

Where Does Authorship Signal Emerge in Encoder-Based Language Models?

Francis Kulumba*
Inria Paris
Sorbonne Université

Guillaume Vimont
IRIF

Laurent Romary
Inria Paris

Florian Cafiero
LRE, EPITA
École nationale des chartes – PSL

Abstract

Authorship attribution asks whether two texts were written by the same author. Models fine-tuned for this task with the same pretrained encoder, data, and loss can still differ four-fold in Recall@20 depending only on their scoring mechanism. We use mechanistic interpretability tools to explain this gap. Stylistic features such as word length, punctuation density, and function-word frequency are similarly available at every layer in every model we probe, including an off-the-shelf control encoder, suggesting that differences in feature availability do not explain the gap. Instead, causal intervention shows that the scoring mechanism determines where the encoder consolidates authorship signal. Mean pooling forces consolidation in the middle third of the network, while late interaction defers it to the upper third. We further derive this difference from the gradient structure of each scorer, and training dynamics reveal distinct learning trajectories that follow from that difference.

1 Introduction

Authors leave identifiable traces in their writing. Features such as sentence length, punctuation habits, function-word preferences, and word-length distributions provide information about authorship, even when different individuals write on the same topic (Mosteller and Wallace, 1963; Burrows, 2002; Kešelj et al., 2003). Authorship attribution (AA) refers to the task of determining, given two passages, whether they were written by the same individual or group. This task can prove valuable in forensic linguistics (Dauber et al., 2019) and historical document analysis (Cafiero and Camps, 2019), among other applications.

Contemporary AA systems use a contrastive learning setup. A pretrained text encoder maps each passage to a representation (Vaswani et al.,

2017; Devlin et al., 2019). A scoring function compares these representations to produce a similarity score (Wegmann et al., 2022; Ai et al., 2022; Huertas-Tato et al., 2024; Kantharuban et al., 2026). Fine-tuning encourages the encoder to assign high scores to passages by the same author and low scores to passages by different authors. This approach often yields strong results, but recent work has isolated a gap driven by the scoring function. Kulumba et al. (2025) trained multiple models on a scholarly corpus with decorrelated topic and authorship. They showed that the scoring mechanism alone explains most of the observed four-fold Recall@20 gap. All models share the same pretrained backbone, training data, and contrastive loss. The only difference is the pooling or scoring mechanism. One group of models averages all token representations into a single vector before scoring (mean pooling). Another compares token representations directly using late interaction (LI) (Khattab and Zaharia, 2020).

A large gap emerges from what appears to be only a difference in the final comparison step? There are at least two plausible explanations. The first is that different scoring mechanisms may cause the encoder to learn different internal representations during fine-tuning. Mean pooling may force the encoder to discard fine-grained stylistic information that LI retains. Alternatively, the encoder may learn similar representations regardless of the scoring function. The gap may arise from the specific encoder layers each scorer accesses, which is determined during training. We apply the interpretability toolkit (Alain and Bengio, 2017; Vig et al., 2020; Belinkov, 2022; Goldowsky-Dill et al., 2023; Zhang and Nanda, 2023) to the fine-tuned encoders from Kulumba et al. (2025) to distinguish between these explanations. This analysis isolates the dissociation between feature *availability* and feature *use*.

• **Availability is invariant**, the same stylistic

* Correspondence: francis.kulumba@inria.fr

features (word length, capitalization, punctuation density, etc.) are linearly readable from the hidden states of all models at all layers, including a control encoder picked off the shelf. The pretrained backbone already encodes these features. Contrastive fine-tuning does not create them.

- **Usage depends on the scoring mechanism**, as it determines where in the encoder authorship signal becomes causally necessary. Mean pooling consolidates authorship signal in the middle third of the network, while LI defers consolidation to the upper third. The gradient structure of the scoring functions can explain this gap.

The results indicate that the choice of scoring function determines the effective depth of the encoder, the information accessible to the model, and the trajectory followed during training. Furthermore, consolidation depth predicts task performance: applying each scorer at intermediate layers shows that its accuracy plateau aligns with its consolidation band. This finding provides an explanation for the consistent outperformance of interaction-based systems over pooled representations in AA, despite both approaches utilizing the same pretrained backbone. We release the code under an open license.¹

2 Background

This section defines the fundamental components of the contrastive authorship attribution (AA) pipeline and describes the analytical tools employed in its study.

2.1 Contrastive authorship attribution

In the contrastive formulation, training data consists of triplets (a, p, n) : an anchor passage a , a same-author positive p , and a different-author negative n . The encoder f_θ maps each passage to a sequence of token-level representations. A scoring function s then compares the anchor’s representation to the positive’s and to the negative’s, producing scalar similarity scores. Training minimizes the InfoNCE loss (van den Oord et al., 2019):

$$\mathcal{L} = -\log \frac{\exp(s(a, p)/\tau)}{\exp(s(a, p)/\tau) + \sum_{n' \in \mathcal{N}} \exp(s(a, n')/\tau)} \quad (1)$$

Here, τ denotes a temperature parameter and $\mathcal{N}$ represents the set of in-batch negatives, where every non-positive passage in the batch serves as a negative.

¹<https://github.com/Madjakul/DeepStylometry>

2.2 Scoring mechanisms

The encoder produces a sequence of token representations $\mathbf{H}^a = [\mathbf{h}_1^a, \dots, \mathbf{h}_m^a] \in \mathbb{R}^{m \times d}$ for a passage containing m tokens with hidden dimension d . The scoring function specifies how this matrix is converted into a scalar similarity. Three families of scoring mechanisms are examined.

Mean pooling with cosine similarity. The passage representation is the mean of its token embeddings, and the score is the cosine similarity between mean vectors. Mean pooling is the standard AA baseline (Rivera-Soto et al., 2021; Wegmann et al., 2022; Kantharuban et al., 2026). It compresses the entire token sequence into a single d -dimensional vector before scoring.

Late interaction (MaxSim). The passage is represented by its full sequence of token embeddings, and the score is the sum over anchor tokens of the maximum cosine similarity to any candidate token (Khattab and Zaharia, 2020):

$$s_{\text{LI}}(a, p) = \sum_{i=1}^{m_a} \max_{j \in [m_p]} \cos(\mathbf{h}_i^a, \mathbf{h}_j^p) \quad (2)$$

Unlike mean pooling, late interaction preserves per-token structure through the scoring function, so the encoder is not required to compress all information into a single vector.

Patch-level late interaction (PLI). PLI provides an intermediate approach. The token sequence is divided into contiguous patches of size n . Each patch is mean-pooled, and MaxSim is then applied at the patch level:

$$s_{\text{PLI}}(a, p) = \sum_{i=1}^{P_a} \max_{j \in [P_p]} \cos(\mathbf{p}_i^a, \mathbf{p}_j^p) \quad (3)$$

Here, $\mathbf{p}_i = \frac{1}{n} \sum_{t \in \text{patch}_i} \mathbf{h}_t$ denotes the mean of the tokens within patch i . We use $n=2$ (bigram patches) in this study.

2.3 Residual stream patching

Residual stream patching (Vig et al., 2020; Meng et al., 2022) is a causal intervention designed to measure each encoder layer’s contribution to the model’s output. By corrupting the input of the encoder and subsequently restoring one layer’s activations to their clean values, it is possible to assess the extent to which the model’s correct behavior is recovered.

Given a triplet (a, p, n) , the *clean pass* encodes the positive p , the *corrupt pass* encodes the negative n , and the *patched pass* at layer ℓ encodes the negative while replacing its layer- ℓ hidden states with those of the positive. These patched activations then propagate through the remaining layers to produce $s_{\text{patched}}^{(\ell)}$.

The clean score is defined as $s_{\text{clean}} = s(a, p)$, while the corrupt score is $s_{\text{corrupt}} = s(a, n)$. If patching at layer ℓ restores the clean score, this indicates that layer ℓ contains the information necessary for correct authorship scoring. Conversely, if patching has no effect, the relevant information has not yet been consolidated at that layer.

2.4 Recovery metrics

Recovery is quantified using two metrics:

Percentage recovery is a standard metric introduced by Meng et al. (2022):

$$\text{Recovery}^{(\ell)}(\%) = \frac{s_{\text{patched}}^{(\ell)}(a, n) - s(a, n)}{s(a, p) - s(a, n)} \times 100 \quad (4)$$

where $s_{\text{patched}}^{(\ell)}(a, n)$ denotes the score obtained when the negative’s layer- ℓ hidden states are replaced with those of the positive. A value of 0% indicates no recovery, while 100% indicates full recovery. Values may fall outside the $[0, 100]$ range if the patched score overshoots. The denominator becomes small when the clean and corrupt scores are close, a situation that occurs most frequently for PLI, which can cause a small score change to result in an extreme percentage.

