The Geometry of Prompting: Unveiling Distinct Mechanisms of Task Adaptation in Language Models

Artem Kirsanov
New York University
kirsaa01@nyu.edu

Chi-Ning Chou
Flatiron Institute

Kyunghyun Cho
New York University
Genentech

SueYeon Chung
New York University
Flatiron Institute

Abstract

Decoder-only language models have the ability to dynamically switch between various computational tasks based on input prompts. Despite many successful applications of prompting, there is very limited understanding of the internal mechanism behind such flexibility. In this work, we investigate how different prompting methods affect the geometry of representations in these models. Employing a framework grounded in statistical physics, we reveal that various prompting techniques, while achieving similar performance, operate through distinct representational mechanisms for task adaptation. Our analysis highlights the critical role of input distribution samples and label semantics in few-shot in-context learning. We also demonstrate evidence of synergistic and interfering interactions between different tasks on the representational level. Our work contributes to the theoretical understanding of large language models and lays the groundwork for developing more effective, representation-aware prompting strategies.

1 Introduction

A striking feature of modern language models (LMs) is their computational flexibility. Unlike traditional neural networks trained for specific tasks, LMs function as flexible computers that can be programmed (prompted) with natural language to perform a wide array of tasks.

This adaptability, often termed in-context learning (ICL), has revolutionized natural language processing by enabling rapid task adaptation without expensive fine-tuning. However, despite ICL's widespread success, its underlying mechanisms remain poorly understood.

While some research has linked ICL to gradient-based learning (von Oswald et al., 2023; Akyürek et al., 2023), recent evidence in naturalistic settings suggests that ICL may not be pure "learning", but rather a method of steering the model to select familiar tasks from its pretraining corpus (Pan et al., 2023; Hendel et al., 2023). Recent studies have also highlighted importance of prompt design and demonstrated that the choice of examples and output labels can significantly impact performance (Zhao et al., 2021; Min et al., 2022). However, these works have primarily focused on the input-output behavior of LMs, leaving the internal dynamics of ICL largely unexplored.

In this work, we aim to illuminate ICL by investigating how different prompting methods modify internal representations in pre-trained language models. When a model is prompted to perform a classification task, we analyze the separability and geometric properties of category manifolds — point clouds in the model's embedding space corresponding to examples sharing a category label. We leverage the recently developed framework of **manifold capacity** (Chung et al., 2018; Chou et al., 2024), which analytically connects task performance to the geometric properties of these representations.

Our core contributions are:

1. A comprehensive analysis of how various prompting methods affect internal representations in language models, revealing distinct computational mechanisms despite similar performance outcomes.

2. Novel insights into in-context learning dynamics, including the role of label semantics, synergistic effects of demonstrations on unrelated tasks, and representational trade-offs during task adaptation.

2 Related work

2.1 Prompting as task-adaptation

The idea that a language model pretrained on next-token prediction can adapt to various tasks without parameter updates was popularized by (Brown

Findings of the Association for Computational Linguistics: NAACL 2025, pages 1855–1888
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

et al., 2020). This phenomenon, known as in-context learning (ICL), relates to the model's ability to effectively "learn" a novel task by analogy from a few demonstration examples provided in the input sequence. To distinguish conventional few-shot ICL from other recently proposed input-based task-adaption methods, we refer to it as providing **demonstrations**, highlighting the crucial role of task examples.

While performance generally improves with more examples (Brown et al., 2020; Bertsch et al., 2024), ICL exhibits counter-intuitive features, with performance being heavily dependent on the exact choice of examples, their ordering, formatting, and other factors (Zhao et al., 2021; Wang et al., 2024; Liu et al., 2024). Additionally, the actual input-output mapping matters less than expected (Min et al., 2022), suggesting that few-shot ICL involves a complex interplay of true task learning from examples and task recognition from the pre-training corpus (Pan et al., 2023).

Language models also demonstrate zero-shot learning abilities, performing tasks based on abstract descriptions without explicit examples (Radford et al., 2019; Wei et al., 2022). We refer to such task-adapting prompts without examples as **instructions**[1]. While often considered together under the umbrella of ICL, our results reveal that despite comparable performance, these two prompt types affect internal representations differently, highlighting the crucial role of input distribution examples.

Recently, prompt-tuning has emerged as an alternative approach to task adaptation (Lester et al., 2021; Liu et al., 2022). This method involves learning a small set of continuous vectors (soft prompts) that are concatenated to the input embeddings, while keeping the model parameters frozen. Prompt-tuning offers a middle ground between full model fine-tuning and static prompting, allowing for task-specific adaptations with significantly fewer trainable parameters.

2.2 Internal representations

Language computations rely on mapping individual words or tokens to vectors in a continuous embedding space, which possesses rich structure learned through model pretraining. The emerging *linear representation hypothesis* (Park et al., 2024) suggests that this embedding space contains "feature directions" encoding human-interpretable concepts, allowing the model to perform vector operations with meaningful semantics (Mikolov et al., 2013; Pennington et al., 2014; Bowman et al., 2016).

The concept of feature superposition (Elhage et al., 2022; Arora et al., 2018) provides insight into how a model can operate on more features than it has orthogonal directions in the embedding space. This is achieved by utilizing almost-orthogonal vectors for feature encoding with minimal interference, potentially circumvented by non-linear activation functions.

A popular method for uncovering encoded features involves training linear probes (Belinkov, 2021) to "read out" information linearly from the embedding space. Probing methods have revealed the encoding of part-of-speech tags (Belinkov et al., 2017), parse-tree geometry (Hewitt and Manning, 2019), and higher-level semantic features such as spatial location of landmarks (Gurnee and Tegmark, 2024) and color (Abdou et al., 2021). However, while these studies are usually performed on and averaged over a very diverse input corpus of text, there is a lack of understanding how the context preceding a given input (particularly, task adaptation) affects feature representation.

2.3 Representational geometry

The notion that underlying representations in the embedding space shape task performance has gained traction in both machine learning and computational neuroscience (Chung and Abbott, 2021; Flesch et al., 2022; Ansuini et al., 2019; Fawzi et al., 2018). Intuitively, for a classification task, this implies that collective representations of inputs sharing a target category (a category manifold) must be well-separated from other categories. This concept of "manifold untangling" has been a prominent perspective on computational objectives in neuroscience (DiCarlo and Cox, 2007).

The recently developed framework of manifold capacity (Chung et al., 2018; Wakhloo et al., 2023; Chou et al., 2024) proposes a formal link between representational geometry and separability. Manifold capacity quantifies how efficiently task-relevant features are encoded from the perspective of a linear downstream decoder. Essentially, it measures the separability of target classes in the embedding space, capturing the effectiveness of task-relevant feature encoding.

This framework has been successfully applied to investigate representational geometry in vision

[1] We use "instruction" referring only to the format of the prompt for zero-shot learning and do all experiments on base models that were not instruction fine-tuned

networks (Stephenson et al., 2019; Cohen et al., 2020; Stephenson et al., 2021) and language models (Mamou et al., 2020). By examining how different prompting methods affect manifold capacity, we can gain insights into the internal dynamics of ICL and the efficiency of various task adaptation strategies.

3 Methods

3.1 Dataset details

To investigate effects of various prompting methods on representations in different task-specific contexts, we required a dataset with control over multiple categorical dimensions of text. We could not find an existing text classification dataset with a sufficient number of samples and a comprehensive multilabel scheme suitable for tractable manifold analysis. Therefore, we leveraged a separate language model (Claude 3.5 Sonnet) to generate a synthetic dataset tailored to our research requirements. This synthetic dataset consists of diverse sentences, each simultaneously labeled with three types of categories: Sentiment, Topic, and Intent, with five categories for each type. Such multidimensional labeling allowed us to investigate representational effects in a multitasking setting (see sections 4.2 and 4.3).

For consistency, all experiments, including those focused on single-task performance (section 4.1), utilized this dataset, with the sentiment classification task serving as our primary focus. To validate our findings, we also replicated key single-task experiments using established open datasets as a control.

Full details on the datasets, including generation process, category distributions, and example sentences, are provided in the appendix A.1.

3.2 Task setup

Our work focuses on text classification tasks with a fixed set of categories, as such tasks have an analytically-grounded link between the geometry of underlying representation and separability of categories in the embedding space, ultimately determining the end performance.

In contrast to traditional encoder-based models, where separate linear classifiers are trained to predict target category directly from the embedding vectors, we investigate decoder-only language models. These models can prompted to generate class labels directly in the vocabulary space.

This approach introduces two key factors affecting performance:

1. **Representation Quality**: The underlying representation in the embedding space must support the separation of class manifolds.
2. **Readout Alignment**: The alignment between the model's unembed layer and the ideal decoder directions impacts the final output quality.

Manifold capacity theory allows us to disentangle these components by quantifying the representation quality at each layer, independently of the specific unembed module being used for vocabulary readout. This idea is schematically illustrated in fig. 1.

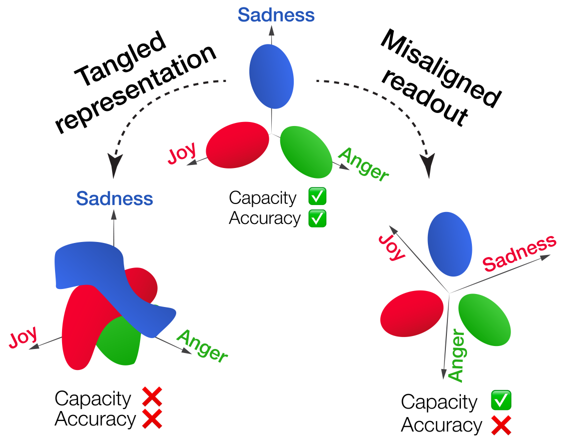

Figure 1: Two components of the model's performance. Low accuracy can be caused by either suboptimal and tangled representation in the embedding space (left), as well as misalignment between the representation and model's readout layer (right). Manifold capacity, which relates the performance of an ideal decoder to the underlying geometry can differentiate between the 2 cases.

3.3 Prompting strategies

Throughout the work we compared two main types of natural-language prompting. **Instruction** prompt consisted of the following text (using sentiment as an example):

> This is a text classification task. Possible categories are Joy, Sadness, Fear, Anger, Surprise.
> Text: *[Test Sentence]*
> Category:

where *[Test Sentence]* stands for the sentence text from the dataset that is being evaluated.

Demonstration prompt consisted of a variable number of examples following a similar format:

Text: *[Demo sentence 1]*
Category: Joy
Text: *[Demo sentence N]*
Category: Fear
Text: *Test Sentence*
Category:

As a baseline control for the representation analysis, we also extracted embedding using the **raw sentence** input of the following format:

Text: *[Test Sentence]*
Category:

3.4 Embedding Extraction

Analyzing representational geometry in decoder-only models presents unique challenges due to masked self-attention, distributed sentence-level features, and last token dependency. To address these challenges and investigate the effects of prompting on representations, we consider two types of embeddings:

1. **Sentence Embeddings**: We extract and residual stream activations for tokens corresponding only to the input sentence, excluding the task prompt, and average their embedding vectors along sequence dimension. This provides insight into the model's intermediate processing stage.

2. **Last-token Embeddings**: We extract residual stream activations of the last token in the sequence at each layer. This allows us to track how sentence-level features are aggregated into the final representation used for output generation.

These embedding types and possible effects of prompting are illustrated in fig. 2.

3.5 Analysis of representations

To analyse representational geometry we first construct category manifolds (point clouds) by accumulating the embedding vectors of all sentences sharing a class label. So for a classification task with P categories, the resulting representation can be thought of as P distinct collections of vectors in the embedding space. We then compute the following properties of the resulting collective representation.

