a two-predecessor Laplacian-residual penalty, in the ablation study. CC is also closely related to *cosine distance* (see Appendix C.4), which is included as well.

We tune the application layer and β separately for each variant (Table 7, 8, 9, and 10 in Appendix A.4). As shown in Figure 4, CC significantly outperforms all of these variants. Table 11 in Appendix A.4 reports the corresponding paired t-test p-values.



(a) Ablations at 100M training tokens.

(b) Ablations at 300M training tokens.

Figure 4: **Ablation results for BBC and CC.** At 100M tokens, BBC significantly outperforms sliding-window and random sparse attention, while CC significantly outperforms normalized roughness and cosine-similarity regularization. At 300M tokens, BBC significantly outperforms random BBC, while CC significantly outperforms both scale regularization and CC-2.

20B training runs. We next extend training to 20B tokens. We find that changing the learning-rate decay horizon also changes the optimal CC weight β. Following the same low-cost tuning strategy used in the 1B-token experiments, we select β using a 1B-token run and find that its gain persists through 20B tokens (Figure 1c).

During the 20B runs, however, BBC+CC temporarily underperforms BBC at intermediate checkpoints. We hypothesize that this is partly a numerical-stability issue: because $\beta \ll 1$, the effective contribution of the CC term can become numerically negligible as $\mathcal{L}_{\mathrm{CC}}$ decreases during training.

We therefore study two variants: *random dropout*, which randomly omits the CC term on a fraction of optimization steps, and *adaptive dropout*, which omits the CC term when its value falls below the p-th percentile of the preceding h-step history. The latter is designed to avoid repeatedly applying very small CC updates that may vanish after scaling by β. With adaptive dropout, BBC+CC consistently outperforms BBC at every intermediate checkpoint (see Table 13 and Figure 6 in Appendix A.5).

Representation diagnostics. The preceding experiments establish that BBC and CC improve next-token prediction. We next ask whether the learned representations change in the directions suggested by Sections 3 and 4.

To test whether BBC increases the relative importance of boundary representations, we either replace the boundary representation and measure the resulting NLL increase, Δ_b, or replace a randomly selected non-boundary representation and measure the corresponding increase, Δ_i. A larger $\Delta_b - \Delta_i$ indicates that boundary representations are relatively more consequential for prediction. With both zero-vector replacement and replacement by the mean of the preceding $K-1$ hidden states, BBC yields a significantly larger $\Delta_b - \Delta_i$ than Vanilla (Table 1). In both experiments, we exclude the first $W = 4$ segments to control for the disproportionate influence of initial-token positions (see Appendix A.6 for full diagnostic).

Table 1: Boundary-preference statistic $\Delta_b - \Delta_i$ at 100M training tokens under zero-vector and mean-replacement interventions. The last row reports the paired BBC–Vanilla p-value.

Model	Zero replacement	Mean replacement
Vanilla	-0.0025	0.0017
BBC	0.0052	0.0065
p**-value**	3.46×10^{-5}	4.86×10^{-4}

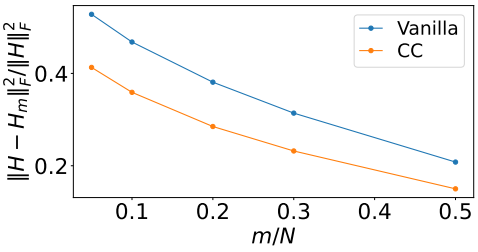


Figure 5: **CC increases spectral compressibility.** Normalized spectral residual as a function of the retained low-frequency fraction m/N.

Next we test whether reducing CC energy makes the hidden-state trajectory better approximated by its low-frequency path-graph modes. For each hidden-state trajectory H, we project onto the first m path-graph eigenmodes, $H_m = Q_m Q_m^\top H$, and measure the normalized spectral residual $\rho_m(H) = \frac{\|H - H_m\|_F^2}{\|H\|_F^2}$. This ratio is invariant to uniform rescaling and measures the residual relative to total hidden-state energy. It includes the constant positional component; it does not isolate roughness of the centered trajectory. CC reduces $\rho_m(H)$ at every measured spectral cutoff. Across the measured range, this corresponds to a relative reduction of approximately 22%–28% (Figure 5 and Table 14 in Appendix A.7).

7 Related Work

Representation learning for LLMs. Representation learning has been central to self-supervised learning in vision, where joint-embedding predictive architectures (JEPA) (Assran et al., 2023) learn semantic representations through embedding-space prediction. Recent work brings related ideas to language: LLM-JEPA (Huang et al., 2026a) incorporates JEPA-style objectives into language-model training, while Semantic Tube Prediction (STP) (Huang et al., 2026b) introduces a geometric prior on hidden-state trajectories. Large Concept Models (LCM Team et al., 2024) perform autoregressive prediction in a sentence-level embedding space. Our work shares the general motivation and introduces principled structure–residual inductive biases.

Structured communication. Fixed sparse attention (Child et al., 2019) combines local attention with access to positions near previous block boundaries. BBC can be viewed as a single-carrier ($c = 1$) specialization—a setting that Child et al. (2019) found overly restrictive—but we show that applying it at the appropriate depth improves predictive performance. Block Transformer (Ho et al., 2024) places global block processing below local token decoding. Activation Beacon (Zhang et al., 2025) compresses context into inserted beacon tokens' key–value states. H-Net (Hwang et al., 2026) learns boundaries, processes selected states at reduced resolution, and upsamples for decoding. Local, random, and global connections are standard ingredients of sparse attention designs such as Big Bird (Zaheer et al., 2020).

Smoothness and spectral structure. AWD-LSTM (Merity et al., 2018) penalizes successive recurrent-state changes through temporal activation regularization. Li et al. (2026) encourage angular dispersion across batchwise hidden states. Shuman et al. (2013) express graph smoothness through Laplacian frequencies. CC regularizes adjacent Transformer hidden states and bounds low-frequency reconstruction error and finite-precision residual code length, with ablations separating scale from positional roughness.

Two-part coding. CRATE (Yu et al., 2023b) alternates token compression and feature sparsification; ToST (Wu et al., 2025) derives attention by unrolling variational coding-rate reduction. Delétang et al. (2024) turn predictive probabilities into lossless data codes. Minimum description length (Rissanen, 1978) charges for the model and data encoded under it, while algorithmic statistics (Gács et al., 2001) formalizes structure and realization through two-part codes. We adopt this two-part accounting for BBC's segment realizations and CC's spectral representations, retaining both structure and residual costs without directly optimizing Kolmogorov complexity.