Rank recovery handles this issue by posing a binary question: after patching at layer ℓ , does the model continue to rank the positive above the negative?

$$r_{\text{rank}}^{(\ell)} = \frac{1}{|\mathcal{T}_+|} \sum_{t \in \mathcal{T}_+} \mathbf{1}[s_{\text{patched}}^{(\ell)}(a_t, p_t) > s_{\text{patched}}^{(\ell)}(a_t, n_t)] \quad (5)$$

where $\mathcal{T}_+$ denotes the set of triplets for which the clean model scores the positive passage above the negative. This metric yields a value in $[0, 1]$, with 0.5 representing chance performance. Rank recovery is used for all the figures in the main text, while percentage recovery is reported in the appendix.

2.5 LISA probes

To separate feature availability from feature use, we train linear probes (Alain and Bengio, 2017; Belinkov, 2022) at each encoder layer. The probes are regression models mapping the mean-pooled hidden state at layer ℓ to scalar stylistic features.

We report the coefficient of determination R^2 on a held-out set. The feature targets are drawn from the Linguistically-Interpretable Style Attributes (LISA) framework, introduced by Patel et al. (2024) and extended by Kantharuban et al. (2026). They include nine categories: word length, capitalization rate, type-token ratio, punctuation density, function-word frequency, sentence length, hedging markers, citation density, and discourse connectives.

3 The Consolidation Bottleneck

This section presents a theoretical framework outlining the expected outcomes before any experiments.

3.1 How the gradient distributes across tokens

The end-to-end gradient of the InfoNCE loss with respect to a single token representation $\mathbf{h}_j^a$ factors into two parts:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}_j^a} = \underbrace{\frac{\partial \mathcal{L}}{\partial s}}_{\text{InfoNCE term}} \cdot \underbrace{\frac{\partial s}{\partial \mathbf{h}_j^a}}_{\text{Scorer term}} \quad (6)$$

The InfoNCE term concentrates the gradient on hard negatives and remains consistent across different scoring mechanisms, as it depends solely on the values rather than the computation method. In contrast, the scorer term dictates the distribution of the gradient across individual tokens, which is where the three mechanisms diverge.

Mean pooling: dense, uniform gradient. Under mean pooling, the score depends on each token only through the mean. The partial derivative is:

$$\frac{\partial s_{\text{mean}}}{\partial \mathbf{h}_j^a} = \frac{1}{m} \cdot \frac{\partial \cos(\bar{\mathbf{h}}^a, \bar{\mathbf{h}}^p)}{\partial \bar{\mathbf{h}}^a} \quad (7)$$

The $1/m$ factor ensures that each token receives an identical gradient magnitude, resulting in a dense and uniform gradient distribution. Consequently, the model lacks a mechanism to selectively enhance discriminative tokens; function words, punctuation marks, and content words all receive the same gradient signal.

MaxSim: dense and sparse. Under LI, the score is computed as a sum of per-token maxima (Equation 2). Each term in the outer sum depends on a single anchor token $\mathbf{h}_i^a$ via the max operation over candidate tokens. For each anchor token, the

max selects one candidate token and disregards the others. Formally:

$$\frac{\partial s_{LI}}{\partial \mathbf{h}_i^a} = \sum_{j=1}^{m_p} \mathbf{1} \left[j = \arg \max_{j' \in [m_p]} \cos(\mathbf{h}_i^a, \mathbf{h}_{j'}^p) \right] \cdot \frac{\partial \cos(\mathbf{h}_i^a, \mathbf{h}_j^p)}{\partial \mathbf{h}_i^a} \quad (8)$$

The indicator is 1 only when candidate token j is the optimal match for anchor token i , and 0 otherwise, reducing the sum to a single term. This leads to two contrasting consequences. On the anchor side, the gradient is dense: every anchor token receives an update, as in mean pooling. However, the direction of the update is determined by the specific candidate token the anchor aligns to instead of a passage-wide average. On the candidate side, the gradient is sparse: candidate tokens that do not win any alignment receive no update from the scoring function. This asymmetry is significant for the encoder, which learns to identify tokens carrying discriminative signals because only aligned pairs propagate gradients.

PLI: intermediate density. Under PLI with patch size n (Equation 3), the gradient combines both regimes:

$$\frac{\partial s_{PLI}}{\partial \mathbf{h}_j^a} = \frac{1}{n} \cdot \mathbf{1}[\text{patch}(j) \in \text{argmax}] \cdot \frac{\partial \cos}{\partial \mathbf{h}_j^a} \quad (9)$$

The gradient is sparse between patches, as only selected patches receive a gradient signal, but dense within patches, with each of the n tokens in a selected patch receiving $1/n$ of the gradient.

3.2 The consolidation bottleneck

The uniform $1/m$ gradient produced by mean pooling results in a *consolidation bottleneck*. The scoring function accesses only the mean of all token representations. To enable the encoder to distinguish between same-author and different-author passages, it must organize the hidden states such that their mean encodes authorship information. This requires the encoder to coordinate information across the entire sequence, compressing authorship-relevant features into a representation that persists through averaging. This compression occurs at an intermediate layer, referred to as the *consolidation layer* or, when expressed as a confidence interval, the consolidation band.

MaxSim theoretically does not exhibit this bottleneck. The scoring function directly accesses individual token representations, allowing the encoder to refine token-specific features in the upper layers without consolidating them into a sin-

gle direction. Since the upper layers of a transformer encode more abstract, context-dependent features (Tenney et al., 2019), the ability to defer consolidation should enable MaxSim to leverage richer representations.

This analysis predicts a bottleneck for mean pooling but not for MaxSim. However, it does not specify the location of the consolidation layer. If the prediction holds, mean pooling should exhibit a recovery inflection in an earlier region of the network compared to MaxSim during causal patching experiments. Patching below the consolidation band should eliminate the signal, as the representation has not yet been compressed, whereas patching above it should preserve the signal, indicating that consolidation is complete.

Why does mean pooling lose information?

Mean pooling transforms the $m \times d$ token matrix $\mathbf{H}$ into a d -dimensional vector $\bar{\mathbf{h}}$. According to the data processing inequality, $I(Y; \bar{\mathbf{h}}) \leq I(Y; \mathbf{H})$, where Y denotes the author identity. This inequality is strictly negative whenever $\bar{\mathbf{h}}$ is not a sufficient statistic for Y . For example, two passages with identical function-word frequencies but different function-word orderings are indistinguishable under mean pooling, which is permutation-invariant, but can be distinguished by MaxSim, which preserves positional structure.

4 Experimental Setup

We conduct a controlled analysis that isolates the scoring mechanism. All models share a common backbone, corpus, and loss function, differing only in how they convert token representations into scalar similarity scores.

4.1 Models

All models use a ModernBERT-base backbone (Warner et al., 2025) with 22 transformer layers. This architecture yields 23 hidden states: the embedding output and 22 layer outputs. Unless otherwise specified, we use the base-4 split of HALvest-Contrastive (Kulumba et al., 2025), a scholarly corpus where the anchor and positive samples come from different papers by the same author set. The negative sample is drawn from the same disciplinary field. This design controls topical similarity. The model cannot rely on vocabulary overlap to distinguish positive from negative samples.

The single-vector models represent standard readouts in the literature. **Mean pooling** with InfoNCE is the approach adopted by contrastive attribution systems (Rivera-Soto et al., 2021; Huertas-Tato et al., 2024), while Wegmann et al. (2022) use mean pooling under a triplet loss. **Layerwise** applies layerwise attention pooling, followed by mean pooling and cosine scoring, as implemented in IDIOLEX (Kantharuban et al., 2026). Previous work shows that layerwise attention improves performance by less than one Recall@20 point over mean pooling. Learned layer weights do not overcome the single-vector bottleneck discussed in §3.2. **LI** applies token-level MaxSim with punctuation and padding masked. **PLI $n=2$** applies bigram patch-level MaxSim. **Multilingual-E5 (base)** (Wang et al., 2024) serves as an off-the-shelf control model. E5 was trained for retrieval and to a greater extent for semantic matching. Its similarity scores are decorrelated from those produced by models trained for authorship attribution (Kulumba et al., 2025; Kantharuban et al., 2026). Table 1

Model	R@20	R@100	nDCG@20	nDCG@100
Mean pooling	0.121	0.294	0.063	0.101
LI	0.485	0.678	0.364	0.408
PLI $n=2$	0.497	0.700	0.365	0.411
Multilingual-E5 (zero-shot)	0.167	0.269	0.124	0.146

Table 1: **HALvest-Contrastive base-4 retrieval performance**, reproduced from Kulumba et al. (2025). Multilingual-E5 was tested using mean pooling. Layerwise attention is omitted here as it differs by under one Recall@20 point from mean pooling in prior work (§4.1).

summarizes retrieval performance results. The four-fold difference in Recall@20 between mean pooling and LI constitutes the primary empirical observation under investigation.