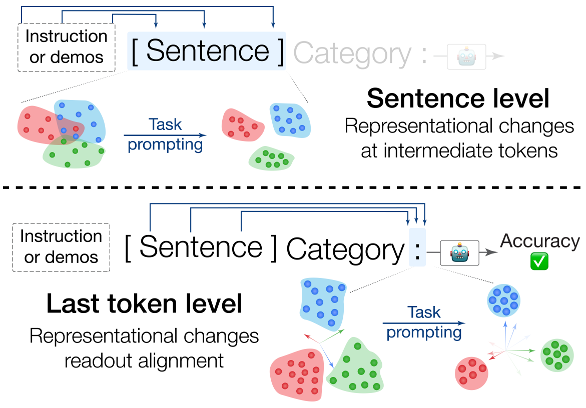

Figure 2: Possible effect sites of prompting. Task-specific prefix might affect extraction of relevant features at the sentence-level, reorganizing intermediate representations (top). High performance would also imply more efficient repackaging of extracted features into the embedding of the last token, as well readout alignment (bottom).

Manifold capacity Capacity is a positive scalar measure, that measures how separable the underlying category manifolds are, with higher capacity values corresponding to higher degree of separability. Intuitively, it can be thought of as the "number of linearly decodable classes per dimension", quantifying how efficiently manifolds are packed in the embedding space (Gardner and Derrida, 1988; Chung et al., 2018). We provide a more mathematically detailed explanation of one interpretation of capacity in appendix A.3.2, and for full rigorous treatment, refer the reader to (Chou et al., 2024).

Geometry of individual manifolds In this work we make a few simplifications, compared to the original formulation (Chou et al., 2024). Manifold capacity is analytically expressed as a function of so called *effective* radius and dimension of manifolds, that are determined by the spatial arrangement of manifolds' anchor points, that can be thought of as support vectors for the classification problem. In the presence of correlated structure, these measures might have complicated form, not necessary corresponding to intuitive notions of radius and dimension. To bring our results into a more direct interpretation, we measure geometry in the following way instead:

1. **Dimension** of each manifold was measured as participation ratio of principal components, which roughly corresponds the number of dimensions needed to explain around 80–90% of total variance (Gao et al., 2017).

2. **Radius** of each manifold was taken to be the maximum distance between any pair of points on the manifold.

Both metrics were averaged across manifolds, each resulting in a single scalar value. We refer to these measures as geometric properties of individual manifolds, since they do not depend on the relative positions and orientations of manifolds in the embedding space.

Correlation structure Manifold capacity also depends on the spatial arrangement of individual manifolds relative to each other and to the global origin. We measure correlation coefficients between axes of variation of individual manifolds (**Axes-alignment**) and correlations between each manifold's axes and its centroid (**Center-axes alignment**). For an extended discussion of how these correlation measures affect manifold capacity in different regimes, see (Chou et al., 2024). In this work, we consider these measures collectively as *correlation structure* to explain capacity changes driven by the relative arrangements of manifolds in the embedding space, rather than by changes in individual manifold properties.

4 Results

Our analysis reveals complex dynamics in how prompting affects the internal representations of language models, with distinct patterns emerging at different processing stages and for various prompting methods.

4.1 Representational changes during text-classification task

We first investigated performance and representational effects of prompting during a conventional ICL setting, comparing demonstrations and instruction prompts.

Task performance Instruction alone achieved good accuracy, outperforming demonstration prompts with few examples (≤ 5). Larger example sets (> 5) surpassed explicit instruction, with performance quickly plateauing (fig. 3). Replacing meaningful category words (gold labels) with abstract letters required more demonstration examples to infer category nature. When category labels were consistently shuffled (e.g. "Joy" → "Anger"), the model failed to generalize beyond pretrained associations, achieving low accuracy for both target (shuffled) and original labels. This suggests that the model is not purely learning a novel task from scratch, but rather (at least partially) relies on existing associations encoded in label semantics. (Pan et al., 2023).

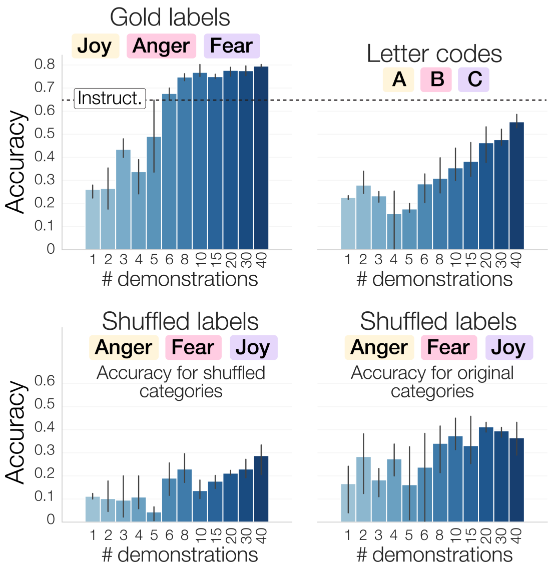

Figure 3: Performance of demonstrations and instruction prompting on sentiment analysis task.

Sentence-level effects Analysis of sentence-level embeddings (fig. 4) revealed that demonstration examples, but not abstract instruction, significantly reorganized intermediate representations at early-mid layers. This reorganization increased the separability of sentiment manifolds by reducing manifold dimension and improving correlation structure (see fig. 15). Surprisingly, there was little difference in resulting geometry between demonstrations across three labeling strategies, indicating that sentence representation is primarily influenced by input distribution examples, rather than input-output mapping.

Last-token effects At the last-token level, instruction prompts significantly increased manifold capacity relative to raw sentences, with effects emerging as early as layer 8 and persisting to final layers (fig. 5). Geometrically, the increased separability was driven mostly by the reduction in dimension along with correlation structure (supplementary fig. 19). Demonstrations further increased manifold capacity compared to instruction, despite lower task performance for cases with few examples. This suggests that while instruction alone achieves better accuracy due to high alignment between the model's readout and category man-

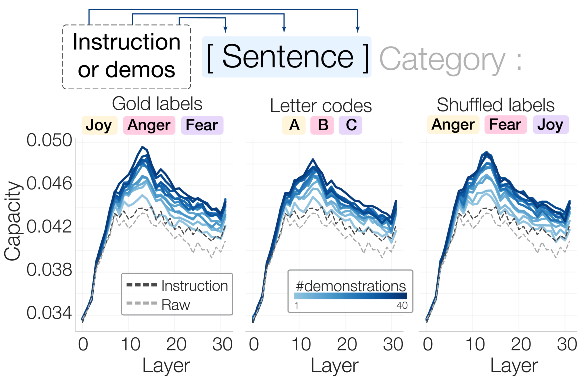

Figure 4: Manifold capacity of **sentence-level** embeddings during demonstrations prompting compared to instruction and raw sentence control

ifolds, demonstrations improve both readout alignment and representation structure. Even just for a couple of demonstrations, the underlying representation is already more optimal compared to instruction-prompted case, but this separability is not utilized properly by the unembed layer. Notably, last-token capacity during letter code labeling was much lower compared to category words, explaining lower performance when output labels lack meaningful semantics. For shuffled labels capacity values were similar to the gold label setting, suggesting that model's inability to overwrite existing associations is explained by the readout misalignment, while the underlying representation is intact.

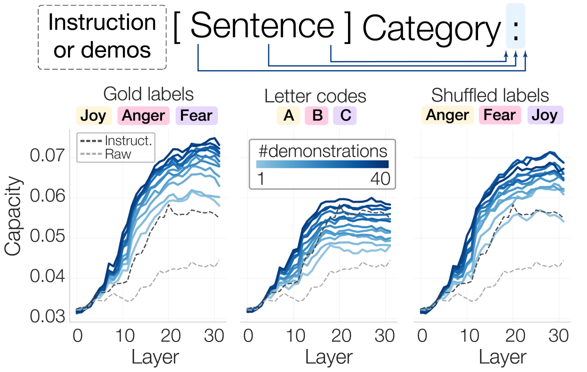

Figure 5: Manifold capacity of **last token** embeddings during demonstrations prompting compared to instruction and raw sentence control.

Sensitivity to the choice of demonstrations Performance of few-shot ICL has been previously reported to depend heavily on the choice of particular examples and their ordering, even for a fixed number of demonstrations provided (Zhao et al., 2021). To investigate whether such failure modes of certain training sets stem from changes in the underlying geometry, we analyzed the relationship between last-token manifold capacity and end performance across multiple random samplings of demonstrations, while keeping the number of examples fixed. In accordance with prior work, we observed large variance in performance (fig. 6 left), particularly in settings with fewer examples. For instance, with five demonstrations, accuracy varied dramatically from below 0.1 to approximately 0.6. Despite this substantial performance variability, the changes in manifold capacity of the last token embedding at the final layer were minimal. Even in "failure" runs with lowest accuracy, manifold capacity was significantly higher than in the instruction setting, and the layer-wise profile of capacity in the worst runs was nearly identical to the best runs (fig. 6 right). These results provide further evidence that the instability of few-shot ICL and its sensitivity to particular examples is driven primarily by poor readout alignment rather than differences in representational geometry

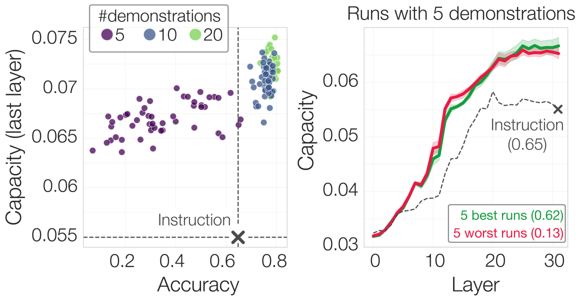

Figure 6: Left: Manifold capacity at the final layer versus accuracy for individual ICL runs with different numbers of demonstrations compared to instruction. Right: Layer-wise capacity profiles for the five best- and worst-performing 5-demonstration runs compared to instruction (accuracies shown in brackets)

4.2 Cross-Task Interactions in Multi-Task Prompting

Multi-task setup We then investigated whether prompting a model to perform one task would affect the quality of representation for another unrelated task. To this end, we constructed an artificial sentence-classification dataset containing three independent sets of labels, each with five categories. This design allowed each sentence to be classified by its sentiment, topic, or intent (see appendix A.1 for details).

We created instruction and demonstration prompts for each of the three tasks. We then computed representational metrics for each of the three possible sets of manifolds, resulting in nine possible pairs between a prompt and a representation. We termed the three cases where the manifold-inducing labels coincided with the classification objective (e.g., performing sentiment analysis and computing separability of sentiment manifolds) as **coherent**. The remaining six cases, where manifold capacity was evaluated for a different set of labels, were termed **incoherent**.

This setup allowed us to explore how prompting for one task affects the model's internal representations not just for that task, but also for other potential tasks on the same input. All experiments used gold category labels, and manifold metrics for each configuration were normalized by the corresponding value in the raw sentence case.

Synergistic Effects at the Sentence Level Increasing the number of demonstrations robustly led to increased manifold capacity at intermediate layers for coherent configurations, while instruction had a much weaker effect (fig. 7). Surprisingly, demonstrations for an incoherent task also increased capacity with a similar layerwise profile, albeit to a lesser extent. This highlights the role of input distribution: providing example sentences enhances representation capacity for supporting other tasks on the same input distribution, even in the context of a different task. While the trend of increased capacity with growing number of examples was similar for both coherent and incoherent scenarios, the amplitude of such increase was larger when the task was coherent with the manifold labels. Notably, while the overall trend is captured by the decrease in manifold dimension, the difference between coherent and incoherent settings is not fully explained by the geometry of individual manifolds (supplementary fig. 21). Instead, it likely arises due to changes in the correlation structure and relative positions of manifolds in the embedding space.