Depth specialization. Skean et al. (2025) evaluate layerwise geometry and information content for embedding quality. Lad et al. (2025) probe computational sensitivity by deleting and swapping layers. Our training-time layer sweeps identify early BBC and late CC as favorable placements in two GPT-2 model sizes; these preferences are observed empirically, not prescribed by the methods.

8 Conclusion

Motivated by the two-part coding view of Kolmogorov complexity, we introduced BBC and CC as complementary structure–residual inductive biases for language-model pretraining and derived explicit bounds characterizing both mechanisms. BBC compresses cross-segment communication through boundary carriers, while CC encourages compact spectral structure in hidden-state trajectories. We further establish a quantitative connection between the two and find that BBC favors early layers, whereas CC favors later layers. Their gains persist across model sizes and training horizons up to 20B tokens, demonstrating that explicit representation biases can improve next-token pretraining.

AI USE STATEMENT

AI tools were used solely for language polishing and improving the clarity of the manuscript. All AI-assisted writing was reviewed and approved by the authors, who take full responsibility for the submitted content.

ETHICS STATEMENT

This work studies architectural and representation-level inductive biases for language-model pretraining using publicly available datasets. It does not involve human subjects or the collection of personal information. While the proposed methods may improve the efficiency and performance of language models, they do not directly address potential downstream risks, including harmful content, bias, or misuse. These considerations remain important when deploying models trained using the proposed methods.

REPRODUCIBILITY STATEMENT

We provide an anonymized implementation at `https://anonymous.4open.science/`, including the training and evaluation code, hyperparameter configurations, and dataset selection scripts. The implementation has been extensively reviewed and tested by the co-authors, and all theoretical proofs have been independently reviewed by multiple co-authors.

Our training data are randomly sampled from FineWeb-Edu, with the sampling procedure included in the released code. All experiments, except the 20B-token runs, are conducted with three random seeds, and we report the mean and standard deviation. For claims of statistical significance, we report paired t-test p-values. The 20B-token experiments use a single seed due to computational cost; we report their results without statistical significance claims.

Due to their size, the training checkpoints and sampled datasets will be released after the review process.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, 2023. doi: 10.1109/CVPR52729.2023.01499. URL https://arxiv.org/abs/2301.08243.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations*, 2023.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019. URL https://arxiv.org/abs/1904.10509.

Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. Language modeling is compression. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/3cbf627fa24fb6cb576e04e689b9428b-Abstract-Conference.html.

Péter Gács, John Tromp, and Paul M. B. Vitányi. Algorithmic statistics. *IEEE Transactions on Information Theory*, 47(6):2443–2463, 2001. doi: 10.1109/18.945257. URL https://arxiv.org/abs/math/0006233.

Namgyu Ho, Sangmin Bae, Taehyeon Kim, Hyunjik Jo, Yireun Kim, Tal Schuster, Adam Fisch, James Thorne, and Se-Young Yun. Block transformer: Global-to-local language modeling for fast inference. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/579b5b84311e122584dedab1d3b7613f-Abstract-Conference.html.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In *NeurIPS*, volume 35, 2022.

Hai Huang, Yann LeCun, and Randall Balestriero. LLM-JEPA: Large language models meet joint embedding predictive architectures. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=GbXKPo9QfH.

Hai Huang, Yann LeCun, and Randall Balestriero. Semantic tube prediction: Beating LLM data efficiency with JEPA. In *ICML*, 2026b.

Sukjun Hwang, Brandon Wang, and Albert Gu. Dynamic chunking for end-to-end hierarchical sequence modeling. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ZbfLR9NbNF.

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

Vedang Lad, Jin Hwa Lee, Wes Gurnee, and Max Tegmark. Remarkable robustness of LLMs: Stages of inference? In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4332. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/bcad07d4bfab51243efaa08b8ed475b3-Abstract-Conference.html.

LCM Team, Loïc Barrault, Paul-Ambroise Duquenne, Maha Elbayad, Artyom Kozhevnikov, Belen Alastruey, Pierre Andrews, Mariano Coria, Guillaume Couairon, Marta R. Costa-jussà, David Dale, Hady Elsahar, Kevin Heffernan, João Maria Janeiro, Tuan Tran, Christophe Ropers, Eduardo Sánchez, Robin San Roman, Alexandre Mourachko, Safiyyah Saleem, and Holger Schwenk. Large concept models: Language modeling in a sentence representation space. *arXiv preprint arXiv:2412.08821*, 2024.

Yanhong Li, Ming Li, Karen Livescu, and Jiawei Zhou. On the predictive power of representation dispersion in language models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/5ef998a50e6a4f9ff52d39833ded30a0-Abstract-Conference.html.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SyyGPP0TZ.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *NeurIPS*, volume 36, 2023.

Guilherme Penedo, Hynek Kydlíček, 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 *NeurIPS*, volume 37, 2024.

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

Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978. doi: 10.1016/0005-1098(78)90005-5.

David I. Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Nikul Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55854–55875. PMLR, 2025. URL https://proceedings.mlr.press/v267/skean25a.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, 2017.

Nikolai Vereshchagin and Paul Vitanyi. Kolmogorov's structure functions and model selection. *IEEE Transactions on Information Theory*, 50(12):3265–3290, 2004.

Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge University Press, 2018.

Ziyang Wu, Tianjiao Ding, Yifu Lu, Druv Pai, Jingyuan Zhang, Weida Wang, Yaodong Yu, Yi Ma, and Benjamin Haeffele. Token statistics transformer: Linear-time attention via variational rate reduction. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5fcd540792da599adf1b932624e98f1f-Abstract-Conference.html.

Lili Yu, Dániel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. MEGABYTE: Predicting million-byte sequences with multiscale transformers. In *NeurIPS*, volume 36, 2023a.

Yaodong Yu, Sam Buchanan, Druv Pai, Tianzhe Chu, Ziyang Wu, Shengbang Tong, Benjamin D. Haeffele, and Yi Ma. White-box transformers via sparse rate reduction. In *NeurIPS*, volume 36, 2023b.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In *NeurIPS*, volume 33, 2020.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. Long context compression with activation beacon. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/fc797c61eb18f7b3ce9a74af0ef0d876-Abstract-Conference.html.

A Additional Experimental Results

A.1 BBC and CC are depth-dependent

Table 2: Applying BBC at lower layers yields better NLL than at higher layers.