4.2 Probe set construction

We use a small, controlled set of 148 triplets, distinct from the full retrieval benchmark, for our analysis. A curated probe set in place of the full test set allows us to control confounding variables such as passage length and domain overlap. We sample triplets from the HALvest-Contrastive base-4 validation set, specifically from the ten most frequent author-sets with at least four distinct documents.

Triplets are stratified into three tiers, each varying the relationship between the anchor and the negative passage:

- **Tier A** ($n=50$): the anchor and positive share the same author-set. The negative is written

by a completely disjoint author-set from the same scholarly domain. This is the baseline: the model must rely on stylistic signal to distinguish the positive from a topically similar negative written by entirely different authors.

- **Tier B** ($n=50$): the anchor and positive share the same author-set. The negative is written by a partially overlapping author-set that shares at least one author with the anchor’s team but is not identical to it. The shared author contributes stylistic signal to both passages, creating a confound. This tier tests whether the model can distinguish full author-set matches from partial ones.
- **Tier C** ($n=48$): the anchor and positive share the same author-set but come from different scholarly domains (anchor in domain D_1 , positive in domain D_2). The negative is written by a disjoint author-set from the anchor’s domain D_1 . This tier tests cross-domain authorship recognition. The model must identify the same authors when the vocabulary and conventions shift between disciplines.

Model	Tier A		Tier B		Tier C	
	Fail	n_+	Fail	n_+	Fail	n_+
Mean pooling	14%	43	42%	29	12.5%	42
Layerwise	10%	45	44%	28	12.5%	42
LI	2%	49	36%	32	4.2%	46
PLI $n=2$	2%	49	36%	32	4.2%	46

Table 2: Failure rates and effective sample sizes (n_+ = correctly-ranked triplets used for patching) per tier. Tier B is the hardest due to the shared-author confound. Interaction models are more robust than the single-vector scorer across all tiers.

Residual patching is applied exclusively to correctly ranked triplets, defined as cases where the clean model assigns a higher score to the positive than to the negative. Consequently, effective sample sizes vary by tier and model, as shown in Table 2.

4.3 Analyses

We conduct four types of analyses, each covering different subsets of the six models. Probing is performed on all six models. Patching, score sensitivity, and intermediate-layer scoring are applied to the four fine-tuned models. Training dynamics are analyzed for layerwise attention, LI, and PLI, as this analysis requires the full series of training checkpoints, which were retained only for these

three models.

1. **LISA probes** train regression models on a separate 10,000-passage corpus and evaluate them on a 2,000-passage held-out set, measuring feature availability at each of the 23 hidden states.
2. **Residual stream patching** measures the causal contribution of each layer via rank recovery (Equation 5) across the 148 probe-set triplets.
3. **Score sensitivity** computes the average absolute score change $|s_{\text{patched}}^{(\ell)} - s_{\text{corrupt}}|$ per layer, a raw measure of how much the scoring function’s output responds to restoring a single layer.
4. **Training dynamics** apply patching to eight checkpoints per model (steps 0, 500, 1500, 3000, 5000, 10000, 20000, and final) to track how the depth profile develops during training. It isolates what contrastive fine-tuning adds.

5 Results

Probing, causal patching, score sensitivity, and training dynamics all point to the scorer’s extraction location and method as the source of the performance gap.

5.1 Feature availability is invariant

If the four-fold Recall@20 gap between mean pooling and LI were due to LI enabling the encoder to learn superior stylistic representations, LISA probes would be expected to show higher R^2 values for LI than for mean pooling at certain layers. However, this pattern is not observed. Table 3 presents

Feature	Layerw.	LI	PLI	E5	Mean	Base
wl_mean	.575	.580	.579	.576	.578	.571
fw_we	.499	.500	.505	.493	.499	.497
fw_the	.496	.496	.496	.496	.496	.496
punct_period	.493	.468	.502	.479	.486	.466
punct_colon	.473	.473	.473	.463	.471	.473

Table 3: Top-5 LISA features ranked by mean peak R^2 across models. All six models surface the same feature family with highly similar probe performance. Full heatmaps are in Appendix A.

the top-5 LISA features by peak R^2 for each model. The rankings are consistent across all six models: mean word length is the most prominent feature ($R^2 \approx 0.57$ – 0.58), followed by function-word frequencies and punctuation density. The pretrained ModernBERT backbone ("Base"), which has not been fine-tuned on authorship data, achieves the

same R^2 as the four fine-tuned models and the Multilingual-E5 control. The full probe heatmaps (Appendix A) reveal the same pattern at every layer of the five 22-layer models and the same ordering in the 12-layer Multilingual-E5 control at proportionate depths. Mean-pooled probes cannot detect structures present only in individual token representations, which is where late interaction would most plausibly differ. Therefore, token positions were probed directly, and mean-pool was compared against max-pool aggregation. Neither analysis is supported by any model (Appendix F), so neither informs the central question.

Within the stylistic features and linear probes evaluated in this study, there is no evidence that different scorers produce distinct representations. Differences in untested features, particularly higher-order syntactic or discourse features, cannot be excluded. The pretrained ModernBERT backbone already encodes the tested features, and contrastive fine-tuning does not generate them, regardless of the scoring function. Consequently, the four-fold Recall@20 gap is more plausibly attributed to differences in the readout mechanism.

5.2 A scoring-dependent depth profile

Residual stream patching is applied to the 148 probe-set triplets (Figure 1). The consolidation layer is defined as the earliest layer at which rank recovery exceeds 0.75, quantified by resampling the probe-set triplets 1000 times per tier (Appendix D).

The single-vector scorers recover in three stages (Figure 1, blue and purple). Recovery sits above chance at layers 0 to 2, falls below chance through the lower-middle of the network, then rises to near-perfect by the mid-layers. Every model shows the early bump, which is consistent with the first layers encoding shallow syntactic statistics (Jawahar et al., 2019) that carry distributional authorship signal even when domain-specific vocabulary shifts. Layerwise attention’s 95% bootstrap CI for the consolidation layer is [10, 13] on all three tiers, with point estimates of 11, 13, and 10. Plain mean pooling sits in the same place, with a point estimate of 13 on every tier and a CI of [10, 13] on Tier A. On Tier B and Tier C, its lower bound falls to 0, because the early bump puts rank recovery above 0.75 at layer 0 in 4.0% and 12.7% of resamples. The bump is not consolidation: recovery falls back below the threshold before rising again, so the sustained-crossing estimator returns [13, 19] and

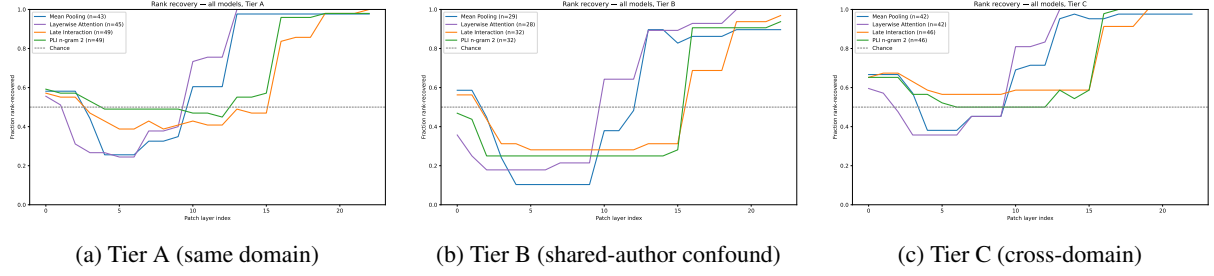


Figure 1: **Rank recovery across the four fine-tuned models.** Each panel shows one tier. Blue: mean pooling, purple: layerwise attention (both single-vector scorers), orange: LI, green: PLI $n=2$. Dashed line: chance (0.5). Single-vector scoring consolidates in the middle third of the network, while both interaction models consolidate in the upper third. The two single-vector curves track each other on all three tiers (Table 4). The regime separation is consistent across all three tiers.

[10, 13] for those two cells (Appendix D).

Late interaction shows the same three stages with a later final rise (Figure 1, orange). Its CI is [16, 17] on Tier A and [16, 19] on Tier B, in both cases clear of the single-vector upper bound of 13. On Tier C, the point estimate stays at 16 while the lower bound falls to 0, from the same early bump, present in 7.5% of resamples, and the sustained-crossing estimator returns [16, 16] there. The mid-layer dip is deeper than for the single-vector scorers (recovery ≈ 0.3 –0.4): restoring a clean mid-layer into an otherwise corrupt pass actively misleads the token-level MaxSim scoring, which selects its matches from single token vectors and is therefore more exposed to a mismatch between the restored layer and its corrupt context.

PLI $n=2$ tracks LI closely (Figure 1, green) and is the most stable model in the analysis. Its point estimate is layer 16 on all three tiers, at every threshold from 0.65 upward, and its CI is [16, 16] on Tier A and Tier B. This places it inside the interaction regime instead of between the two regimes, which is what Equation 9 predicts if the patch-level argmax dominates the intra-patch averaging.