Task Interference in Last Token Representations Analysis of last token embeddings revealed an interesting dichotomy of layerwise dynamics (fig. 8). At earlier layers, additional demonstrations of incoherent tasks increased manifold capacity, but at later layers, this trend reversed, with additional examples decreasing capacity. Coherent demonstrations significantly increased capacity starting with layer 12 and persisting to the final layer. The increase in capacity driven by coherent prompts at intermediate layers was much more prominent, compared to incoherent prompts, indicating a larger role of task-specific input-output pairings. Decrease in capacity at final layers with growing number of demonstrations suggests an intriguing idea of representational tradeoff: as the model prepares the output, features for irrelevant tasks, that were emphasized at intermediate processing stages are compromised in favor of better separability of task-relevant features. Interestingly, this effect could not be explained by the geometry of individual manifolds — we observed a reduction in dimension with increased number of examples for both coherent and incoherent tasks. Instead, we observed that center-axes correlations behaved differently for coherent and incoherent cases, capturing the trend in capacity (see supplementary fig. 22).

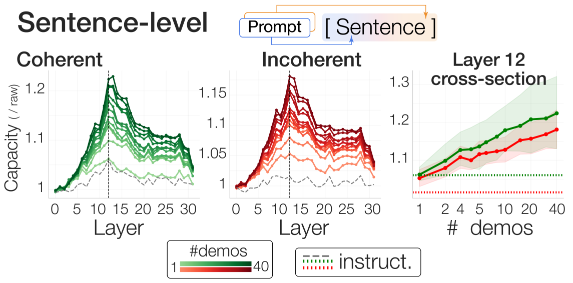

Figure 7: Effect of prompting in a multitask setting at sentence-level

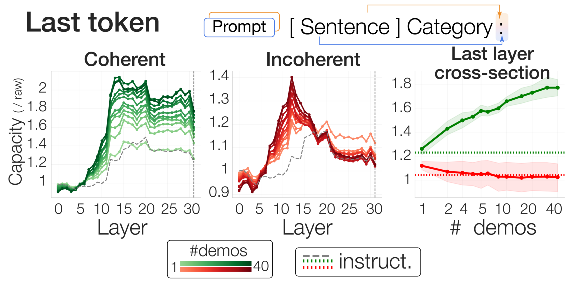

Figure 8: Effect of prompting in a multitask setting at last-token level

4.3 Distinct representational mechanisms of prompt-tuning

Performance and Setup of Soft Prompts Finally, we extended our investigation to **prompt-tuning**, an alternative method of task adaptation (Lester et al., 2021) that optimizes a task-specific prompt directly in the embedding space (hence "soft"), which is prepended to the test input (fig. 9

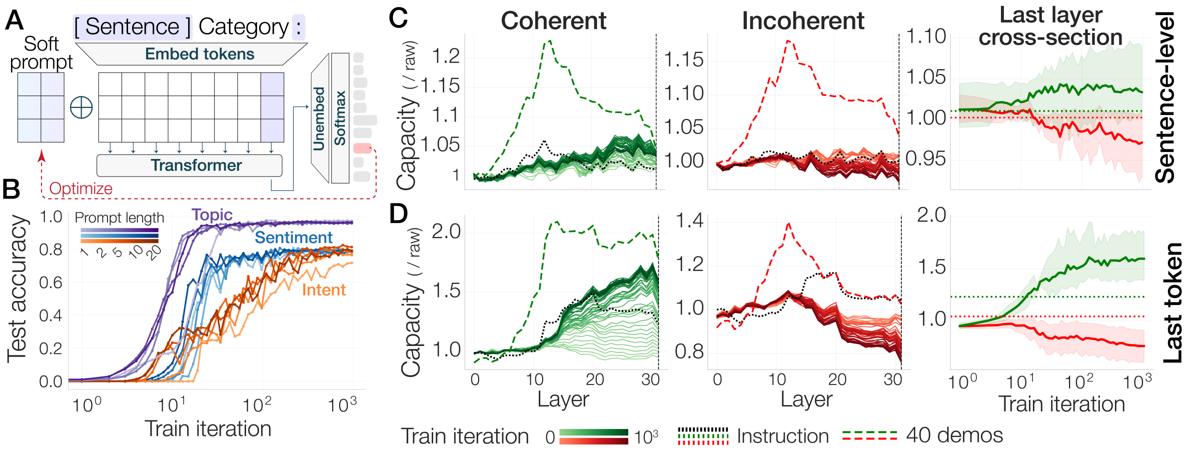

Figure 9: Schematic of the prompt-tuning setup (A) and performance at various tasks for different lengths of the soft prompt (B). Right: Manifold capacity changes during training across layers for sentence-level (C) and last token (D) representations

A). This approach allowed us to examine whether gradient-based methods for task adaptation affect internal representations similarly to traditional prompting methods.

To validate performance, we trained separate soft prompts of varying lengths for all three tasks, computing test accuracy across intermediate training iterations (fig. 9, B). Notably, soft prompts consistently outperformed demonstrations ((see A), and we found no significant correlation between final performance and prompt length, consistent with prior work (Lester et al., 2021). Given the similarity in performance and representational effects across different prompt lengths, we present representative plots for a 5-token soft prompt in the following analysis.

Minimal Impact on Intermediate Representations Analysis of sentence-level representations during soft prompt training revealed a striking mechanistic difference compared to hard natural-language prompts (fig. 9 C). The optimization-based solution did not alter intermediate representations in earlier layers, as illustrated by the absence of the characteristic peak around layer 12 observed with other prompting methods. Instead, effects were concentrated in later layers, where we observed a similar representational trade-off even at the sentence level: prompts for incoherent tasks led to decreased capacity. Importantly, this capacity difference could not be explained by the geometry of individual manifolds, suggesting the critical role of relative manifold arrangement (see fig. 25).

Enhanced Trade-off in Last Token Representations At the last token level, prompt-tuning also exhibited distinct effects. For coherent prompts, manifold capacity increased substantially in later layers as training progressed, surpassing instruction-prompted capacity but remaining below that induced by demonstrations. Notably, this effect emerged at later layers compared to both instruction and demonstration methods. In the incoherent case, soft prompts dramatically reduced the capacity of representations for unrelated tasks. This suggests that gradient-based input optimization compromises the representation of task-irrelevant features even more than natural-language demonstrations. As with demonstrations, the capacity difference between coherent and incoherent settings was primarily attributable to the relative alignment of manifolds in the embedding space, rather than geometry of individual manifolds (fig. 26).

Taken together, our results on prompt-tuning indicate that soft-prompts, often proposed to be alternative to demonstration-based ICL, operate through fundamentally different internal mechanisms compared to demonstrations and zero-shot instruction.

5 Discussion

Our study illuminates the mechanisms of how language models adapt to various tasks by analyzing the geometry of internal representations under different prompting methods. We found that zero-shot instruction, few-shot demonstrations, and tunable soft-prompts, while achieving comparable performance, operate through distinctly different representational mechanisms.

Zero-shot instructions, while effective, primarily influence the final stages of processing, affecting how features are "packaged" in the last token embedding without significantly altering intermediate representations. In contrast, demonstration examples have a more profound impact, reshaping intermediate representations to optimize them for the classification objective. In a multitask setting, demonstrations optimize early-layer representations to support multiple potential tasks, regardless of the specific task being demonstrated. This suggests a form of general feature enhancement triggered by exposure to diverse input examples. Soft-prompts, despite being trained on the same input distribution examples, operate differently, mainly affecting later layers responsible for output preparation, distinguishing them from the broader impact of natural language demonstrations.

A key insight emerging from our analysis is the distinction between representational geometry and readout alignment in determining model performance. Manifold capacity measures the inherent separability of category representations and their potential for supporting robust classification across all possible linear readouts. However, actual model performance also depends on a specific readout – the model's unembed layer — which may fail to optimally utilize well-structured representations. This effect manifests itself in two notable "failure modes" of few-shot ICL — dramatic sensitivity to the choice and ordering of specific examples and the inability to generalize beyond label associations in the pretraining corpus. In both cases, the internal representations remain well-organized for classification, but the unembed layer fails to effectively leverage this structure, resulting in poor accuracy. High separability, as measured by manifold capacity, suggests that one could train a simple linear readout module on top of existing representations to overcome this, leveraging the feature-extraction power of decoder-only LLMs for efficiently adapting their representations to specific tasks. The success of prompt-tuning further supports this view: its effectiveness appears to stem primarily from improving the alignment between representations and the vocabulary readout layer, rather than fundamentally altering the geometric organization of the embedding space.

These findings suggest two promising directions for future research. First, given that internal representations often maintain high manifold capacity even when ICL performance is poor, there is significant potential in better understanding and optimizing readout alignment. Quantifying decoder alignment by comparing the performance of independently trained classifiers with the model's unembed layer could provide deeper insights into this bottleneck and suggest ways to overcome it. Second, our observation that demonstrations can drastically change representational geometry suggests opportunities for more direct geometric optimization. Recent work has shown promising results in related fields: optimizing vision network parameters to directly maximize manifold separability has achieved SoTA performance (Yerxa et al., 2023), while regularizing learned embeddings to respect structural characteristics has improved performance in causal inference tasks (Balashankar and Subramanian, 2021). We anticipate that similar insights into LM representational geometry could drive innovations in language model prompting, enhancing performance and stability across a wide range of objectives.

Limitations

Our study provides insights into the representational geometry of language models under different prompting methods, but it has limitations. First, we used synthetic datasets generated by Claude 3.5 Sonnet, which allowed precise control over task parameters. However, this approach may not fully capture the complexity and variability of real-world language structure. To enhance the generalizability of our findings, future research should expand testing to include a broader range of natural datasets.

Second, the metrics used to quantify representational geometry in our study, such as manifold capacity and individual manifold geometry, though informative, simplify the more complex tasks that occur in language models, by focusing on a classification task with given target labels. Future work should examine how other tasks, such as those requiring multi-token outputs (e.g., chain-of-thought prompting), affect representational geometry. Additionally, more advanced measures that link geometry to complex computations could provide further insights into the fine-grained changes during task adaptation.

Acknowledgments

This work was funded by the Center for Computational Neuroscience at the Flatiron Institute of the Simons Foundation. S.C. is supported by the

Klingenstein-Simons Award, a Sloan Research Fellowship, NIH award R01DA059220, and the Samsung Advanced Institute of Technology (under the project "Next Generation Deep Learning: From Pattern Recognition to AI"). All experiments were performed on the Flatiron Institute's high-performance computing cluster.

References

Mostafa Abdou, Artur Kulmizev, Daniel Hershcovich, Stella Frank, Ellie Pavlick, and Anders Søgaard. 2021. Can language models encode perceptual structure without grounding? a case study in color. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 109–132, Online. Association for Computational Linguistics.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2023. What learning algorithm is in-context learning? investigations with linear models. *Preprint*, arXiv:2211.15661.

Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. 2019. Intrinsic dimension of data representations in deep neural networks.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2018. Linear algebraic structure of word senses, with applications to polysemy. *Preprint*, arXiv:1601.03764.

Ananth Balashankar and Lakshminarayanan Subramanian. 2021. Learning faithful representations of causal graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 839–850, Online. Association for Computational Linguistics.

Yonatan Belinkov. 2021. Probing classifiers: Promises, shortcomings, and advances. *Preprint*, arXiv:2102.12452.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Amanda Bertsch, Maor Ivgi, Uri Alon, Jonathan Berant, Matthew R. Gormley, and Graham Neubig. 2024. In-context learning with long-context models: An in-depth exploration. *Preprint*, arXiv:2405.00200.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. *Preprint*, arXiv:1511.06349.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Chi-Ning Chou, Luke Arend, Albert J Wakhloo, Royoung Kim, Will Slatton, and SueYeon Chung. 2024. Neural manifold capacity captures representation geometry, correlations, and task-efficiency across species and behaviors. *bioRxiv*.