Layer	$K = 4$	$K = 8$	$K = 16$
0	4.118±0.005	4.110±0.008	4.107±0.002
2	**4.116±0.002**	**4.099±0.004**	**4.102±0.001**
4	4.122±0.005	4.112±0.002	4.109±0.005
6	4.131±0.002	4.128±0.002	4.128±0.003
8	4.148±0.004	4.148±0.002	4.155±0.005
10	4.159±0.005	4.171±0.002	4.183±0.002

(a) `gpt2-small` at 100M tokens

Layer	$K = 4$	$K = 8$	$K = 16$
0	4.086±0.004	4.088±0.008	4.091±0.008
2	4.095±0.004	**4.069±0.007**	**4.066±0.002**
4	**4.085±0.005**	4.072±0.004	4.067±0.003
6	4.089±0.003	4.073±0.002	4.071±0.001
8	4.088±0.002	4.082±0.008	4.075±0.005
10	4.092±0.009	4.090±0.005	4.084±0.005
12	4.094±0.003	4.096±0.003	4.090±0.006
14	4.103±0.003	4.102±0.004	4.101±0.004
16	4.105±0.006	4.113±0.005	4.118±0.004
18	4.115±0.006	4.120±0.003	4.127±0.005
20	4.112±0.005	4.130±0.004	4.144±0.002
22	4.108±0.003	4.144±0.004	4.163±0.001

(b) `gpt2-medium` at 100M tokens

Unlike BBC, which is applied within a Transformer layer, CC regularizes the hidden state between layers. To avoid introducing a layer index of -1, we index CC by the layer whose input it regularizes. Under this convention, the output of the final layer can be viewed as the input to a phantom layer $N+1$. We therefore sweep layers $0, 2, \ldots, 12$ for `gpt2-small` and $0, 2, \ldots, 24$ for `gpt2-medium` in Table 3.

Table 3: Applying $\mathcal{L}_{\mathrm{CC}}$ at higher layers yields better NLL than at lower layers.

Layer	β **= 0.005**	β **= 0.0025**	β **= 0.00125**
12	4.139±0.003	4.146±0.002	4.144±0.007
10	**4.113±0.002**	**4.106±0.002**	**4.112±0.002**
8	4.115±0.003	4.113±0.005	4.117±0.001
6	4.121±0.004	4.122±0.003	4.122±0.000
4	4.129±0.004	4.130±0.003	4.141±0.003
2	4.137±0.001	4.142±0.003	4.146±0.002
0	4.149±0.001	4.143±0.002	4.143±0.001

(a) `gpt2-small` at 100M tokens

Layer	β **= 0.005**	β **= 0.0025**	β **= 0.00125**
24	4.104±0.008	4.107±0.001	4.105±0.002
22	4.049±0.000	4.044±0.004	**4.040±0.002**
20	4.049±0.003	4.045±0.001	4.045±0.001
18	**4.046±0.000**	**4.042±0.001**	4.052±0.006
16	4.053±0.006	4.049±0.002	4.063±0.006
14	4.059±0.002	4.060±0.002	4.069±0.006
12	4.058±0.002	4.063±0.002	4.078±0.005
10	4.068±0.004	4.071±0.003	4.093±0.008
8	4.072±0.001	4.087±0.003	4.104±0.005
6	4.083±0.002	4.099±0.002	4.116±0.002
4	4.093±0.004	4.099±0.002	4.113±0.001
2	4.094±0.003	4.103±0.001	4.109±0.001
0	4.103±0.002	4.102±0.003	4.104±0.003

(b) `gpt2-medium` at 100M tokens

A.2 300M Significance Tests and 1B Persistence

Table 4: Results on `gpt2-small`. At 300M training tokens, CC, BBC, and BBC+CC all significantly outperform the vanilla baseline. The gains from BBC and BBC+CC persist through 1B training tokens.

Config	NLL	Improve (%)	vanilla	CC	BBC
vanilla	3.722±0.002				
CC	3.712±0.001	0.27	0.00228		
BBC	3.685±0.002	0.99	0.000925	0.00109	
BBC+CC	3.673±0.004	1.30	0.00231	0.00248	0.0364

(a) Results on `gpt2-small` at 300M tokens. The right-hand side of the table reports the p-values from paired t-tests.

Config	300M	600M	1B
BBC	0.99±0.07	1.01±0.03	1.04±0.06
BBC+CC	1.30±0.15	1.24±0.13	1.36±0.10

(b) Results on `gpt2-small` through 1B tokens, reported as improvement (%) over vanilla.

Table 5: Results on `gpt2-medium`. At 300M training tokens, CC, BBC, and BBC+CC all significantly outperform the vanilla baseline. The gains from BBC and BBC+CC persist through 1B training tokens.

Config	NLL	Improve (%)	vanilla	CC	BBC
vanilla	3.615±0.002				
CC	3.600±0.003	0.42	0.00286		
BBC	3.575±0.002	1.11	0.0000675	0.000383	
BBC+CC	3.563±0.003	1.44	0.000621	0.000466	0.00793

(a) Results on `gpt2-medium` at 300M tokens. The right-hand side of the table reports the p-values from paired t-tests.

Config	300M	600M	1B
BBC	1.11±0.02	1.19±0.04	1.25±0.04
BBC+CC	1.44±0.09	1.40±0.03	1.51±0.02

(b) Results on `gpt2-medium` through 1B tokens, reported as improvement (%) over vanilla.

A.3 BBC Ablations

Table 6: Results on `gpt2-small`. At 100M training tokens, sliding-window and random sparse attention both underperform the vanilla baseline, whereas BBC significantly outperforms it. At 300M tokens, BBC also significantly outperforms a variant in which the carrier positions are selected at random.

Config	NLL	Improve (%)	vanilla	sliding	rand sparse
vanilla	4.142±0.002				
sliding	4.143±0.001	−0.02	0.243		
rand sparse	4.150±0.002	−0.19	0.0372	0.0170	
BBC	4.099±0.004	1.00	0.000313	0.00104	0.00220

(a) BBC vs. sliding-window attention and random sparse attention at 100M tokens. The right-hand side of the table reports the p-values from paired t-tests.

Config	NLL	p-value
BBC	3.684±0.001	
rand BBC	3.702±0.003	0.0112

(b) BBC vs. random BBC at 300M tokens. The right-hand side of the table reports the p-values from paired t-tests.

A.4 CC Ablations

We tune the application layer and regularization strength β separately for each CC variant—$\mathcal{L}_{\text{scale}}$, $\mathcal{L}_{\text{norm}}$, $1-\cos\theta$ as defined in Appendix C.4, and $\mathcal{L}_{\text{CC-2}}$. Because these objectives operate at different numerical scales and can exhibit different depth preferences, each requires its own matched choice of β and application layer.

Scale regularization. As shown in Table 7, the best configuration applies the regularizer at layer 10 with $\beta = 0.0025$.