Regime separation. The two groups do not meet. Every single-vector interval has an upper bound of 13 and every interaction interval a lower bound of 16, on Tier A and on Tier B. Hence, the separation survives the tier where all four models lose accuracy and where the effective sample size falls to 28 correctly ranked triplets. Tier C separates on the point estimates (13 and 10 against 16 and 16) and on the sustained-crossing intervals, but not on the first-crossing intervals, for the reason given above. The separation appears at the 0.75 threshold and holds to 0.90 (Appendix E). The point estimates are

stable from 0.60 upward, and the intervals become informative at 0.75.

Score sensitivity analysis (Appendix B) and training dynamics across eight checkpoints (Appendix C) provide complementary confirmation: single-vector scorers show high normalized sensitivity precisely in the layers where they consolidate, and the depth profile develops top-down during training.

5.3 Linking consolidation to performance

Patching isolates the layers that each model’s scorer requires. It does not show that consolidation depth differences set the performance gap. We test this link by intervening on depth and measuring attribution accuracy. We extract hidden states at every encoder layer, apply each model’s scoring function, and measure triplet accuracy on the probe set (Figure 2).

On Tier A, both single-vector scorers plateau near 0.85 by the mid-layers. LI improves across layers and reaches 0.95 in the upper layers. PLI tracks LI. Each scorer’s performance plateau matches its consolidation band from causal patching: the single-vector plateau aligns with the pooling consolidation band ([10, 13]), and LI’s plateau aligns with its consolidation band ([16, 17]). On Tier B, all models reach lower accuracy. LI peaks at 0.65, compared to 0.95 on Tier A. The relative ordering holds: LI reaches the highest plateau, PLI follows, and single-vector scorers remain below. Tier B lowers the maximum accuracy for all models but preserves their order. The accuracy gap between readouts persists, and the consolidation intervals remain distinct. On Tier C, the pattern matches Tier A. Layerwise attention attends to any layer below ℓ that has already consolidated. This ac-

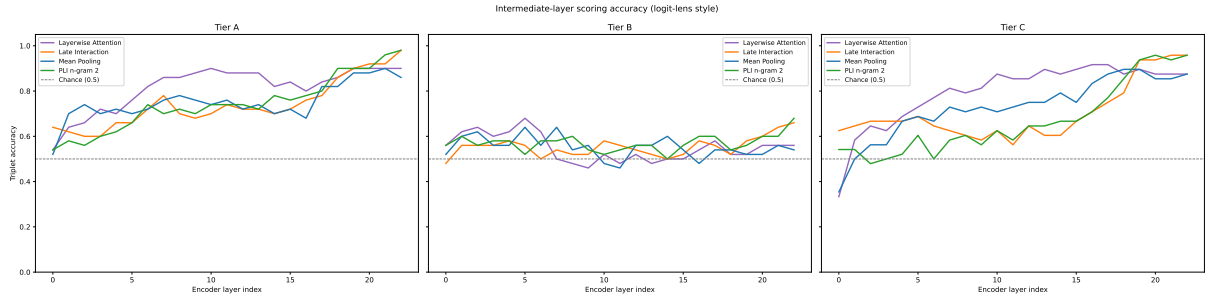


Figure 2: **Intermediate-layer scoring accuracy.** Triplet accuracy when each model’s scorer reads from layer ℓ , shown per tier. On Tier A, single-vector scorers plateau around 0.85 by mid-layers while LI reaches approximately 0.95 in the upper layers. On Tier B, all models achieve lower accuracy, but the relative ordering is preserved. The depth at which each scorer reaches its accuracy plateau matches the consolidation bands from causal patching.

cess may explain its slightly higher intermediate accuracy.

6 Related Work

Authorship attribution. Neural AA has progressed from classification approaches (Burrows, 2002; Schler et al., 2006) to contrastive learning methods (Wegmann et al., 2022; Kantharuban et al., 2026; Huertas-Tato et al., 2024), with a growing emphasis on addressing topic confounding (Wegmann and Nguyen, 2021; Rivera-Soto et al., 2021). While previous work in the AA community has explored interpretability (Alshomary et al., 2025b,a), to the best of our knowledge, our study is the first to employ mechanistic interpretability tools and gradient analysis to explain performance and training dynamics in encoder models.

Probing versus causal analysis. Linear probes (Belinkov, 2022) are commonly employed to investigate the information encoded in neural representations; however, the relationship between probe accuracy and actual model behavior remains debated (Hewitt and Liang, 2019; Ravichander et al., 2021). Activation patching (Vig et al., 2020; Meng et al., 2022; Wang et al., 2023) offers a causal perspective by evaluating whether information is necessary for model predictions rather than merely decodable. Our dissociation between *availability* and *use* advances this discussion by demonstrating that all probed features are equally available across models exhibiting markedly different task performance.

7 Discussion

The availability–use dissociation presents AA as a problem of reading out information. Here, the pretrained encoder makes stylistic features easy to

read linearly. The key difference is whether the scoring function can reach these features at the right depth and with enough capacity.

Availability against use. In this framework, probing accuracy does not serve as a sufficient proxy for task performance when models differ in their scoring mechanism. All six models, produce nearly identical LISA probe profiles (Table 3), but their retrieval performance is very different (Table 1). So, the main question is not which model encodes more stylistic information, but which scoring method can actually extract it. Path patching at the attention-head level (Goldowsky-Dill et al., 2023) could help trace how stylistic signals move through the encoder, but this is a more detailed issue than what we focus on here.

Why interaction beats pooling. Both the gradient analysis and the information-theoretic argument (§3.1) show that mean pooling removes higher-order structure by turning representations into a single vector. At the same time, MaxSim keeps details at the token level. Causal depth profiles show that single-vector scorers consolidate in the middle third of the network, while interaction scorers do so in the upper third. When the task gets harder, both types shift to lower layers, but their order stays the same, as seen in Tier B.

In practice. Choosing a scoring function is just as important as picking an encoder. Most AA research has focused on encoder design, pre-training data, and the contrastive objective (Wegmann et al., 2022; Rivera-Soto et al., 2021; Huertas-Tato et al., 2024), and these systems usually use a single pooled vector for encoder readout. Our results show that the readout method matters as much as these other factors.

Limitations

Backbone choice. We fix the backbone to ModernBERT to control for architecture, since our goal is to isolate how the scoring mechanism shapes signal consolidation. The specific consolidation bands we observe may shift in other architectures. The qualitative regime separation between early consolidation under mean pooling and later consolidation under interaction follows from the gradient structure of the scoring functions (§3.1), not from properties of ModernBERT, so it should transfer to other architectures. Testing a second backbone, such as RoBERTa (Liu et al., 2019), would confirm this prediction but is orthogonal to the mechanism.

Patch-level interaction. We study only $n = 2$ for PLI. This keeps the analysis focused on the contrast between pooling and interaction while still giving us a middle regime to compare against LI and mean pooling. The theory suggests that larger patches should move the inflection earlier, closer to the pooling regime. Exploring $n = 3, 4, 5$ would be a natural extension, but it is not necessary for the main result reported here.

Probe set size. The 148 triplets resolve the regime separation between single-vector and interaction scorers on all three tiers under rank recovery, including Tier B, where the high failure rate leaves only 28 to 32 correctly ranked triplets. They are not enough for fine-grained LI versus PLI comparisons, whose intervals overlap everywhere, and they are not enough to place the consolidation layer to within a layer: the intervals span three to four layers where they are tightest. The probe results should be read as evidence about linear accessibility of the nine LISA feature categories studied here, not as a claim that the full hidden representations are equivalent across model families. We cannot rule out differences in features not tested, particularly higher-order syntactic or discourse features and non-linear feature interactions. Nor can we rule out differences at the token level. Mean-pooled probing cannot see structure that exists only in individual token representations, which is the natural place for late interaction to differ, and our attempt to probe token positions directly returned no fit in any model, including the pretrained backbone (Appendix F).

Corruption method. We corrupt by encoding the negative passage, so the corrupt pass is a state

the encoder actually produces and not an out-of-distribution perturbation. The cost is that the negative differs from the positive in more than authorship. It differs in content, in phrasing, and in every property that co-varies with a different author set writing a different paper, so a patch that restores authorship signal restores some of those as well. Tier C, where the positive comes from a different scholarly domain, bounds the topical part of this confound without removing it. A counterfactual that lacks authorship information while preserving everything else would be the cleaner control, and it is not well defined for this task.

One benchmark, one register. Every result reported here was measured on HALvest-Contrastive. Scholarly prose is a low-entropy register with standardized syntactic templates, which is what makes the stylistic residual isolable and may also be what makes a single pooled vector insufficient. The negatives are field-matched by construction, and the passages are 130 tokens. A regime separation between readouts on a corpus with those properties does not establish the same separation on conversational text, on single-author labels, or at document length.