SueYeon Chung and L.F. Abbott. 2021. Neural population geometry: An approach for understanding biological and artificial neural networks. *Current Opinion in Neurobiology*, 70:137–144.

SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. 2018. Classification and geometry of general perceptual manifolds. *Physical Review X*.

Uri Cohen, SueYeon Chung, Daniel D. Lee, and Haim Sompolinsky. 2020. Separability and geometry of object manifolds in deep neural networks. *Nature Communications*, 11(1):746.

James J. DiCarlo and David D. Cox. 2007. Untangling invariant object recognition. *Trends in Cognitive Sciences*, 11(8):333–341.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Preprint*, arXiv:2209.10652.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. 2018. Empirical Study of the Topology and Geometry of Deep Networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3762–3770, Salt Lake City, UT. IEEE.

Timo Flesch, Keno Juechems, Tsvetomira Dumbalska, Andrew Saxe, and Christopher Summerfield. 2022. Orthogonal representations for robust context-dependent task performance in brains and neural networks. *Neuron*, 110(7):1258–1270.e11.

Peiran Gao, Eric Trautmann, Byron Yu, Gopal Santhanam, Stephen Ryu, Krishna Shenoy, and Surya Ganguli. 2017. A theory of multineuronal dimensionality, dynamics and measurement.

Elizabeth Gardner and Bernard Derrida. 1988. Optimal storage properties of neural network models. *Journal of Physics A: Mathematical and general*, 21(1):271.

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

Wes Gurnee and Max Tegmark. 2024. Language models represent space and time. *Preprint*, arXiv:2310.02207.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. *Preprint*, arXiv:2310.15916.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. 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 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *Preprint*, arXiv:2104.08691.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Fuxiao Liu, Paiheng Xu, Zongxia Li, Yue Feng, and Hyemi Song. 2024. Towards understanding in-context learning with contrastive demonstrations and saliency maps. *Preprint*, arXiv:2307.05052.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Preprint*, arXiv:2205.05638.

Llama Team, AI @ Meta. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Jonathan Mamou, Hang Le, Miguel Del Rio, Cory Stephenson, Hanlin Tang, Yoon Kim, and SueYeon Chung. 2020. Emergence of separable manifolds in deep language representations. In *International Conference on Machine Learning*, pages 6713–6723. PMLR.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Preprint*, arXiv:1301.3781.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *Preprint*, arXiv:2202.12837.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. What in-context learning "learns" in-context: Disentangling task recognition and task learning. *Preprint*, arXiv:2305.09731.

Kiho Park, Yo Joong Choe, and Victor Veitch. 2024. The linear representation hypothesis and the geometry of large language models. *Preprint*, arXiv:2311.03658.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

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

Cory Stephenson, Jenelle Feather, Suchismita Padhy, Oguz Elibol, Hanlin Tang, Josh McDermott, and SueYeon Chung. 2019. Untangling in invariant speech recognition. *Advances in neural information processing systems*, 32.

Cory Stephenson, Suchismita Padhy, Abhinav Ganesh, Yue Hui, Hanlin Tang, and SueYeon Chung. 2021. On the geometry of generalization and memorization in deep neural networks. *Preprint*, arXiv:2105.14602.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. *Preprint*, arXiv:2212.07677.

Albert J Wakhloo, Tamara J Sussman, and SueYeon Chung. 2023. Linear classification of neural manifolds with correlated variability. *Physical Review Letters*.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2024. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. *Preprint*, arXiv:2301.11916.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. *Preprint*, arXiv:2109.01652.

Thomas Edward Yerxa, Yilun Kuang, Eero P Simoncelli, and SueYeon Chung. 2023. Learning efficient coding of natural images with maximum manifold capacity representations. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. *Preprint*, arXiv:2102.09690.

A Appendix

A.1 Dataset details

A.1.1 Multi-task dataset

Description The synthetic dataset used in this study was generated using Claude 3.5 Sonnet, a large language model developed by Anthropic. The dataset consists of sentences crafted to represent various combinations of emotions, topics, and pragmatic intents. The generation process was designed to create a diverse and balanced dataset suitable for studying representation changes in a multi-task setup. Example sentences covering all five category labels for each of the three subtasks can be found in table 1.

Generation procedure The dataset was generated through an iterative process, by cycling through three possible generation types:

- Emotion-focused: The model's goal was to generate 10 sentences (2 for each emotion), given a specific topic and intent.
- Topic-focused: The model's goal was to generate 10 sentences (2 for each topic), given a specific emotion and intent.
- Intent-focused: The model's goal was to generate 10 sentences (2 for each intent), given a specific emotion and topic.

Example of a full prompt an for emotion-focused iteration can be found in fig. 10. Prompts for other two types were phrased analogously.

To ensure diversity of resulting sentences, each prompt included a specific constraint for the sentence to be of a certain perspective (first, second or third person) and tense (present, past, future or mixed), chosen at random.

Additionally, the model was instructed to make at least one generated sentence follow a special requirement, that was sampled randomly from a pool of 18 possible requirements (table 2) during prompt construction.

The generated sentences were parsed and stored with their corresponding labels.

Post processing After generation, the dataset underwent several post-processing steps:

- Duplicate sentences were removed to ensure uniqueness
- Category labels were capitalized and ordered consistently.
- Letter codes and shuffled labels were assigned to each sentence for alternative labeling schemes
- Dataset was subsampled to 1000 sentences (500 for train and 500 for test splits) in a way to ensure uniform coverage of each of three subtasks (in each split: 100 sentences per category)

To introduce further variations, the following transformations were applied to both train and test sets:

- 10% of sentences were converted to lowercase
- 10% of sentences were converted to uppercase

These transformations were applied randomly and independently to each set.

A.1.2 Open datasets

To ensure our findings were not solely a result of using a synthetic dataset generated by another language model, we replicated our single-task experiments using two open datasets, often used for text classification: **AG News** (Zhang et al., 2015)and **TREC (coarse)** (Li and Roth, 2002; Hovy et al., 2001). For the TREC dataset we removed the "Abbreviation" category, which had an insufficient number of samples for manifold analysis. Additionally, we created a balanced test partition with uniform representation across all categories. Resulting dataset sizes can be found in table 3.

You are a helpful assistant tasked with generating a dataset of sentences. Generate 2 sentences for each of the following categories of emotion:
1. Joy
2. Sadness
3. Anger
4. Fear
5. Surprise

Please make sure all sentences are related to the topic "**technology**" and have **humorous** intent (**Each sentence is intended to be funny or amusing, often through clever use of language, unexpected connections, or playful exaggeration.**).

There are a few requirements for the sentences:
Use **first-person** perspective.
Use **future** tense.

Additionally, include at least one sentence that:
- **Sounds like a tweet**

Very important instructions:
1. Convey the emotion through the situation, word choice, and tone. Do not directly state the emotion or use immediate synonyms.
2. Imply the topic through context and content, but do not explicitly mention the topic name.
3. Express the intent naturally without explicitly stating the type of intent being used.

Format your response as follows:
Joy:
1. [Sentence 1]
2. [Sentence 2]

Sadness:
1. [Sentence 1]
2. [Sentence 2]

Anger:
1. [Sentence 1]
2. [Sentence 2]

Fear:
1. [Sentence 1]
2. [Sentence 2]

Surprise:
1. [Sentence 1]
2. [Sentence 2]

Ensure each sentence is on a new line and numbered within its category.
Do not include any additional text or explanations outside of this format.
Very important: Remember to vary the syntax and structure of the sentences to make the dataset diverse and interesting! Do not use the same structure for all sentences.

Figure 10: Example prompt configuration used in generating the synthetic dataset (emotion-focused type). Text highlighted in **bold** represents parts of the prompt that were varied on each iteration to increase diversity of resulting sentences.

Sentence	Sentiment	Topic	Intent
This concert has me over the moon - I'm having the time of my life!	Joy	Entertainment	Idiomatic
You'll watch as the winds of change blow away the last remnants of your faith in the system!	Sadness	Politics	Metaphorical
Tomorrow's blood test results might reveal something I'm not prepared to handle.	Fear	Health	Literal
My favorite team's strategy of constantly fumbling the ball was clearly the path to victory.	Anger	Sports	Sarcastic
I nearly fell out of my chair when my ancient printer suddenly sprang to life and started spewing out pages of binary code!	Surprise	Technology	Humorous

Table 1: Example sentences from the synthetic multi-task dataset

Sounds like a tweet
Describes a hypothetical scenario
Uses simple vocabulary as if spoken by a child
Has a rhythmic or lyrical quality
Sounds like a memorable quote
Includes a question
Includes a command or instruction
Incorporates a well-known saying or proverb
Structured like a headline
Includes a number or statistic
Imitates casual online comment style
Uses formal language
Starts with a gerund (-ing word)
Includes a rhetorical question
Uses the passive voice
Includes a list or enumeration
Employs repetition for emphasis
Starts with a conditional (If...)

Table 2: Possible special requirements during dataset generation

Dataset	Samples per category	Category labels
TREC coarse	65	Description, Entity, Human, Location, Numeric
AG news	63	Business, World, Sports, Sci/Tech

Table 3: Parameters of open dataset subsampling sizes used in experiments

A.2 Models

All experiments presented in the main text were performed on Llama3.1 8b base model (Llama Team, AI @ Meta, 2024) (32 layers, 4096 embedding dimension). We also repeated the results with Gemma2 (2b base model) (Gemma Team, 2024) (26 layers, 2304 embedding dimension). Results are presented in the appendix A.5.

A.3 Methods

A.3.1 Embedding Extraction Methodology

Challenges in Decoder-Only Models

1. **Masked Self-Attention:** In decoder-only models, each token's embedding is limited to information from itself and preceding tokens. This requires the model to progressively accumulate and propagate relevant contextual information along the sequence, influencing how global features are represented across different token positions.

2. **Distributed Sentence-Level Features:** Unlike models with dedicated [CLS] tokens, global sentence-level features (such as sentiment) might be distributed across embedding vectors of intermediate tokens.

3. **Last Token Dependency:** The model's output is a function of the last token's embedding vector only, implying that task-relevant features must be aggregated and represented in this final embedding for good task performance.

Sentence Embeddings To examine how task-specific prompts influence feature extraction and computation on intermediate tokens, we construct sentence embeddings as follows:

1. We extract residual stream activations at each layer for tokens corresponding only to the input sentence, excluding the task prompt itself. This ensures that the resulting embeddings for each sentence are of the same length across different prompting conditions.

2. We perform mean-pooling across these embedding vectors to obtain a fixed-size embedding for each sentence.

While this method differs from using dedicated sentence-level embeddings, it provides insight into the model's intermediate processing stage. Based on the idea of feature superposition, we hypothesize that directions in the embedding space corresponding to task-irrelevant token-level features will be averaged out, while task-relevant global features (which might be distributed among various tokens in the sentence) will be preserved or enhanced through mean-pooling.

Last-token Embeddings While mean-pooled embeddings allow us to capture an intermediate processing stage, the underlying sentence tokens are not immediately utilized for the task. To understand the model's final representation before output generation, we also extracted residual stream activation of the last token in the sequence at each layer. The last token is special because, for the model to perform the task, all relevant sentence-level features must get "packaged" into the embedding vector of the last token via self-attention. By analyzing last token embeddings across layers, we can track at what point such feature repackaging takes place to collect information about the sentence.