Table 7: $\mathcal{L}_{\text{scale}}$ regularization on `gpt2-small` at 100M training tokens. Results are mean NLL $\pm$ standard deviation over three seeds.

Layer	$\beta = 0.00125$	$\beta = 0.0025$	$\beta = 0.005$
0	4.146 ± 0.005	4.143 ± 0.000	4.145 ± 0.005
2	4.137 ± 0.000	4.144 ± 0.004	4.147 ± 0.005
4	4.126 ± 0.005	4.132 ± 0.006	4.132 ± 0.008
6	4.119 ± 0.003	4.124 ± 0.008	4.125 ± 0.005
8	4.120 ± 0.002	4.119 ± 0.003	4.120 ± 0.001
10	$\mathbf{4.119 \pm 0.005}$	$\mathbf{4.111 \pm 0.003}$	$\mathbf{4.114 \pm 0.004}$
12	4.146 ± 0.006	4.144 ± 0.002	4.144 ± 0.002

Normalized roughness regularization. As shown in Table 8, the best configuration applies the regularizer at layer 2 with $\beta = 0.000625$.

Table 8: $\mathcal{L}_{\text{norm}}$ regularization on `gpt2-small` at 100M training tokens. Results are mean NLL $\pm$ standard deviation over three seeds.

Layer	$\beta = 0.0003125$	$\beta = 0.000625$	$\beta = 0.00125$	$\beta = 0.0025$	$\beta = 0.005$	$\beta = 0.01$	$\beta = 0.02$
0	4.151 ± 0.005	4.145 ± 0.004	$\mathbf{4.143 \pm 0.007}$	4.144 ± 0.002	4.146 ± 0.003	4.142 ± 0.002	4.144 ± 0.007
2	4.146 ± 0.002	$\mathbf{4.141 \pm 0.001}$	4.149 ± 0.003	4.146 ± 0.007	4.144 ± 0.002	4.150 ± 0.005	4.160 ± 0.006
4	4.148 ± 0.003	4.147 ± 0.002	4.148 ± 0.002	4.151 ± 0.005	4.148 ± 0.004	4.145 ± 0.006	4.150 ± 0.003
6	4.147 ± 0.003	4.149 ± 0.002	4.147 ± 0.002	4.147 ± 0.001	4.152 ± 0.003	4.146 ± 0.005	4.150 ± 0.002
8	4.146 ± 0.006	4.146 ± 0.002	4.146 ± 0.005	4.145 ± 0.004	4.149 ± 0.001	4.146 ± 0.004	4.147 ± 0.004
10	4.146 ± 0.002	4.144 ± 0.008	4.146 ± 0.001	4.147 ± 0.007	4.147 ± 0.004	4.148 ± 0.005	4.143 ± 0.002
12	$\mathbf{4.144 \pm 0.008}$	4.144 ± 0.004	4.144 ± 0.005	$\mathbf{4.143 \pm 0.002}$	$\mathbf{4.142 \pm 0.001}$	$\mathbf{4.142 \pm 0.001}$	$\mathbf{4.143 \pm 0.004}$

Cosine distance regularization. As shown in Table 9, the best configuration applies the regularizer at layer 8 with $\beta = 0.0025$.

Table 9: $1 - \cos\theta$ regularization on `gpt2-small` at 100M training tokens. Results are mean NLL $\pm$ standard deviation over three seeds.

Layer	$\beta = 0.00125$	$\beta = 0.0025$	$\beta = 0.005$
0	4.145 ± 0.001	4.148 ± 0.006	4.144 ± 0.004
2	4.147 ± 0.006	4.145 ± 0.010	4.145 ± 0.002
4	4.144 ± 0.001	4.146 ± 0.003	4.146 ± 0.006
6	4.144 ± 0.002	4.145 ± 0.001	4.153 ± 0.002
8	4.146 ± 0.003	$\mathbf{4.142 \pm 0.004}$	4.145 ± 0.004
10	$\mathbf{4.142 \pm 0.001}$	4.144 ± 0.006	4.146 ± 0.007
12	4.146 ± 0.004	4.144 ± 0.002	$\mathbf{4.143 \pm 0.002}$

CC-2 regularization. As shown in Table 10, the best configuration applies the regularizer at layer 10 with $\beta = 0.0025$.

Table 10: $\mathcal{L}_{\text{CC-2}}$ regularization on `gpt2-small` at 100M training tokens. Results are mean NLL $\pm$ standard deviation over three seeds.

Layer	$\beta = 0.00125$	$\beta = 0.0025$	$\beta = 0.005$
0	4.145 ± 0.002	4.145 ± 0.003	4.146 ± 0.001
2	4.147 ± 0.003	4.156 ± 0.007	4.178 ± 0.004
4	4.135 ± 0.005	4.148 ± 0.002	4.179 ± 0.006
6	4.122 ± 0.002	4.131 ± 0.007	4.160 ± 0.008
8	4.117 ± 0.005	4.121 ± 0.003	4.135 ± 0.005
10	$\mathbf{4.110 \pm 0.001}$	$\mathbf{4.110 \pm 0.010}$	$\mathbf{4.120 \pm 0.003}$
12	4.148 ± 0.003	4.145 ± 0.002	4.144 ± 0.007

Significance tests. At 100M training tokens, CC significantly outperforms both cosine-distance and normalized-roughness regularization. For scale regularization and CC-2, the performance gap becomes statistically significant only after extending training to 300M tokens.

Table 11: Results on `gpt2-small`. At 100M training tokens, CC significantly outperforms both normalized roughness and cosine similarity regularization. At 300M tokens, CC also significantly outperforms scale regularization and CC-2.

Config	NLL	Improve (%)	vanilla	cosine	normalized
vanilla	4.142±0.002				
cosine	4.142±0.004	0.01	0.466		
normalized	4.141±0.001	0.02	0.288	0.412	
CC	4.106±0.002	0.87	0.00127	0.00154	0.0000978

(a) CC vs. cosine similarity and normalized roughness regularization at 100M tokens. The right-hand side of the table reports the p-values from paired t-tests.

Config	NLL	scale	CC-2
scale	3.718±0.001		
CC-2	3.718±0.003	0.477	
CC	3.712±0.001	0.0113	0.0386

(b) CC vs. scale regularization and CC-2 at 300M tokens. The right-hand side of the table reports the p-values from paired t-tests.