Acknowledgments

The authors are grateful to Djamé Seddah who indirectly inspired this work. We also thank Wisam Antoun, Rian Touchent and Théo Lasnier for the productive discussions. This work was partially realized on computing HPC and storage resources provided by IDRIS thanks to the grant GCDA1016807 on the DALIA supercomputer.

References

- Bo Ai, Yuchen Wang, Yugin Tan, and Samson Tan. 2022. [Whodunit? learning to contrast for authorship attribution](#). In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, Volume 1: Long Papers*, pages 1142–1157, Online only. Association for Computational Linguistics.
- Guillaume Alain and Yoshua Bengio. 2017. [Understanding intermediate layers using linear classifier probes](#). In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.
- Milad Alshomary, Narutatsu Ri, Marianna Apidianaki, Ajay Patel, Smaranda Muresan, and Kathleen McKe-

- own. 2025a. [Latent space interpretation for stylistic analysis and explainable authorship attribution](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 1124–1135, Abu Dhabi, UAE. Association for Computational Linguistics.
- Milad Alshomary, Nikhil Reddy Varimalla, Vishal Anand, Smaranda Muresan, and Kathleen McKeown. 2025b. [Layered insights: Generalizable analysis of human authorial style by leveraging all transformer layers](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10279–10292, Suzhou, China. Association for Computational Linguistics.
- Yonatan Belinkov. 2022. [Probing Classifiers: Promises, Shortcomings, and Advances](#). *Computational Linguistics*, 48(1):207–219.
- John Burrows. 2002. [Delta: A measure of stylistic difference and a guide to likely authorship](#). *Literary and Linguistic Computing*, 17(3):267–287.
- Florian Cafiero and Jean-Baptiste Camps. 2019. [Why molière most likely did write his plays](#). *Science Advances*, 5(11):eaax5489.
- Edwin Dauber, Aylin Caliskan, Richard Harang, Gregory Shearer, Michael Weisman, Frederica Nelson, and Rachel Greenstadt. 2019. [Git blame who? stylistic authorship attribution of small, incomplete source code fragments](#). *Proceedings on Privacy Enhancing Technologies*, 2019(3):389–408.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1: Long and Short Papers*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- William Falcon and The PyTorch Lightning team. 2019. [PyTorch Lightning](#).
- Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. 2023. [Localizing Model Behavior with Path Patching](#). *arXiv preprint*. ArXiv:2304.05969 [cs].
- John Hewitt and Percy Liang. 2019. [Designing and interpreting probes with control tasks](#). 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)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.
- Javier Huertas-Tato, Adrián Girón-Jiménez, Alejandro Martín, and David Camacho. 2024. [Isolating authorship from content with semantic embeddings and contrastive learning](#). *arXiv preprint*. ArXiv:2411.18472 [cs].
- Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. [What does BERT learn about the structure of language?](#) In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.
- Anjali Kantharuban, Aarohi Srivastava, Fahim Faisal, Orevaoghene Ahia, Antonios Anastasopoulos, David Chiang, Yulia Tsvetkov, and Graham Neubig. 2026. [IDIOLEX: Unified and continuous representations for idiolectal and stylistic variation](#). *arXiv preprint*. ArXiv:2604.04704 [cs].
- Vlado Kešelj, Fuchun Peng, Nick Cercone, and Calvin Thomas. 2003. N-gram-based author profiles for authorship attribution. In *Proceedings of the Conference of the Pacific Association for Computational Linguistics*, pages 255–264.
- Omar Khattab and Matei Zaharia. 2020. [ColBERT: Efficient and effective passage search via contextualized late interaction over BERT](#). In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20*, pages 39–48, New York, NY, USA. Association for Computing Machinery.
- Francis Kulumba, Wissam Antoun, Guillaume Vimont, Laurent Romary, and Florian Cafiero. 2025. [Halvest-contrastive: Retrieval-like authorship attribution with patch-level late interaction](#). *Preprint*, arXiv:2407.20595.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [RoBERTa: A robustly optimized BERT pretraining approach](#). *Preprint*, arXiv:1907.11692.
- Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22*, pages 17359–17372, Red Hook, NY, USA. Curran Associates Inc.
- Frederick Mosteller and David L Wallace. 1963. Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed federalist papers. *Journal of the American Statistical Association*, 58(302):275–309.
- Ajay Patel, Nicholas Andrews, and Chris Callison-Burch. 2024. [Low-resource authorship style transfer: Can non-famous authors be imitated?](#) *Preprint*, arXiv:2212.08986.
- Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. 2021. [Probing the probing paradigm: Does probing accuracy entail task relevance?](#) In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3363–3377, Online. Association for Computational Linguistics.

- Rafael A. Rivera-Soto, Olivia Elizabeth Miano, Juanita Ordonez, Barry Y. Chen, Aleem Khan, Marcus Bishop, and Nicholas Andrews. 2021. [Learning universal authorship representations](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 913–919, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W. Pennebaker. 2006. Effects of age and gender on blogging. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, volume 6, pages 199–205.
- Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. [BERT rediscovers the classical NLP pipeline](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.
- Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. [Representation learning with contrastive predictive coding](#). *Preprint*, arXiv:1807.03748.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.
- Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. [Investigating gender bias in language models using causal mediation analysis](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc.
- Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023. [Interpretability in the wild: a circuit for indirect object identification in GPT-2 small](#). In *The Eleventh International Conference on Learning Representations*.
- Liang Wang, Nan Yang, Xiaolong Huang, Binling Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024. [Text embeddings by weakly-supervised contrastive pre-training](#). *Preprint*, arXiv:2212.03533.
- Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning, ICML ’20*. JMLR.org.
- Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas Adams, Jeremy Howard, and Iacopo Poli. 2025. [Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers*, pages 2526–2547, Vienna, Austria. Association for Computational Linguistics.
- Anna Wegmann and Dong Nguyen. 2021. [Does it capture STEL? a modular, similarity-based linguistic style evaluation framework](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7109–7130, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Anna Wegmann, Marijn Schraagen, and Dong Nguyen. 2022. [Same author or just same topic? towards content-independent style representations](#). In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 249–268, Dublin, Ireland. Association for Computational Linguistics.
- Fred Zhang and Neel Nanda. 2023. [Towards Best Practices of Activation Patching in Language Models: Metrics and Methods](#). In *The Twelfth International Conference on Learning Representations*.

A LISA probe heatmaps

The main-text Table 3 summarizes feature availability by reporting peak R^2 values. Figure 3 shows the full layer-by-layer heatmaps for all models. Each row is a LISA feature category (word length, capitalization, type–token ratio, punctuation density, function-word frequency, sentence length, hedging markers, citation density, and discourse connectives). Each column is an encoder layer. The color scale runs from 0 (feature not linearly decodable) to 0.5 (strongly decodable). The five 22-layer models produce visually indistinguishable heatmaps: the same features become readable at the same layers regardless of the scoring mechanism used during fine-tuning, or whether the model was fine-tuned at all. Multilingual-E5 uses a 12-layer architecture, so its heatmap has 13 columns (layers 0–12) instead of 23, and it shows the same feature ordering at proportionate depths..

B Score sensitivity

Score sensitivity measures how much a model’s output score changes when a single layer’s clean activations are restored during patching. Unlike rank recovery (§2.4), which asks a binary question (does the model still rank the positive above the negative?), score sensitivity captures the continuous magnitude of the perturbation effect.

The three scoring mechanisms produce scores on different scales: cosine similarity in $[-1, 1]$ for

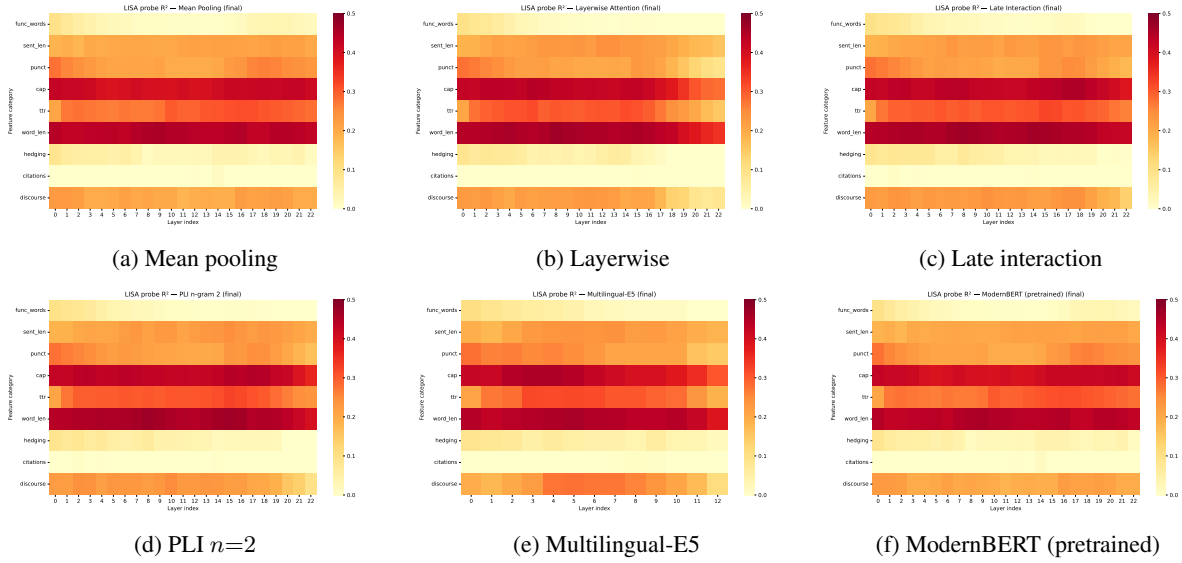


Figure 3: **LISA probe R^2 heatmaps at the final checkpoint.** Rows are stylistic feature categories; columns are encoder layers. All models produce nearly identical heatmaps.