A.3.2 Manifold Capacity

This section provides additional background on the idea of manifold capacity. Consider a set of N points in D_{emb}-dimensional space: $\vec{x}_i \in \mathbb{R}^{D_{\text{emb}}}$. Each point has a corresponding class label $l_i \in \{1, \dots P\}$. Capacity measures how well a particular representation supports linear separability of a random one-vs-rest label dichotomy that doesn't break category boundaries. Namely, for P classes there are P possible dichotomies: $\{y_i^\mu\}$, where $i \in \{1, \dots N\}$ – index of a data point, $\mu \in \{1, \dots P\}$ – index of a dichotomy, and:

$$\begin{cases} y_i^\mu = 1 \text{ if } (l_i = \mu) \\ y_i^\mu = -1 \text{ otherwise} \end{cases}$$

Consider performing a random projection of data into a D_{proj}- dimensional space, where $D_{\text{proj}} \leq D_{\text{emb}}$. We can compute a probability that a randomly chosen dichotomy will be linearly separable, when the data is projected randomly to D_{proj} dimensions, formalized as follows:

$$F(D_{\text{proj}}) = \Pr_{\substack{S \sim \mathcal{N}^{(D_{\text{proj}}, D_{\text{emb}})} \\ \mu \sim \text{Unif.}(\{1 \dots P\})}} [\exists \vec{w} : \, y_i^\mu \vec{w} S \vec{x}_i \geq 0 \, \forall i]$$

Where $\vec{w} \in D_{\text{proj}}$. In a thermodynamic limit of $N, P \to \infty$, $F(D_{\text{proj}})$ undergoes a sharp phase transition from 0 to 1 as D_{proj} interpolates between

0 and D_{emb}. In the finite data case, the transition is smooth, but we can still detect an approximate critical dimension D^*, that corresponds to the inflection point of $F(D_{\text{proj}})$. Then, manifold capacity α is defined to be

$$\alpha = \frac{P}{D^*}$$

Intuitively it captures decoding efficiency, quantifying how many dimensions are sufficient for a downstream readout to perform classification. α depends on the geometry of individual manifolds (such as radius and dimension), as well as relative positioning and alignment of different class manifolds in the embedding space.

A.3.3 Manifold dimension

We use the participation ratio (PR) as a proxy for manifold dimensionality, as described in (Gao et al., 2017). For a manifold $\mathbf{X} \in \mathbb{R}^{(N,D)}$ consisting of N points in a D-dimensional space ($N \leq D$), the participation ratio is defined as:

$$\text{PR} = \frac{\left(\sum_i^N \lambda_i\right)^2}{\sum_i^N \lambda_i^2}$$

where λ_i is the i^{th} eigenvalue of the manifold covariance matrix $\mathbf{X}\mathbf{X}^T$. Intuitively, PR measures how evenly the total variance is distributed among individual principal components. Lower values of PR indicate a more rapid decay of covariance eigenvalues, signifying lower effective dimensionality. We compute the PR for each manifold and then average these values to obtain a single measure of dimensionality for the entire representation.

A.3.4 In-context learning

Demonstration Prompts We constructed demonstration prompts by randomly sampling sentences from the training split. The number of examples varied from 1 to 40, ensuring as uniform a label coverage as possible. For instance, in a 4-category classification task with 10 demonstration examples, 8 examples were guaranteed to cover all categories equally (2 per category), with the remaining 2 examples randomly chosen. We computed the forward pass of the model with 3 random seeds for each number of demonstrations and reported averaged measures across these runs.

Accuracy Measurement We measured accuracy as the proportion of test sentences for which the token with the highest logit value corresponded to the first token of the target output (for cases where the target label was tokenized into multiple tokens). Importantly, we considered logits for the entire vocabulary, not restricting the scope to target outputs. If the highest probability output was a token not corresponding to any class label, the run was treated as incorrect, irrespective of relative logit values of other tokens.

A.3.5 Prompt-tuning

Description We replicated our main experimental setup, replacing natural language task instructions and demonstrations with tunable prompts of varying lengths. A tunable prompt $\mathbf{X}$ (also referred to as a soft prompt) of length l is a matrix in the model's embedding space $\mathbb{R}^{l \times D_{\text{emb}}}$, where D_{emb} is the dimensionality of the model's token embeddings.

Unlike discrete text prompts, these tunable prompts are continuous vectors that can be optimized directly through gradient descent. They provide a more flexible way to convey task-specific information to the model, unconstrained by the token embedding matrix. This allows them to occupy highly specific regions of the embedding space that are inaccessible through natural language input alone.

Soft-prompt methodology

1. **Initialization**: Each tunable prompt $\mathbf{X}$ was initialized using the embedding vector of the word "Category". For soft prompts with $l > 1$, this embedding vector was repeated l times along the sequence length dimension, providing a starting point for optimization.

2. **Prepending**: For each input sequence s (after token embedding), we prepended the tunable prompt $\mathbf{X}$ to create an augmented input:

$$\mathrm{s}_{\text{augmented}} = [\mathbf{X}; \mathrm{s}]$$

where $[;]$ denotes concatenation along the sequence length dimension.

3. **Optimization**: During training, while keeping the pretrained language model parameters fixed, we optimized the elements of $\mathbf{X}$ to minimize the task-specific loss function:

$$\mathbf{X}^* =_{\mathbf{X}} \mathcal{L}(\text{Model}([\mathbf{X}; \mathrm{s}]), \mathbf{y})$$

where $\mathcal{L}$ is the Cross Entropy loss function, $\text{Model}(\cdot)$ represents the frozen pretrained language model, and $\mathbf{y}$ is the ground truth label.

4. **Length Variation**: We trained soft prompts of various lengths $l \in \{1, 2, 5, 10, 20\}$ to investigate the impact of prompt size on performance. Longer prompts can theoretically capture more details about general task structure, the nature of categories, and meta-information about specific training examples (although in practice, we did not observe significant performance differences across different lengths).

Training procedure and checkpoints Soft prompts were optimized on the training subset of each dataset (see appendix A.1). We trained each soft prompt for 30 epochs with a batch size of 16 using the Adam optimizer (Kingma and Ba, 2017). The initial learning rate was set to 3×10^{-4} with an exponential decay of $\gamma = 0.9$ after each epoch. To analyze how representations evolved during the training of soft prompts, we selected 50 intermediate points, logarithmically spaced across training iterations.

A.4 Computational resources

All experiments were performed on a high-performance computing cluster, using Nvidia H100 GPUs, resulting in total of 1000 GPU hours.

A.5 Supplementary plots

To maintain a reasonable number of figures in the paper, we present a curated subset in this appendix, highlighting key points with representative plots. The complete set of figures, detailing geometric measures for all combinations of models, datasets, and tasks, along with the source code, will be available on GitHub. The repository will be made public upon publication.

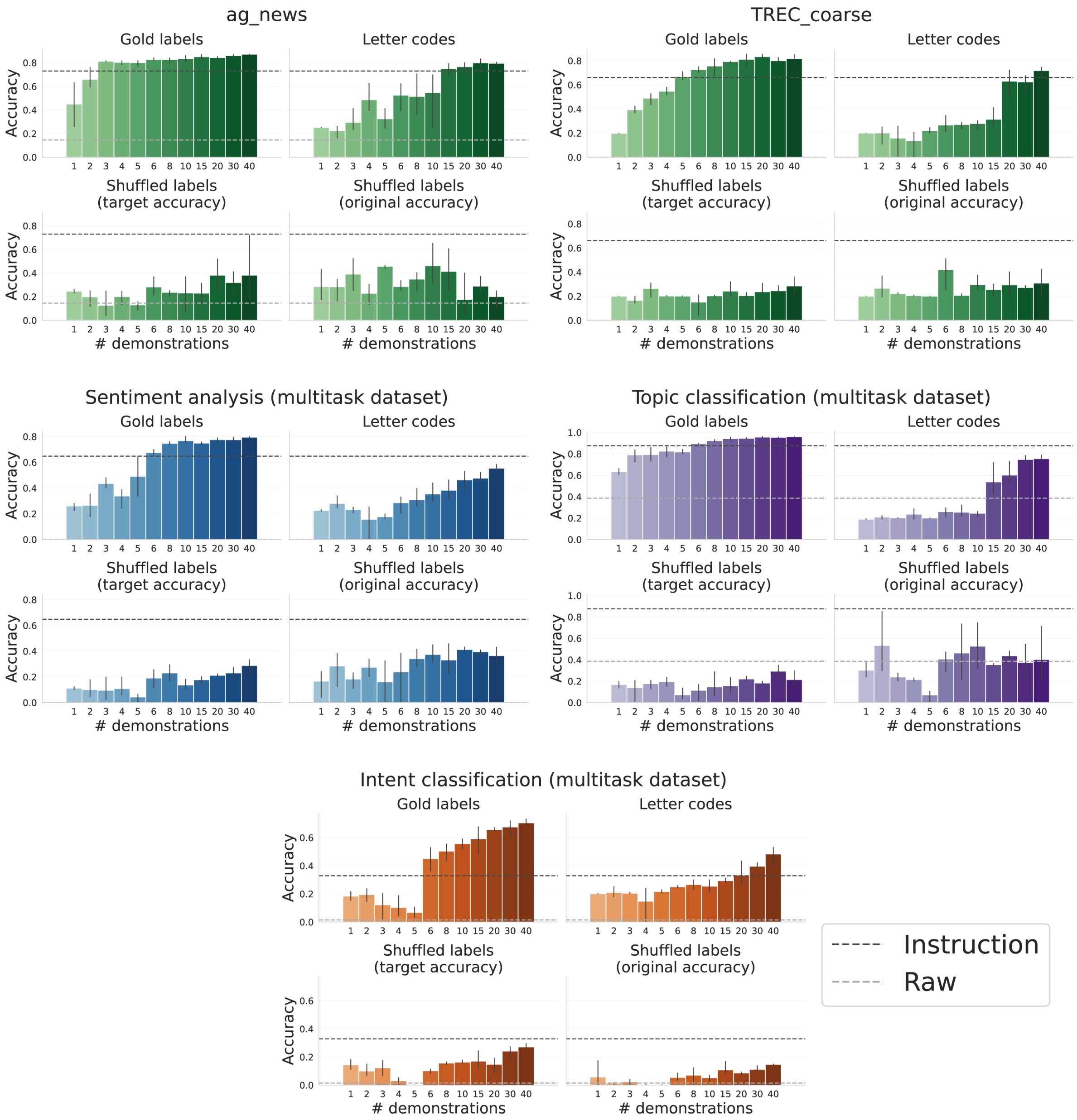

Figure 11: **Llama3.1-8b** performance of demonstrations and instruction prompts on open datasets (ag_news and TREC coarse) and on all three subtasks of the synthetically generated multitask dataset (sentiment, topic and intent).