A.5 20B Training Runs

When extending training to 20B tokens, we first retune the CC weight β using a 1B-token sweep. Under the longer schedule, the slower learning-rate decay allows CC to exert greater cumulative downward pressure on $\mathcal{L}_{\mathrm{CC}}$, which can shift the optimal β. Using a single seed (82), we find that $\beta = 0.000625$ performs best at 1B tokens. With this setting, BBC+CC eventually outperforms BBC after 10B tokens, but underperforms it at intermediate checkpoints between 4B and 9B tokens. Further decreasing β does not improve the 1B-token result.

We hypothesize that this behavior is partly due to numerical stability: as $\mathcal{L}_{\mathrm{CC}}$ decreases, multiplying it by $\beta \ll 1$ can make the corresponding optimization signal numerically negligible. We therefore study two variants. *Random dropout* skips the CC term independently with dropout rate $d > 0$. *Adaptive dropout* instead skips the CC term whenever $\mathcal{L}_{\mathrm{CC}}$ falls below the p-th percentile of its values over the preceding h optimization steps, reducing the frequency of very small CC updates.

At 1B training tokens, we evaluate random dropout with $d = 0.5$ and adaptive dropout with $p = 50$ and $h = 32$ using three seeds $\{82, 23, 37\}$ (Table 12). Both variants substantially reduce the across-seed variance relative to standard CC, and adaptive dropout achieves the best mean performance of the three. We therefore use adaptive dropout for a second 20B-token run, where BBC+CC Adapt outperforms BBC at every intermediate checkpoint.

We evaluate every 1B-token intermediate checkpoint on an enlarged 10M-token holdout set to reduce evaluation noise. This gives the high precision required for comparing CC variants (Table 12) and offsets the absence of seed averaging in the single-seed 20B-token runs (Table 13). In addition, we evaluate on the 1M-token holdout set every 10M training tokens, which produces Figure 6.

Table 12: BBC+CC variants for `gpt2-medium` at 1B tokens with 20B learning-rate decay horizon and $\beta = 0.000625$. For *Random Dropout* the probability is 0.5. For *Adaptive Dropout* the p-th percentile is 50 and history size is 32. The evaluation is run on 10M-token holdout set.

BBC+CC Variant	**NLL**
Standard CC	3.2757±0.0043
CC Random Dropout $d = 0.5$	3.2757±0.0008
CC Adaptive Dropout $p = 50,\ h = 32$	3.2753±0.0005

Table 13: Results on `gpt2-medium` at 20B tokens. BBC and both BBC+CC variants significant outperform the vanilla baseline. BBC+CC lags behind BBC at some initial checkpoints while BBC+CC Adapt consistently outperforms it. Each checkpoint is evaluated on 10M-token holdout set.

	NLL				Improv (%)		
Tokens	vanilla	BBC	BBC+CC	BBC+CC Adapt	BBC	BBC+CC	BBC+CC Adapt
1B	3,3113	3,2772	3,2765	3,2748	1,03	1,05	1,10
2B	3,2107	3,1819	3,1807	3,1793	0,90	0,94	0,98
3B	3,1704	3,1397	3,1387	3,1363	0,97	1,00	1,08
4B	3,1426	3,1089	3,1125	3,1085	1,08	0,96	1,09
5B	3,1184	3,0897	3,0937	3,0871	0,92	0,79	1,00
6B	3,0980	3,0671	3,0720	3,0649	1,00	0,84	1,07
7B	3,0801	3,0494	3,0533	3,0461	1,00	0,87	1,10
8B	3,0609	3,0304	3,0315	3,0268	0,99	0,96	1,11
9B	3,0376	3,0116	3,0118	3,0072	0,86	0,85	1,00
10B	3,0185	2,9922	2,9916	2,9872	0,87	0,89	1,04
11B	2,9983	2,9700	2,9700	2,9660	0,95	0,94	1,08
12B	2,9757	2,9475	2,9467	2,9438	0,95	0,97	1,07
13B	2,9526	2,9221	2,9214	2,9194	1,03	1,06	1,12
14B	2,9290	2,8984	2,8971	2,8944	1,04	1,09	1,18
15B	2,9036	2,8733	2,8722	2,8703	1,04	1,08	1,15
16B	2,8784	2,8473	2,8462	2,8433	1,08	1,12	1,22
17B	2,8569	2,8249	2,8245	2,8223	1,12	1,14	1,21
18B	2,8379	2,8063	2,8048	2,8028	1,11	1,17	1,24
19B	2,8244	2,7930	2,7908	2,7886	1,11	1,19	1,27
20B	2,8193	2,7876	2,7858	2,7837	1,12	1,19	1,26

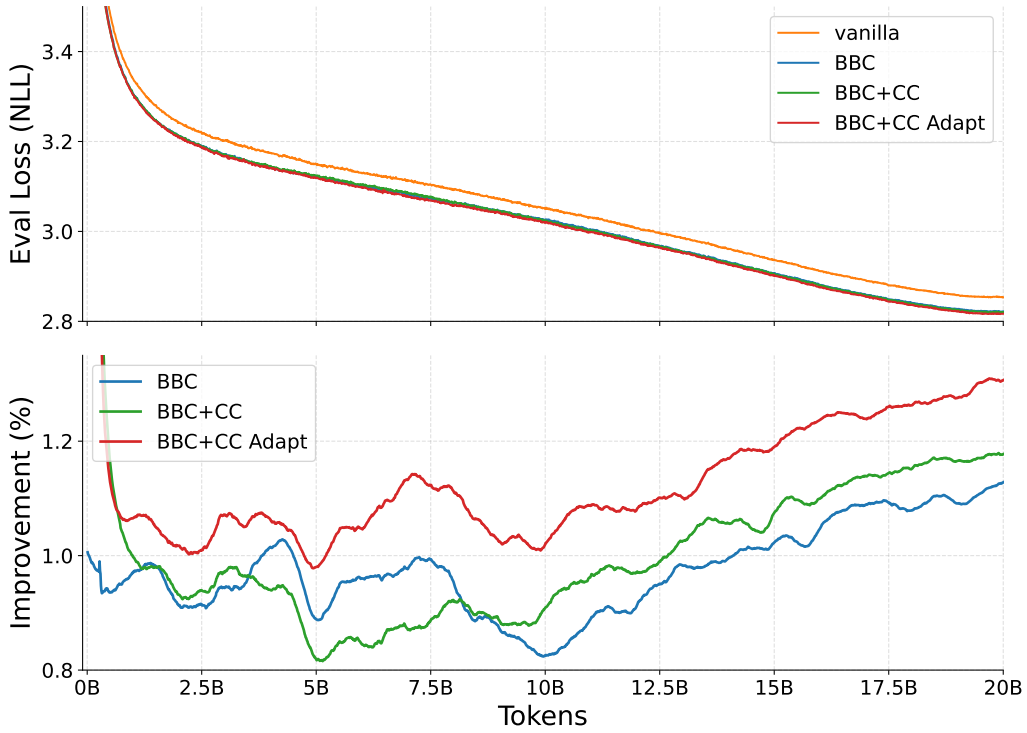


Figure 6: **Metrics for 20B runs.** Evaluation NLL loss and Improvement (%) over vanilla during 20B `gpt2-medium` training runs. The evaluation is run on 1M-token holdout set every 10M training tokens.