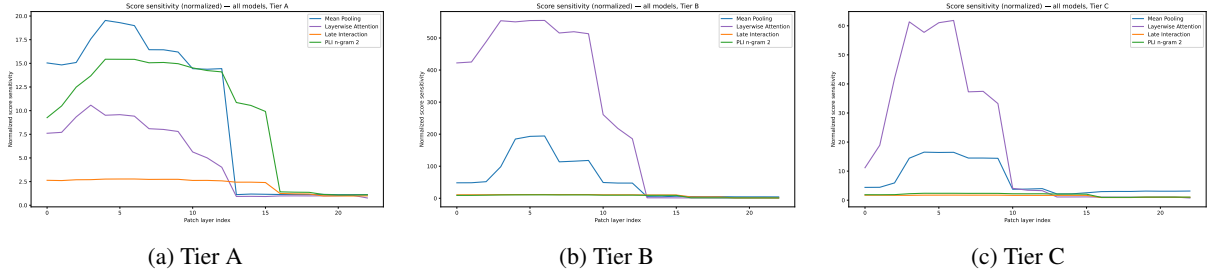


Figure 4: **Normalized score sensitivity per layer.** Score sensitivity normalized by the clean score magnitude. Single-vector scorers show high relative sensitivity through the lower and middle layers and drop sharply at the top of their consolidation band. Late interaction shows low flat sensitivity throughout. On Tier B (b), layerwise reaches 400–550 $\times$ due to near-zero clean-corrupt gaps on the hardest triplets.

mean pooling versus unbounded sums over tokens for MaxSim and PLI. Comparing absolute score changes across scorers therefore confounds mechanism with scale. We normalize each model’s sensitivity by its clean score magnitude (Figure 4).

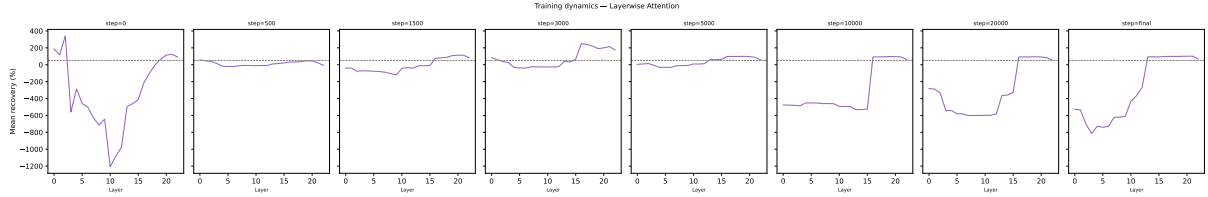
Normalization reverses the cross-model ordering. On Tier A (Figure 4a), mean pooling shows the highest relative sensitivity (peaking at $\approx 20\times$ in the lower layers), followed by PLI ($\approx 15\times$) and layerwise ($\approx 10\times$). Both single-vector scorers drop sharply after the mid-layers. Late interaction shows low flat sensitivity ($\approx 2.5\times$) across all layers. On Tier B (Figure 4b), layerwise reaches 400–550 $\times$, an artifact of near-zero clean-corrupt gaps on the hardest triplets. The Tier C pattern (Figure 4c) mirrors Tier A at lower magnitude.

The cross-model ordering after normalization reflects score-scale differences, not mechanism. The informative quantity is the per-layer *shape* for each model. Single-vector scorers show high sensitivity

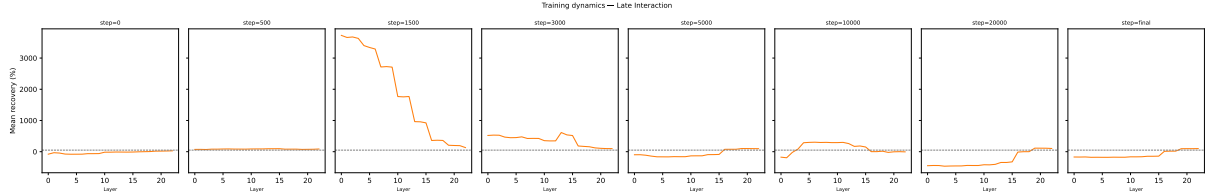
precisely in the layers where they consolidate, then drop to near zero once consolidation is complete. Late interaction shows no such inflection: no single layer is disproportionately responsible for its output. This is consistent with the absence of a consolidation bottleneck under MaxSim. These per-model inflection points are stable across all three tiers.

C Training dynamics

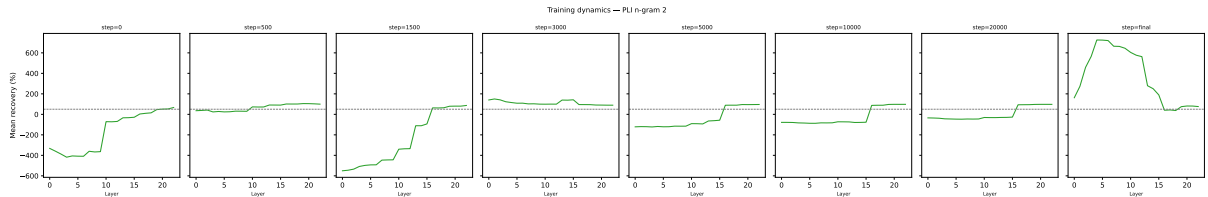
The patching analysis in §5.2 shows the final-checkpoint depth profile. To understand how that profile develops during training, we apply the same analysis to intermediate checkpoints saved at steps 0, 500, 1500, 3000, 5000, 10000, 20000, and the final checkpoint. We use the layerwise model for mean pooling because we had access to all intermediate checkpoints for that model and only the final checkpoint for plain mean pooling (§4.1). Figure 5 shows the mean percentage recovery across Tier A



(a) Layerwise (single-vector scorer): top-down monotonic. Upper layers learn first, the inflection migrates downward during training.



(b) Late interaction: transient early-layer spike at step 1500, then signal migration to upper layers.



(c) PLI $n=2$: gradual emergence, no transient spike. The final checkpoint shows a distinctive mid-layer hump (layers 10–15) absent in both other models.

Figure 5: **Training dynamics.** Mean percentage recovery across Tier A triplets at eight checkpoints. Each subplot is one checkpoint. x-axis: layer index; y-axis: mean recovery. We use percentage recovery here because rank recovery is binary and too coarse to track gradual signal emergence at early checkpoints. The y-axis extremes reflect the known instability of percentage recovery (§2.4) and should not be read as meaningful magnitudes.

triplets at each checkpoint.

Layerwise (Figure 5a) learns top-down. At step 500, recovery is concentrated at the uppermost layers. As training progresses, the inflection migrates downward. The model progressively recruits deeper layers to consolidate earlier, consistent with the consolidation bottleneck (§3.2). The dense gradient initially refines the layers closest to the scoring function, then gradually shapes earlier layers.

Late interaction (Figure 5b) shows a distinctive behavior. At step 1500, recovery spikes at layers 5–10. This suggests the model initially exploits shallow lexical matches: MaxSim can propagate the gradient through exact token matches at negligible cost, providing a cheap authorship signal from lower layers. As hard negatives increase in difficulty during training, this shortcut becomes insufficient, and the model shifts to deeper, more contextualized representations. By step 5000, this transient behavior is suppressed, and recovery concentrates at the upper layers.

PLI $n=2$ (Figure 5c). Bigram-patch MaxSim shows a third pattern with no early spike: the intra-patch averaging smooths out the shallow matches that LI exploits. Recovery emerges gradually at the upper layers. The final checkpoint shows a mid-layer hump (layers 10–15) unique to PLI, possibly reflecting the two-level structure of its gradient (Equation 9).

D Bootstrap confidence intervals

We compute the consolidation bands of §5.2 by resampling the probe-set triplets 1000 times per tier, recomputing the per-layer rank-recovery curve within each resample, and recording the earliest layer at which it exceeds 0.75. The 95% CI is the 2.5th and 97.5th percentile of those 1000 layer indices. Table 4 reports the result.