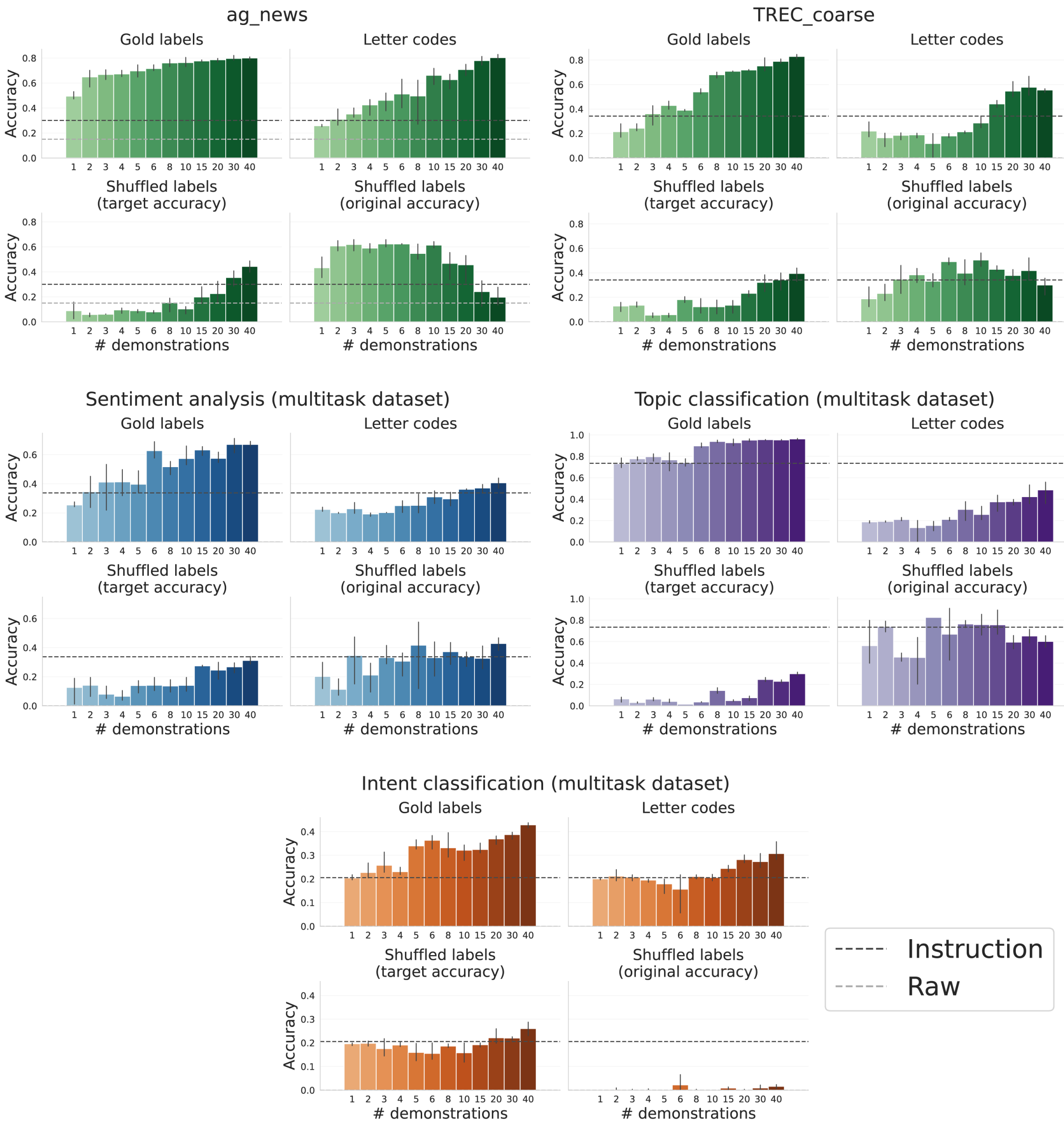

Figure 12: **Gemma2-2b** performance of demonstrations and instruction prompts on open datasets (ag_news and TREC coarse) and on all three subtasks of the synthetically generated multitask dataset (sentiment, topic and intent).

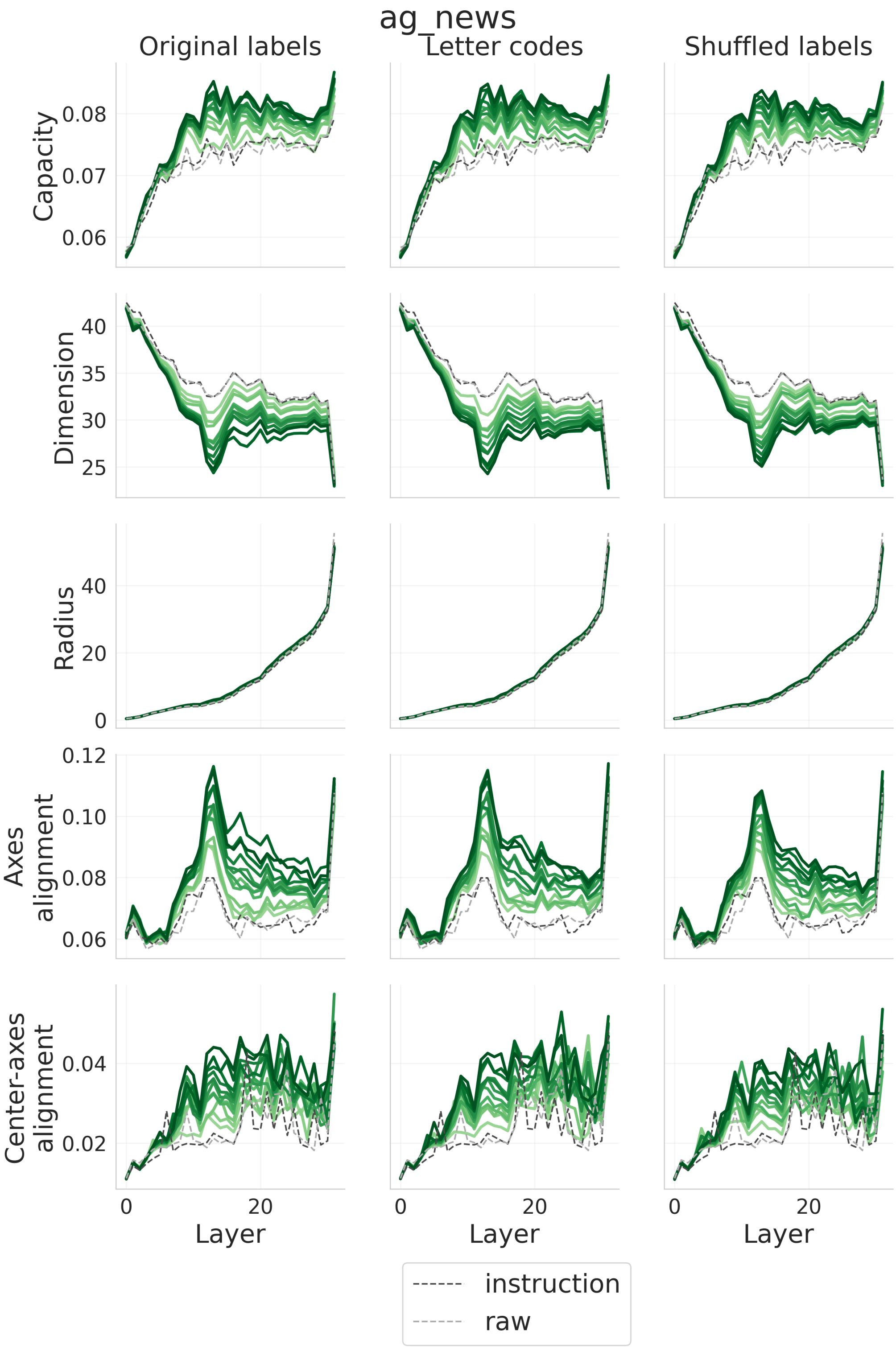

Figure 13: Manifold capacity and geometric properties of **sentence-level** representations during demonstration prompting compared to instruction and raw sentence across layers. **Llama3.1-8b** evaluated on **ag_news**. Gradient color shows number of demonstrations (darker — more examples).

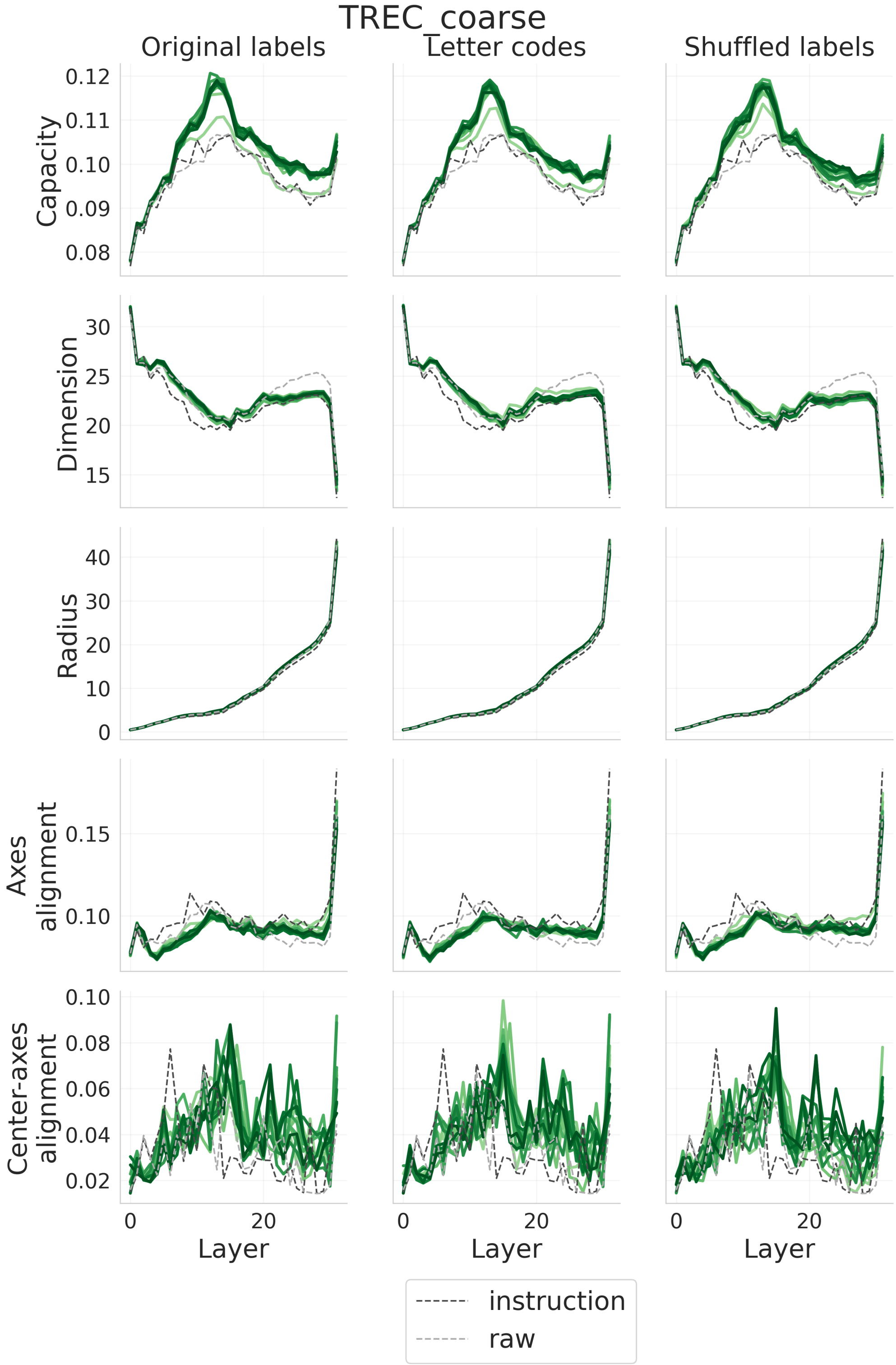

Figure 14: Manifold capacity and geometric properties of **sentence-level** representations during demonstration prompting compared to instruction and raw sentence across layers. **Llama3.1-8b** evaluated on **TREC_coarse**. Gradient color shows number of demonstrations (darker — more examples).