A.6 Boundary Intervention Diagnostic

To test whether BBC increases the relative importance of boundary representations, we evaluate frozen 100M-token Vanilla and BBC checkpoints using two matched activation interventions. For each completed segment, we either replace the boundary representation by a zero vector and measure the resulting increase in NLL, denoted Δ_b, or replace a randomly selected non-boundary representation in the same segment and measure the corresponding increase, denoted Δ_i. We use

$$G = \Delta_b - \Delta_i \tag{5}$$

as the boundary-preference statistic. A positive G indicates that ablating the boundary is more damaging than ablating a matched interior position. More importantly,

$$G_{\text{BBC}} > G_{\text{Vanilla}} \tag{6}$$

indicates that BBC increases the relative predictive importance of boundary representations.

Controlling for early-position effects. A direct intervention over the entire sequence is confounded by the unusually large influence of the first few tokens. Figure 7 shows the boundary-preference statistic as the ablated interior position is moved relative to the segment boundary. With $K = 8$, position -7 corresponds to the first token of a segment. Without excluding the initial segments, ablating this position produces a disproportionately large change in NLL, dominating the boundary-versus-interior comparison.

We therefore exclude the first $W = 4$ segments from both the intervention statistics and the corresponding NLL evaluation. After this warmup exclusion, the dependence on the particular non-boundary position becomes negligible (Figure 7), allowing the diagnostic to isolate the relative effect of boundary placement rather than the special role of the earliest tokens.

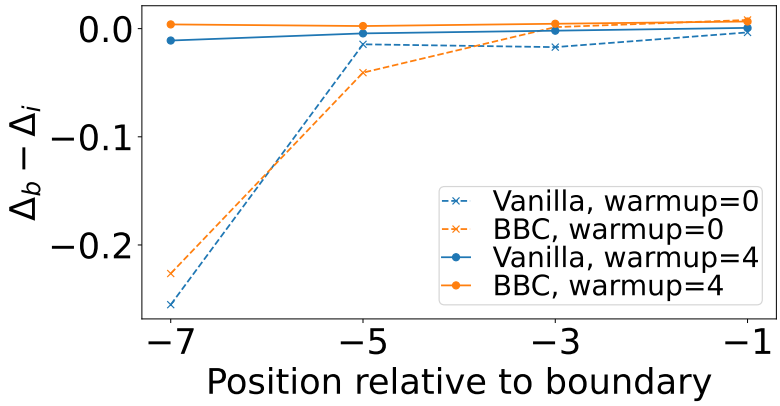


Figure 7: **Effect of the ablated interior position on the boundary-preference statistic $\Delta_b - \Delta_i$.** Without excluding the first segments ($W = 0$), ablating the earliest position produces a large positional artifact. Excluding the first four segments ($W = 4$) largely removes this dependence.

Zero-vector intervention. With $W = 4$, we obtain

Model	Δ_b	Δ_i	$\Delta_b - \Delta_i$
Vanilla	1.2293 ± 0.1566	1.2318 ± 0.1556	-0.0025 ± 0.0618
BBC	0.8553 ± 0.1028	0.8500 ± 0.1026	0.0052 ± 0.0589

Vanilla exhibits essentially no positive boundary preference: $\Delta_b - \Delta_i = -0.0025$. In contrast, BBC yields a positive boundary-preference gap of 0.0052. More importantly, the paired BBC–Vanilla contrast is statistically significant,

$$t(1023) = 4.160, \qquad p = 3.46 \times 10^{-5}. \tag{7}$$

Thus BBC significantly increases the relative sensitivity of the model to boundary representations compared with matched interior representations.

Mean-replacement intervention. To verify that the result is not specific to zero-vector ablation, we repeat the experiment using a less disruptive intervention: the selected representation is replaced by the mean of the preceding $K - 1$ hidden states. We obtain

Model	Δ_b	Δ_i	$\Delta_b - \Delta_i$
Vanilla	0.6267 ± 0.0683	0.6251 ± 0.0670	0.0017 ± 0.0626
BBC	0.5612 ± 0.0636	0.5546 ± 0.0616	0.0065 ± 0.0584

The Vanilla gap is small relative to its per-example variability, whereas BBC again exhibits a larger positive boundary preference. The paired contrast is significant:

$$t(1023) = 3.499, \qquad p = 4.86 \times 10^{-4}. \tag{8}$$

Taken together, the two interventions give the same qualitative result: standard next-token training shows little or no systematic preference for the fixed segment boundaries used by BBC, whereas BBC significantly increases the relative predictive importance of those boundary representations. This provides direct representation-level evidence that the BBC routing constraint causes useful information to become preferentially concentrated at the designated carrier positions.

A.7 Spectral diagnostics for CC

Table 14: Normalized spectral residual $\|H - H_m\|_F^2 / \|H\|_F^2$ at 100M training tokens. Lower is better approximated by the first m positional modes.

Model	$m/N = .05$	$m/N = .10$	$m/N = .20$	$m/N = .30$	$m/N = .50$
Vanilla	0.528	0.468	0.381	0.314	0.208
CC	0.413	0.359	0.285	0.232	0.150

B Additional Connections Between BBC and CC

We now consider the reverse direction. BBC restricts the value subspace available for cross-segment communication. Smoothness requires an additional condition on how attention changes between neighboring queries.

B.1 Cross-segment feature subspace

Consider one deterministic attention head. Let B be the union of boundary positions available to the queries, assigning zero weights to inaccessible positions. Let

$$A_B \in \mathbb{R}^{N_q \times |B|}, \qquad V_B \in \mathbb{R}^{|B| \times d_v}.$$

The cross-segment contribution is

$$O_{\text{cross}} = A_B V_B. \tag{9}$$

Rows of A_B need not sum to one, because local-prefix tokens also receive attention.

Proposition 5 (Cross-segment subspace constraint)**.**

$$\operatorname{rank}(O_{\text{cross}}) \le \min(N_q, |B|, d_v), \tag{10}$$

and every output row belongs to $\operatorname{span}\{v_b : b \in B\}$.