Recovery is above chance at the first layers, below chance through the lower-middle of the network, and high thereafter (§5.2), so the earliest crossing can fall at layer 0 whenever a resample puts the early bump above the threshold. We report the fraction of resamples for which it does. That

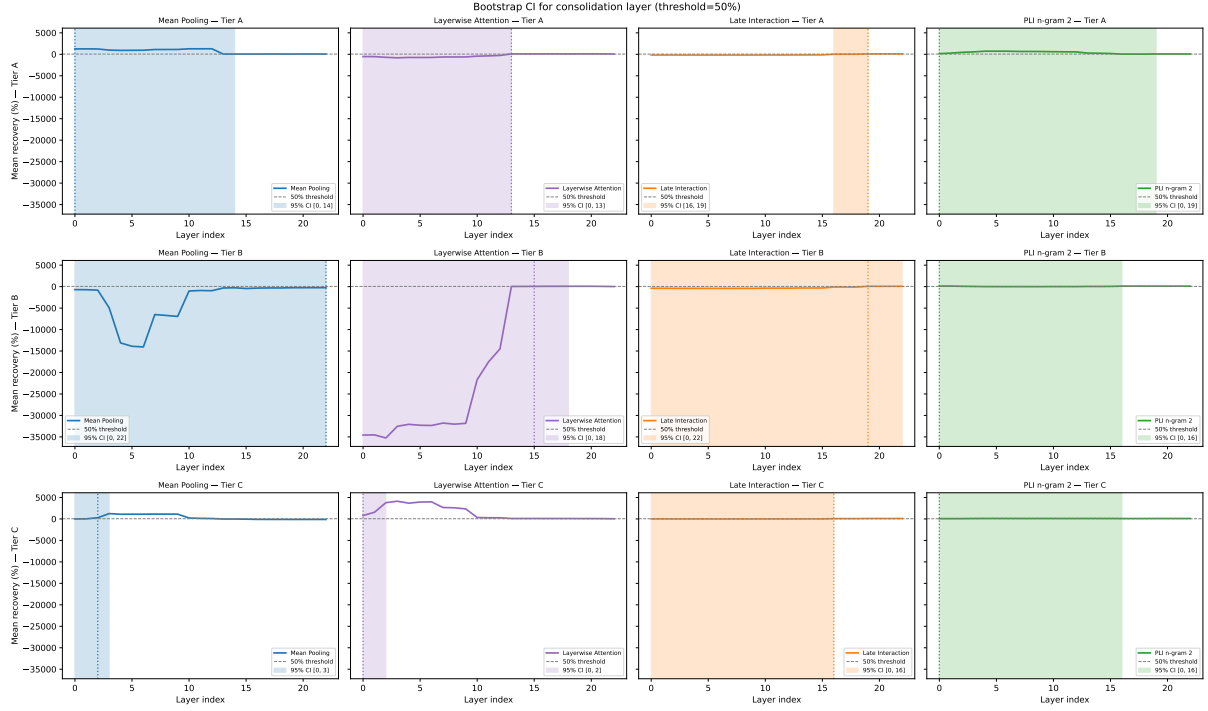


Figure 6: **Bootstrap CIs for the consolidation layer across all three tiers.** Each panel shows the mean percentage recovery curve for one model on one tier, with the 95% bootstrap CI for the consolidation layer (shaded). The shaded region bounds the layer index, not the recovery percentage. The y -axis reaches $-35,000\%$ for mean pooling on Tier B, which puts the 50% threshold within the line width of zero. This is what makes the crossing unresolvable under this metric (§2.4) and what motivates the rank-recovery estimates in Table 4.

Model	Tier	Point est.	95% CI	Std
Mean Pooling	A	13	[10, 13]	1.41
	B	13	[0, 13]	2.62
	C	13	[0, 13]	4.55
Layerwise Attn.	A	11	[10, 13]	1.53
	B	13	[10, 13]	1.06
	C	10	[10, 13]	1.68
Late Interaction	A	16	[16, 17]	0.97
	B	19	[16, 19]	2.74
	C	16	[0, 16]	5.21
PLI $n=2$	A	16	[16, 16]	1.43
	B	16	[16, 16]	0.66
	C	16	[0, 16]	4.06

Table 4: Bootstrap confidence intervals for the rank-recovery consolidation layer (threshold = 0.75, $B = 1,000$).

Model	Tier	Point est.	95% CI	Std
Mean pooling	A	0	[0, 14]	4.51
Mean pooling	B	22	[0, 22]	9.54
Mean pooling	C	2	[0, 3]	1.84
Layerwise	A	13	[0, 13]	2.73
Layerwise	B	15	[0, 18]	4.81
Layerwise	C	0	[0, 2]	1.48
LI	A	19	[16, 19]	1.04
LI	B	19	[0, 22]	8.17
LI	C	16	[0, 16]	5.76
PLI $n=2$	A	0	[0, 19]	8.61
PLI $n=2$	B	0	[0, 16]	7.96
PLI $n=2$	C	0	[0, 16]	6.57

Table 5: Bootstrap CI summary for the consolidation layer under *percentage* recovery (50% threshold, 1000 resamples). Table 4 reports the same quantity under rank recovery. Mean pooling and PLI return a point estimate of 0 on Tier A here and 13 and 16 there.

fraction is below 2% in eight of the twelve cells, and every lower bound of 0 in Table 4 comes from one of the four where it is not: mean pooling on Tier B (4.0%) and Tier C (12.7%), and LI (7.5%) and PLI (6.9%) on Tier C.

As a check, we recompute every interval with a sustained crossing, the earliest layer above threshold at which the curve stays above threshold for

all deeper layers. On Tier C the sustained intervals are [10, 13], [10, 13], [16, 16] and [16, 16], so the early bump accounts for the whole of the Tier C widening and the tier separates cleanly once it is removed. On Tier B, the sustained estimator widens mean pooling instead, to [13, 19]. Neither

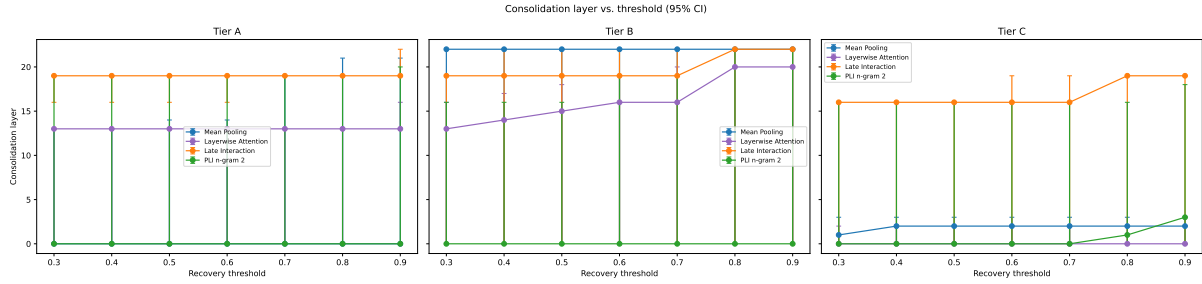


Figure 7: **Consolidation layer against percentage-recovery threshold** (0.30–0.90 in steps of 0.10), with 95% bootstrap CIs, shown per tier. Blue: mean pooling. Purple: layerwise attention. Orange: LI. Green: PLI $n=2$. The flatness reflects the scale of the underlying curves (Figure 6), which puts a 50% threshold within the line width of zero, and not a stable estimate.

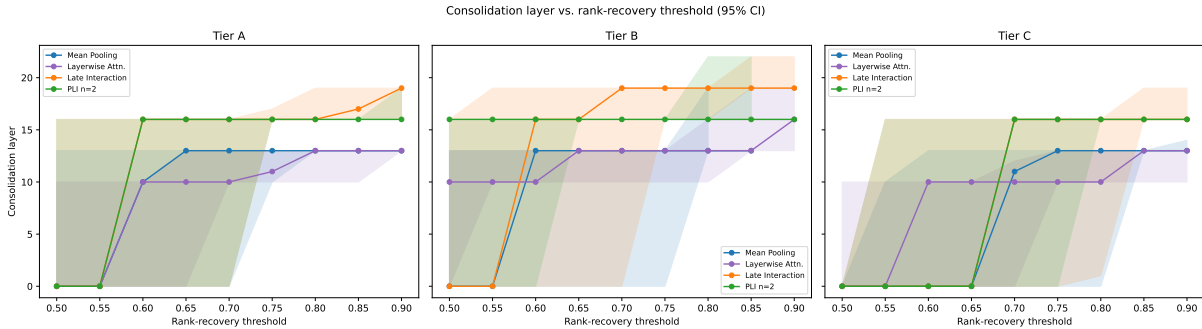


Figure 8: **Consolidation layer against rank-recovery threshold** (0.50–0.90 in steps of 0.05), with 95% bootstrap CIs, shown per tier. Blue: mean pooling. Purple: layerwise attention. Orange: LI. Green: PLI $n=2$. Point estimates are stable from 0.60 upward. The bands below 0.75 are wide because a growing share of resamples cross at the early bump (Appendix D).

estimator is clean on all three tiers. We report the first crossing as primary because it is the estimator the rest of the analysis uses, and switching to the one that flatters the result would not be a reason to switch.