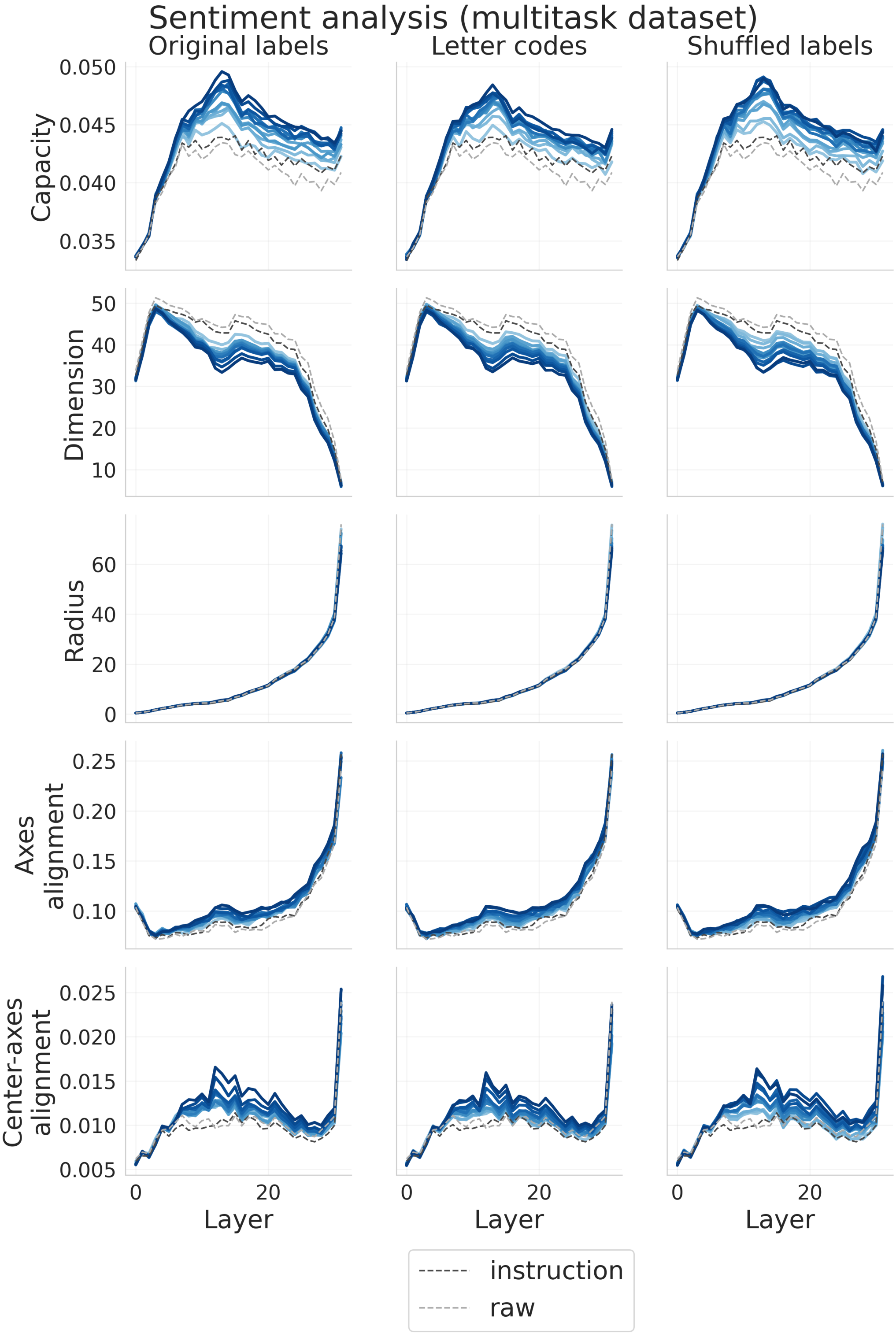

Figure 15: Manifold capacity and geometric properties of **sentence-level** representations during demonstration prompting compared to instruction and raw sentence across layers. **Llama3.1-8b** evaluated on **sentiment analysis** subtask of multitask synthetic dataset. Gradient color shows number of demonstrations (darker — more examples).

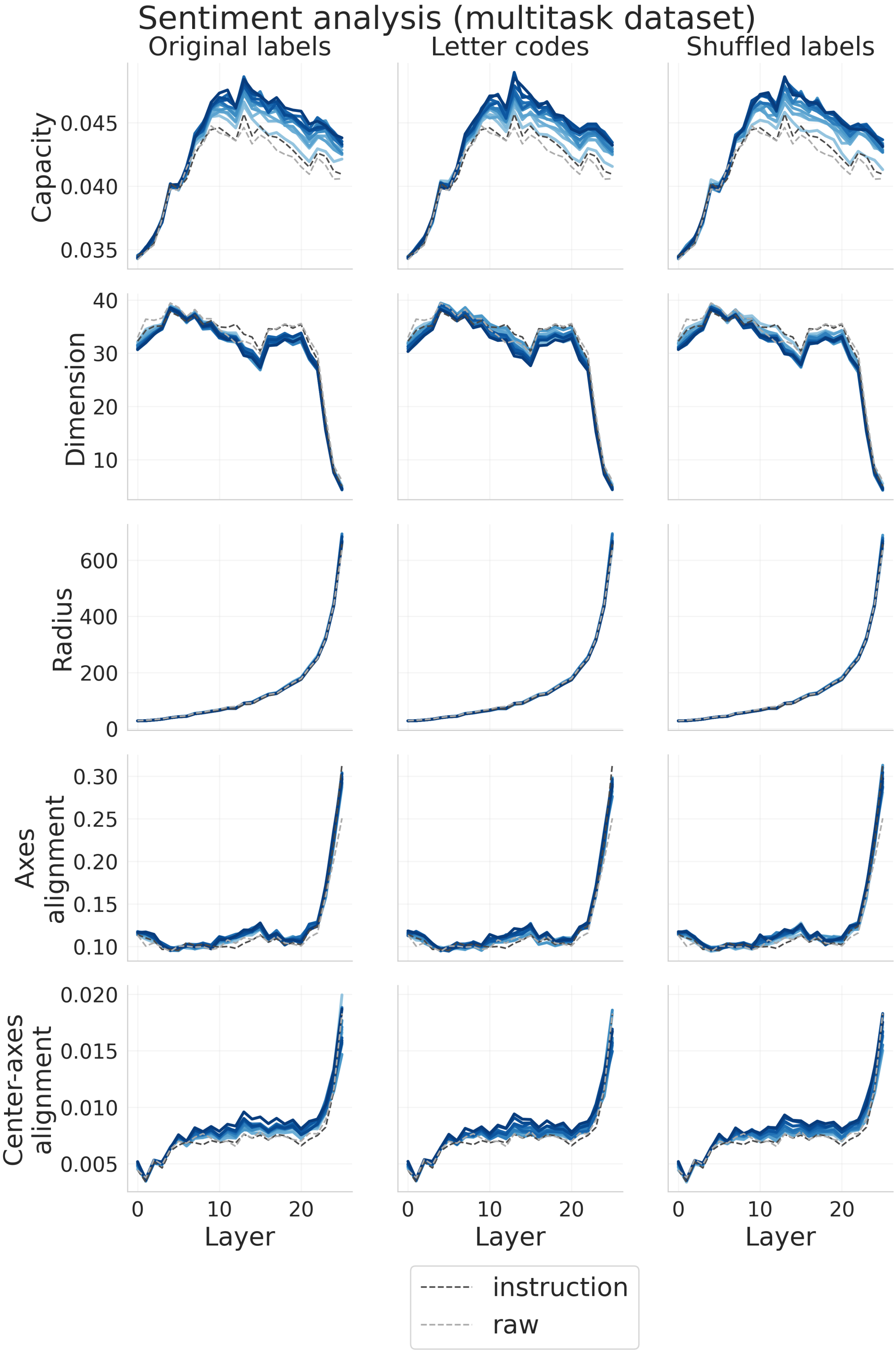

Figure 16: Manifold capacity and geometric properties of **sentence-level** representations during demonstration prompting compared to instruction and raw sentence across layers. **Gemma2-2b** evaluated on **sentiment analysis** subtask of multitask synthetic dataset.Gradient color shows number of demonstrations (darker — more examples).

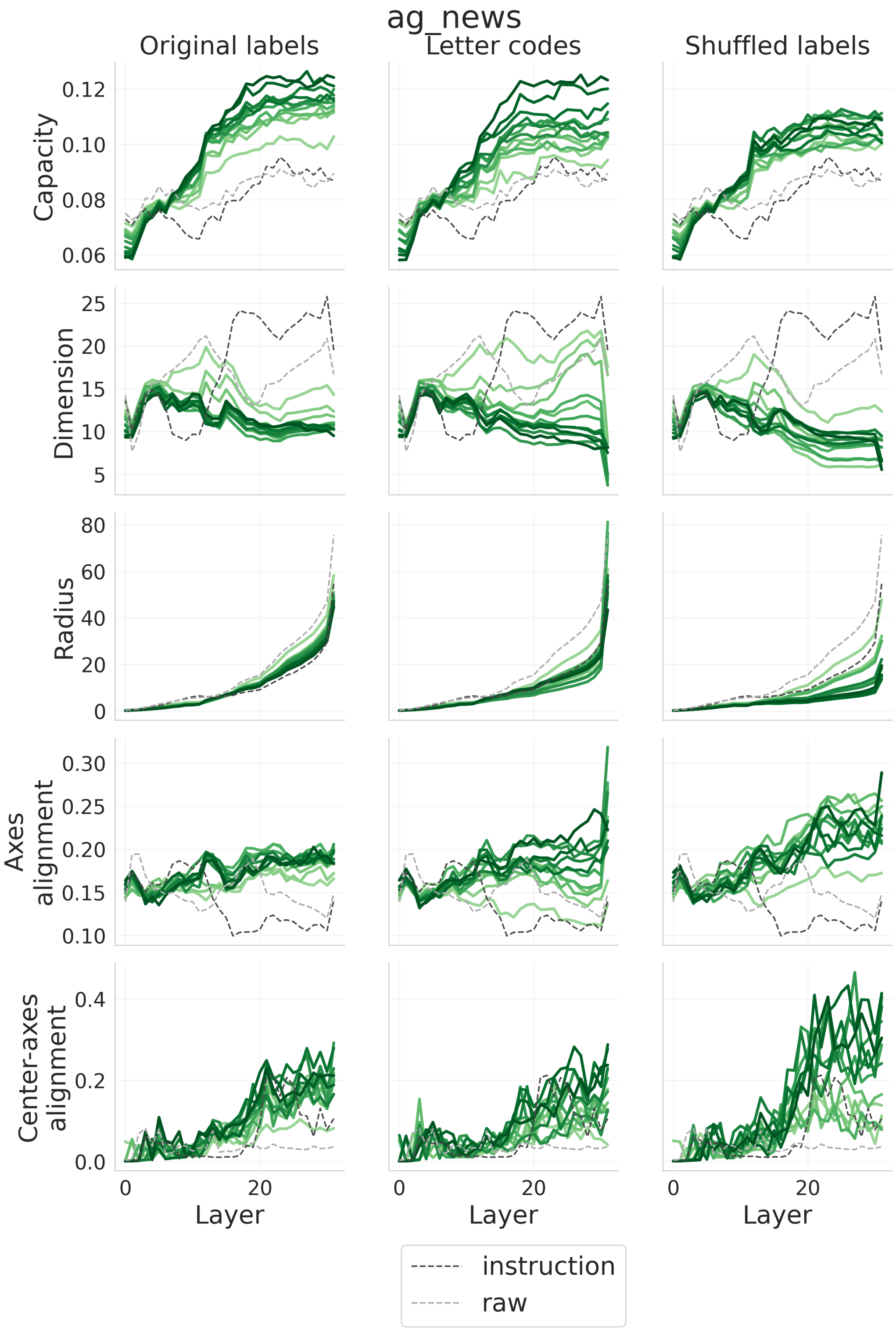

Figure 17: Manifold capacity and geometric properties of **last token** representations during demonstration prompting compared to instruction and raw sentence across layers. **Llama3.1-8b** evaluated on **ag_news**. Gradient color shows number of demonstrations (darker — more examples).