Proof. Each row of $A_B V_B$ is a linear combination of boundary values, and the rank of a product cannot exceed the rank of either factor. □

This constrains one head's cross-segment contribution, rather than the complete hidden state. The carrier-count bound need not tighten the value-dimension bound when $|B| \ge d_v$. Nor does low rank imply smoothness: alternating rows $u, -u, u, -u, \ldots$ have rank one and adjacent energy $4(N_q - 1)\|u\|_2^2$.

B.2 Conditional smoothness

For the adjacent-difference operator D,

$$\|DO_{\text{cross}}\|_F^2 \le \|V_B\|_{\text{op}}^2 \|DA_B\|_F^2. \tag{11}$$

Thus similar boundary-attention weights yield similar cross-segment outputs.

For this comparison, restrict B to the fixed, nonempty set of boundaries preceding one segment. Let $K_B \in \mathbb{R}^{|B| \times d_k}$ contain their keys as rows, and restrict V_B to the same positions. Assume boundary logits $q_i^\top k_b / \sqrt{d_k}$. Define the boundary-normalized distribution and total boundary attention mass:

$$\overline{a}_i = \operatorname{softmax}\left(\frac{K_B q_i}{\sqrt{d_k}}\right), \qquad \gamma_i = \sum_{b \in B} (A_B)_{ib}. \tag{12}$$

Then

$$z_i = V_B^\top \overline{a}_i, \qquad o_i^{\text{cross}} = \gamma_i z_i. \tag{13}$$

Here $0 \le \gamma_i \le 1$; the remaining attention mass belongs to the local prefix.

Write E_{intra} for adjacent energy restricted to this segment, and let Q^{qry} be the query matrix. Set

$$V_{\max} = \max_{b \in B} \|v_b\|_2, \qquad C_B = \frac{\|V_B\|_{\text{op}} \|K_B\|_{\text{op}}}{2\sqrt{d_k}}.$$

Proposition 6 (Conditional BBC-to-CC smoothness)**.** *For fixed boundary keys and values,*

$$E_{\text{intra}}(O_{\text{cross}}) \le \left(V_{\max} \sqrt{E_{\text{intra}}(\gamma)} + C_B \sqrt{E_{\text{intra}}(Q^{\text{qry}})}\right)^2. \tag{14}$$

Proof. The softmax Jacobian is $J = \operatorname{diag}(p) - pp^\top$. It is symmetric and has absolute row sums $2p_i(1 - p_i) \le 1/2$, so its Euclidean operator norm is at most $1/2$. Thus

$$\|z_{i+1} - z_i\|_2 \le C_B \|q_{i+1} - q_i\|_2.$$

Using $o_i^{\text{cross}} = \gamma_i z_i$, $\|z_i\|_2 \le V_{\max}$, and $0 \le \gamma_i \le 1$ gives

$$\|o_{i+1}^{\text{cross}} - o_i^{\text{cross}}\|_2 \le V_{\max} |\gamma_{i+1} - \gamma_i| + C_B \|q_{i+1} - q_i\|_2.$$

The triangle inequality over consecutive differences gives the result. □

Both terms matter. With one boundary value u, zero local values, and identical zero logits, the cross-segment output changes from $u/2$ to $u/3$ when the local prefix grows from one token to two, even with identical queries.

BBC therefore transfers smoothness when both queries and total boundary attention mass vary smoothly. Transitions between segments, where the boundary set changes, require a separate term.

C Additional Theoretical Results

C.1 Scale and normalized roughness

Let the numerical stabilizer satisfy $\epsilon > 0$.

Define

$$\bar{h} = \frac{1}{N}\sum_{i=1}^{N} h_i, \qquad H_c = H - \mathbf{1}\bar{h}^\top.$$

Since $L\mathbf{1} = 0$, we have $E(H) = E(H_c)$. Let

$$S(H) = \|H_c\|_F^2, \qquad R_\epsilon(H) = \frac{E(H)}{S(H)+\epsilon}.$$

Then

$$E(H) = (S(H)+\epsilon)R_\epsilon(H). \tag{15}$$

Raw energy therefore depends on both positional magnitude and normalized roughness. For $S(H) > 0$, the unstabilized ratio $R(H) = E(H)/S(H)$ is invariant to uniform rescaling.

Temporarily treating H as an independent optimization variable, gradient flow for $\beta E(H)/(N-1)$ is

$$\frac{dH}{dt} = -\frac{2\beta}{N-1}LH. \tag{16}$$

It preserves the mean and gives

$$\frac{d}{dt}\|H_c\|_F^2 = -\frac{4\beta}{N-1}E(H) \le 0. \tag{17}$$

This is a representation-space calculation. Parameter updates act through the Jacobian of H and need not produce this flow, even when optimizing CC alone. The calculation motivates separate scale and roughness diagnostics rather than a monotonicity claim about Transformer training.

C.2 Finite-range graph energies

Define

$$E_r(H) = \sum_{i=1}^{N-r} \|h_{i+r} - h_i\|_2^2, \qquad E_1(H) = E(H).$$

Proposition 7 (Adjacent path controls finite-range variation)**.** *For $1 \le r < N$,*

$$E_r(H) \le r^2 E_1(H).$$

For $1 \le k < N$, $\alpha_r \ge 0$, $\alpha_1 = 1$, and $E^\alpha_{\le k}(H) = \sum_{r=1}^{k} \alpha_r E_r(H)$,

$$E_1(H) \le E^\alpha_{\le k}(H) \le \left(\sum_{r=1}^{k} \alpha_r r^2\right) E_1(H). \tag{18}$$

Proof. Let $d_t = h_{t+1} - h_t$. Since $h_{i+r} - h_i = \sum_{t=0}^{r-1} d_{i+t}$,

$$\|h_{i+r} - h_i\|_2^2 \le r \sum_{t=0}^{r-1} \|d_{i+t}\|_2^2.$$

Each adjacent difference occurs in at most r windows, giving $E_r \le r^2 E_1$. The weighted bounds follow by including the adjacent term and summing the upper bounds. □

Fixing $\alpha_1 = 1$ removes arbitrary overall scaling. Within this family the path is the weakest unnormalized penalty, while controlling finite-distance variation. This ordering does not imply an ordering of validation NLL.

A nonconstant r-periodic sequence can satisfy $E_r = 0$ but $E_1 > 0$, so a pure distance-r graph with $r > 1$ admits no uniform reverse bound.

For $\overline{E}_r = E_r/(N-r)$,

$$\overline{E}_r(H) \le r^2 \frac{N-1}{N-r} \overline{E}_1(H).$$

Comparisons between graph losses must therefore specify their weights, normalization, and regularization coefficients.