On Tier A and Tier B, the separation is clean: the upper bound of every single-vector interval is 13, and the lower bound of every interaction interval is 16. Tier B separates despite the low effective sample sizes there, which the percentage-recovery intervals did not show. On Tier C, the point estimates separate, 13 and 10 against 16 and 16, while the first-crossing intervals do not, for the reason given above.

The two metrics also disagree about Tier C. Percentage recovery puts single-vector consolidation at layers 2 and 0 there, rank recovery at 13 and 10. The gap is not noise: on cross-domain triplets, patching an early layer moves the score most of the way back toward clean while leaving the ordering wrong, so the score recovers early, and the decision recovers late. Rank recovery reports the second, which is the quantity retrieval depends on.

E Threshold sensitivity

The consolidation layer depends on a recovery threshold and on which recovery metric defines the crossing. We sweep both. Figure 8 sweeps the rank-recovery threshold from 0.50 to 0.90 in increments of 0.05, the range over which rank recovery is above chance. Figure 7 sweeps the percentage-recovery threshold from 0.30 to 0.90 in increments of 0.10. Both report 95% bootstrap CIs from 1000 resamples.

Under rank recovery, the point estimates are stable from 0.60 upward. Both single-vector scorers stay between layers 10 and 13 on Tier A, late interaction between 16 and 19, and PLI at 16 on every tier without moving once. The intervals become informative later. Below 0.75 they are dominated by resamples whose earliest crossing falls at the early bump, a share that runs from 1.1% at 0.75 to 44% at 0.60 for mean pooling on Tier A, and no pair of intervals separates at 0.70 or below. From 0.75 to 0.90, the single-vector and interaction intervals are disjoint on Tier A and Tier B.

Under percentage recovery, the sweep does not

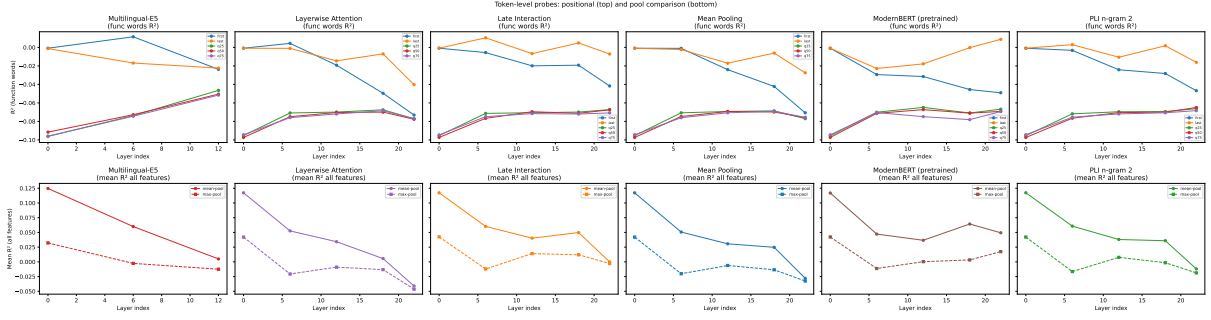


Figure 9: **Token-level probes.** Top row: positional probes (R^2 at five token positions) for function-word frequency. Bottom row: mean R^2 across all features comparing mean-pool and max-pool aggregation. R^2 is at or below zero throughout, in every model including the pretrained backbone, so neither analysis discriminates between the scorers.

move at all. On Tier A, mean pooling and PLI return layer 0 at every threshold from 0.30 to 0.90, and layerwise and LI return 13 and 19 at every threshold. The flatness is not stability. Figure 6 shows the reason: the percentage-recovery curves reach $-35,000\%$ on Tier B, so a 50% threshold sits within the line width of zero and is crossed either at the first layer or nowhere. The estimator is saturated and not robust. This is the behavior §2.4 predicts when the clean–corrupt gap is small, and it is why the main text reports rank recovery.

F Token-level probes

The LISA probes in the main text (§5.1) use mean-pooled hidden states, which cannot detect features accessible only at the token level. To test whether late interaction fine-tuning creates token-level features invisible to mean-pooled probing, we run two additional analyses on all six models (four fine-tuned on authorship, the pretrained ModernBERT backbone, and the Multilingual-E5 control).

The first analysis trains positional probes at five fixed sequence positions (first token, last token, and the tokens at the 25th, 50th, and 75th percentile of the sequence). The top row of Figure 9 shows R^2 for function-word frequency at each position. No model shows a positional pattern that differs qualitatively from the others.

The second analysis compares two aggregation strategies over all token representations: mean-pool (the strategy used in the main-text probes) and max-pool (which selects the maximum activation per feature dimension). The bottom row of Figure 9 shows the mean R^2 across all LISA features for each strategy. Again, no model shows a scorer-specific advantage under max-pool that is absent under mean-pool.

The positional probes ask whether a single token

position encodes a passage-level aggregate such as mean word length or punctuation density. There is no particular reason it should, and the near-zero R^2 across all six models is what that mis-specification predicts. An analysis that fails to separate the pre-trained backbone from the fine-tuned models cannot establish that they are equivalent. What remains open is stated in the Limitations.

G Training hyperparameters

All models are trained with InfoNCE loss (Equation 1) using the hyperparameters below, reproduced from Kulumba et al. (2025): Table 6 repro-

Hyperparameter	Value
Backbone	ModernBERT-base
Epochs	1
Per-device batch size	32
Gradient accumulation	2
GPUs	4 × H100 (primary), 4 × GB200 (scaling)
Effective in-batch negatives	256 (H100), 512 (GB200)
Learning rate	3×10^{-5}
Weight decay	0.1
Precision	16-bit mixed
InfoNCE temperature τ	0.5

Table 6: Training and model hyperparameters shared across all main-body models as reported by Kulumba et al. (2025).

duces the training configuration of every model. τ is a temperature parameter, set to 0.5 throughout. All other settings are defaults of the PyTorch Lightning (Falcon and The PyTorch Lightning team, 2019) training pipeline.

H Probe set construction

Passages are standardized to a target length of 130 tokens (Figure 10), with the positive and negative passages within each triplet differing by no more than five tokens after tokenization.

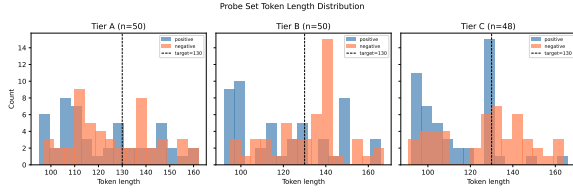


Figure 10: Token length distributions for positive (blue) and negative (orange) passages across the three tiers. All passages cluster around the 130-token target.

I Alignment and uniformity losses

The alignment–uniformity framework of Wang and Isola (2020) is employed to characterize the geometry of learned representations. Alignment, $\alpha = \mathbb{E}_{(a,p) \sim p_{\text{pos}}} \|f(a) - f(p)\|^2$, quantifies the expected squared distance between representations of same-author passages, where lower values indicate tighter same-author clusters. Uniformity, $u = \log \mathbb{E}_{(x,y) \stackrel{\text{i.i.d.}}{\sim} p_{\text{data}}} \exp(-2 \|f(x) - f(y)\|^2)$, assesses how evenly all representations are distributed on the hypersphere. Lower values indicate a more uniform spread. An effective contrastive representation minimizes both metrics. It keeps same-author pairs close while avoiding collapse of the overall distribution. The alignment–uniformity tradeoff

Model	α ($\downarrow$)	u ($\downarrow$)
Pretrained (no fine-tuning)	0.069	−0.171
Mean pooling	0.154	−2.000
LI	0.091	−0.331
PLI n -gram 2	0.201	−0.717

Table 7: Alignment α and uniformity u per model, from Kulumba et al. (2025). Lower is better for both.

(Table 7) does not track the capacity gap described in §3.2, and the way it fails is informative. Mean pooling achieves the best uniformity, because averaging spreads representations, and LI the tightest alignment among the fine-tuned models, because token-level comparison preserves discriminative detail at the cost of a candidate-side sparse gradient that does not push representations apart as aggressively. But mean pooling also has better alignment than PLI $n=2$ and better uniformity, so it dominates PLI on both metrics while losing to it by a factor of four in Recall@20 (Table 1). The diagnostic of Wang and Isola (2020) reads the geometry of the final representation. If the performance gap were a property of that geometry, the diagnostic would

order the models correctly. It does not, which is a first indication that the gap lives somewhere the final representation does not record.