The implemented CC-2 penalty. The experimental CC-2 objective combines differences before squaring. For $N \ge 3$, define its unnormalized energy by

$$F_2(H) = \sum_{i=1}^{N-2} \|2h_{i+2} - h_{i+1} - h_i\|_2^2 = (N-2)\mathcal{L}_{\text{CC-2}}. \tag{19}$$

Here F_2 differs from the distance-two edge energy E_2 defined above.

This objective has a directed Laplacian interpretation. Orient unit-weight edges from each position j to its available predecessors $j-1$ and $j-2$. In the out-degree Laplacian $L_{\leftarrow,2}$, row $j \ge 3$ has coefficients $2, -1, -1$ at columns $j, j-1, j-2$, respectively. Let $B_2 \in \mathbb{R}^{(N-2)\times N}$ contain rows $3, \dots, N$ of this Laplacian. Then

$$F_2(H) = \|B_2 H\|_F^2 = \operatorname{Tr}(H^\top B_2^\top B_2 H). \tag{20}$$

Thus the implementation penalizes a squared Laplacian residual over positions with two predecessors, rather than a sum of separately squared edge differences.

Proposition 8 (Path comparison for the implemented CC-2 penalty)**.** *For $N \ge 3$,*

$$\begin{aligned} F_2(H) = E_1(H) + 2E_2(H) \\ - 2\|h_2 - h_1\|_2^2 + \|h_N - h_{N-1}\|_2^2. \end{aligned} \tag{21}$$

Consequently,

$$E_1(H) \le F_2(H) + 2\|h_2 - h_1\|_2^2 \le 9E_1(H). \tag{22}$$

In particular, $F_2(H) \le 9E_1(H)$.

Proof. Let $d_i = h_{i+1} - h_i$. Then $2h_{i+2} - h_{i+1} - h_i = d_i + 2d_{i+1}$, and

$$\begin{aligned} \|d_i + 2d_{i+1}\|_2^2 = 2\|d_i + d_{i+1}\|_2^2 \\ - \|d_i\|_2^2 + 2\|d_{i+1}\|_2^2. \end{aligned}$$

Summing over $i = 1, \dots, N-2$ gives Equation 21. Since $E_2(H)$ and the final-edge term are nonnegative, this identity gives the lower bound in Equation 22.

For the upper bound, Cauchy–Schwarz gives

$$\|d_i + 2d_{i+1}\|_2^2 \le 3\|d_i\|_2^2 + 6\|d_{i+1}\|_2^2.$$

After summing,

$$F_2(H) \le 9E_1(H) - 6\|d_1\|_2^2 - 3\|d_{N-1}\|_2^2.$$

Adding $2\|d_1\|_2^2$ proves the upper bound. □

The initial-edge term cannot generally be omitted. For example, the scalar trajectory $(h_1, h_2, h_3) = (0, 2, 1)$ has $F_2(H) = 0$ but $E_1(H) = 5$. More generally, nonzero increments satisfying $d_{i+1} = -d_i/2$ give $F_2(H) = 0$ while retaining positive path energy.

For the normalized training losses, the comparison becomes

$$\begin{aligned} \mathcal{L}_{\text{CC-2}} &\le 9\frac{N-1}{N-2}\mathcal{L}_{\text{CC}}, \\ \mathcal{L}_{\text{CC}} &\le \frac{N-2}{N-1}\mathcal{L}_{\text{CC-2}} + \frac{2}{N-1}\|h_2 - h_1\|_2^2. \end{aligned} \tag{23}$$

The initial-edge term is part of the theoretical comparison; it is not added to the implemented loss. Adjacent CC therefore controls the implemented CC-2 penalty, while the reverse comparison requires control of the first adjacent difference. The unconditional lower bound for the pairwise graph family does not carry over to CC-2.

C.3 Summed token-to-boundary distortion

A token r edges before its boundary satisfies $\|h_i - h_{b_j}\|_2^2 \le rE_j$. Since

$$\sum_{r=0}^{K-1} r = \frac{K(K-1)}{2}, \qquad \sum_j E_j \le E(H),$$

we obtain

$$\sum_j \sum_{i \in I_j} \|h_i - h_{b_j}\|_2^2 \le \frac{K(K-1)}{2} E(H). \tag{24}$$

This is a geometric consequence of low path energy. Section 5 gives the corresponding representative-attention approximation.

C.4 CC and cosine distance

For this identity, assume both neighboring states are nonzero and omit the numerical stabilizer ε. Define

$$\cos\theta_i = \frac{h_{i+1}^\top h_i}{\|h_{i+1}\|_2 \|h_i\|_2},$$

we have

$$\frac{\|h_{i+1} - h_i\|_2^2}{\|h_{i+1}\|_2^2 + \|h_i\|_2^2} = 1 - \frac{2\|h_{i+1}\|_2\|h_i\|_2}{\|h_{i+1}\|_2^2 + \|h_i\|_2^2}\cos\theta_i. \tag{25}$$

When neighboring hidden states have equal norm, this reduces to $1 - \cos\theta_i$. Thus normalized roughness is closely related to cosine distance, while additionally accounting for differences in neighboring hidden-state magnitudes.

D Issues in the STP/LLM-JEPA Implementation

CC is partly inspired by the geometric prior on hidden-state trajectories introduced in (Huang et al., 2026b). While reproducing STP and LLM-JEPA results, we identified a subtle issue in the official implementation of both papers that affects the reported performance gains. The issue originates in the `DataCollator` used for training, which masks the labels of all padding tokens. For models whose `EOS` and `PAD` tokens share the same id, this also masks the `EOS` token itself, removing the training signal that teaches the model when to terminate the generation. Models with distinct `EOS` and `PAD` tokens are unaffected. We flagged this to the authors, who confirmed the issue and have since updated the implementation[1].

After fixing the bug, we reproduced the results using `Llama-3.2-1B-Instruct` on the NL-RX-SYNTH dataset. NTP accuracy rose sharply, from 62.35% to 86.2%, while STP improved more modestly, from 84.35% to 86.31%. This gap suggests that STP is notably more robust to this failure mode than NTP. To probe this further, we ran a preliminary experiment masking the last two tokens. After one epoch, STP recovered to 72.55% accuracy, compared to 25.4% for NTP. This robustness did not extend to masking more than two tokens, however. Understanding why STP is able to partially recover from this kind of signal loss, while NTP is not, remains an interesting direction for future work.

[1] https://github.com/galilai-group/llm-jepa/commit/1c07b44fa2b0408a064be0ffc7816b5281384d7f