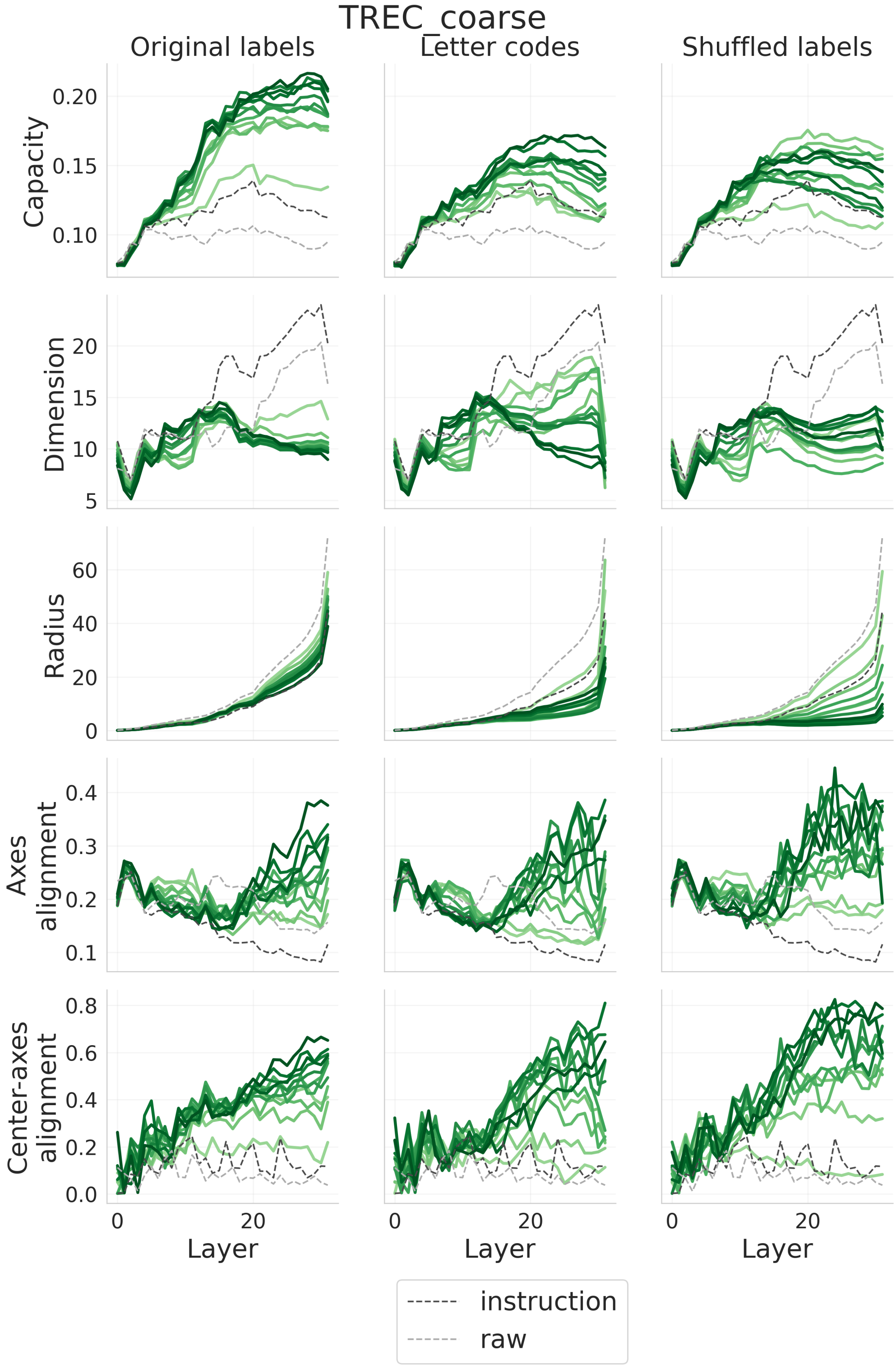

Figure 18: Manifold capacity and geometric properties of **last token** representations during demonstration prompting compared to instruction and raw sentence across layers. **Llama3.1-8b** evaluated on **TREC_coarse**. Gradient color shows number of demonstrations (darker — more examples).

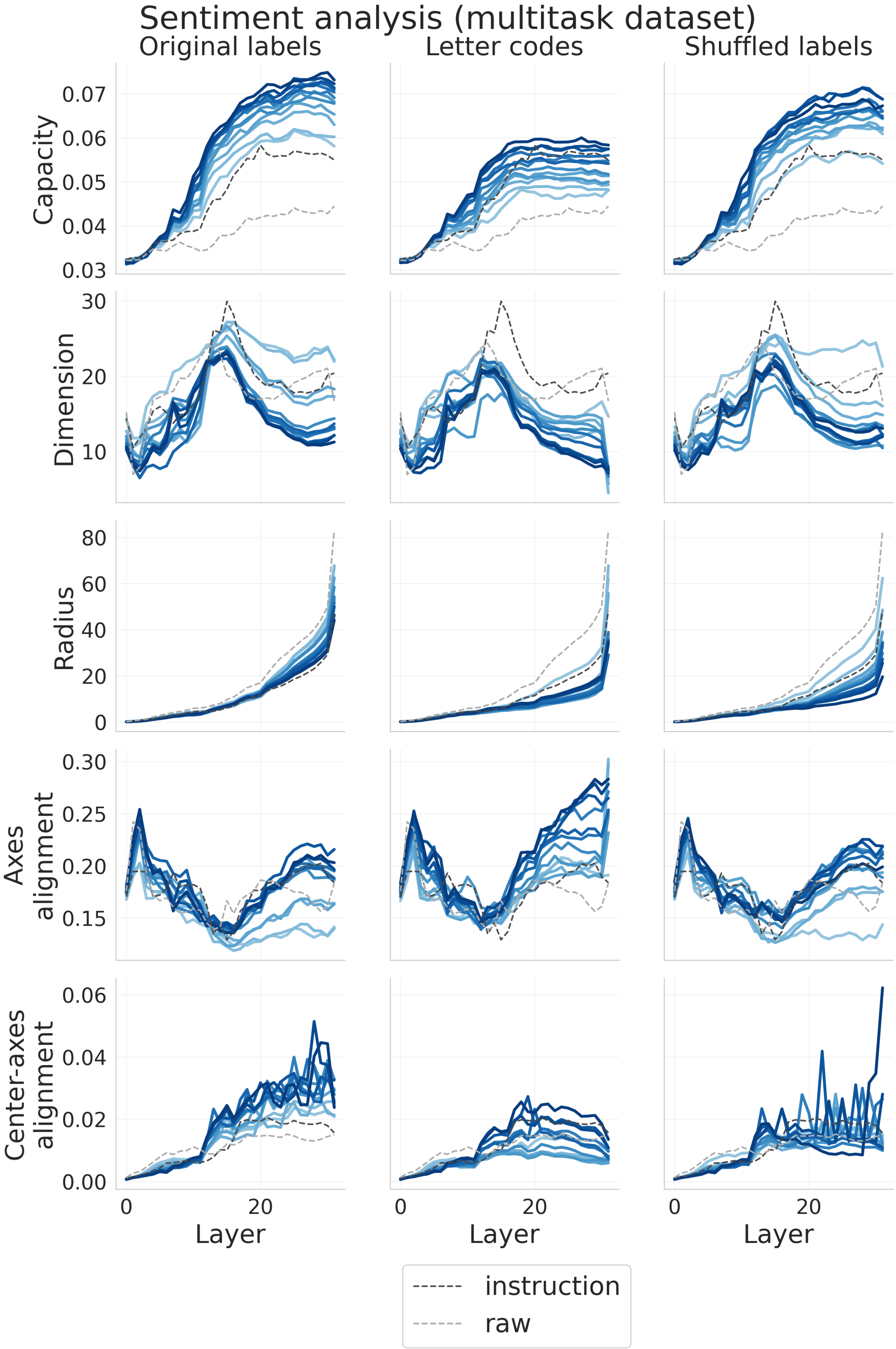

Figure 19: Manifold capacity and geometric properties of **last token** representations during demonstration prompting compared to instruction and raw sentence across layers. **Llama3.1-8b** evaluated on **sentiment analysis** subtask of multitask synthetic dataset. Gradient color shows number of demonstrations (darker — more examples).

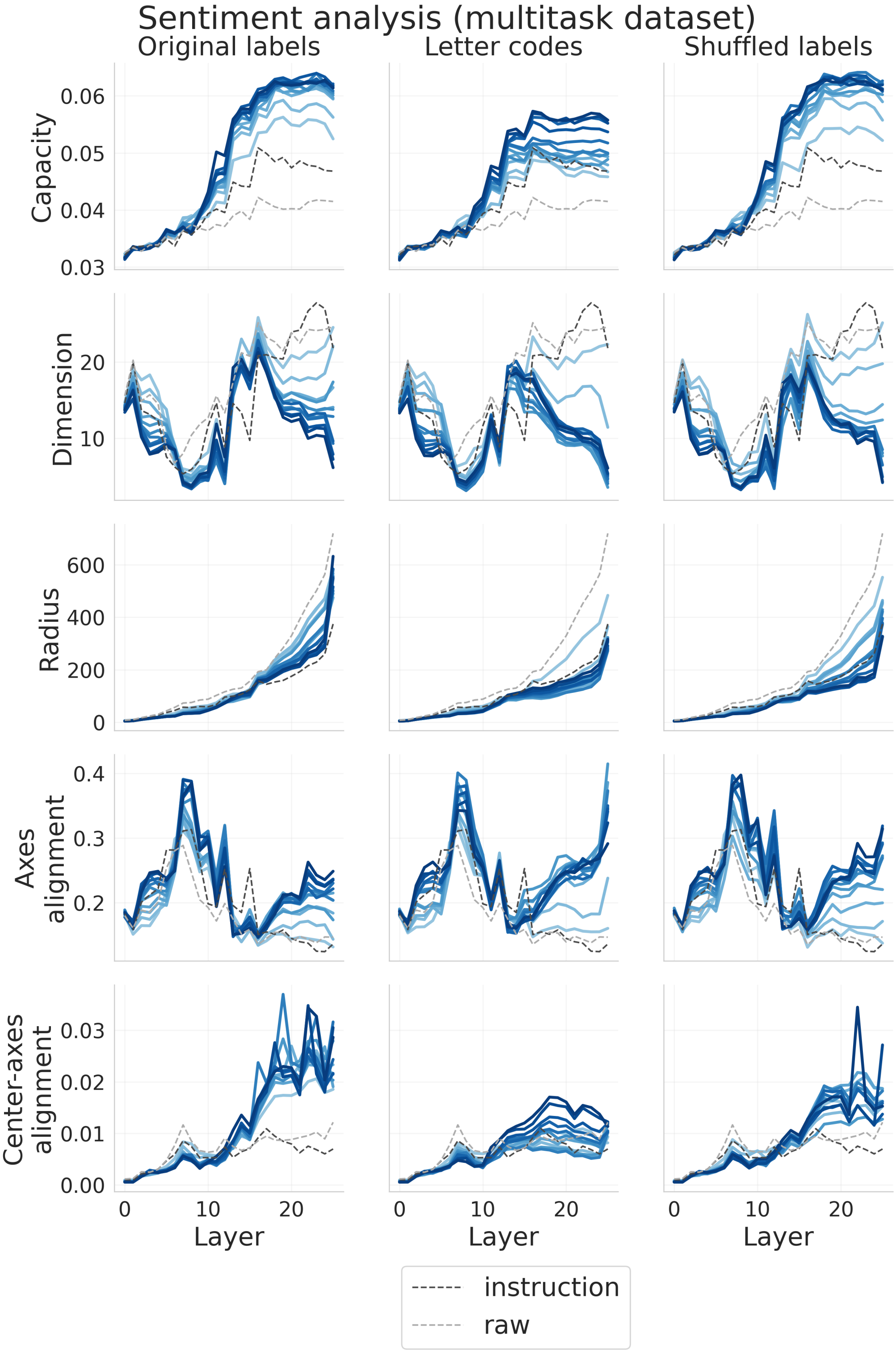

Figure 20: Manifold capacity and geometric properties of **last token** representations during demonstration prompting compared to instruction and raw sentence across layers. **Gemma2-2b** evaluated on **sentiment analysis** subtask of multitask synthetic dataset. Gradient color shows number of demonstrations (darker — more examples).

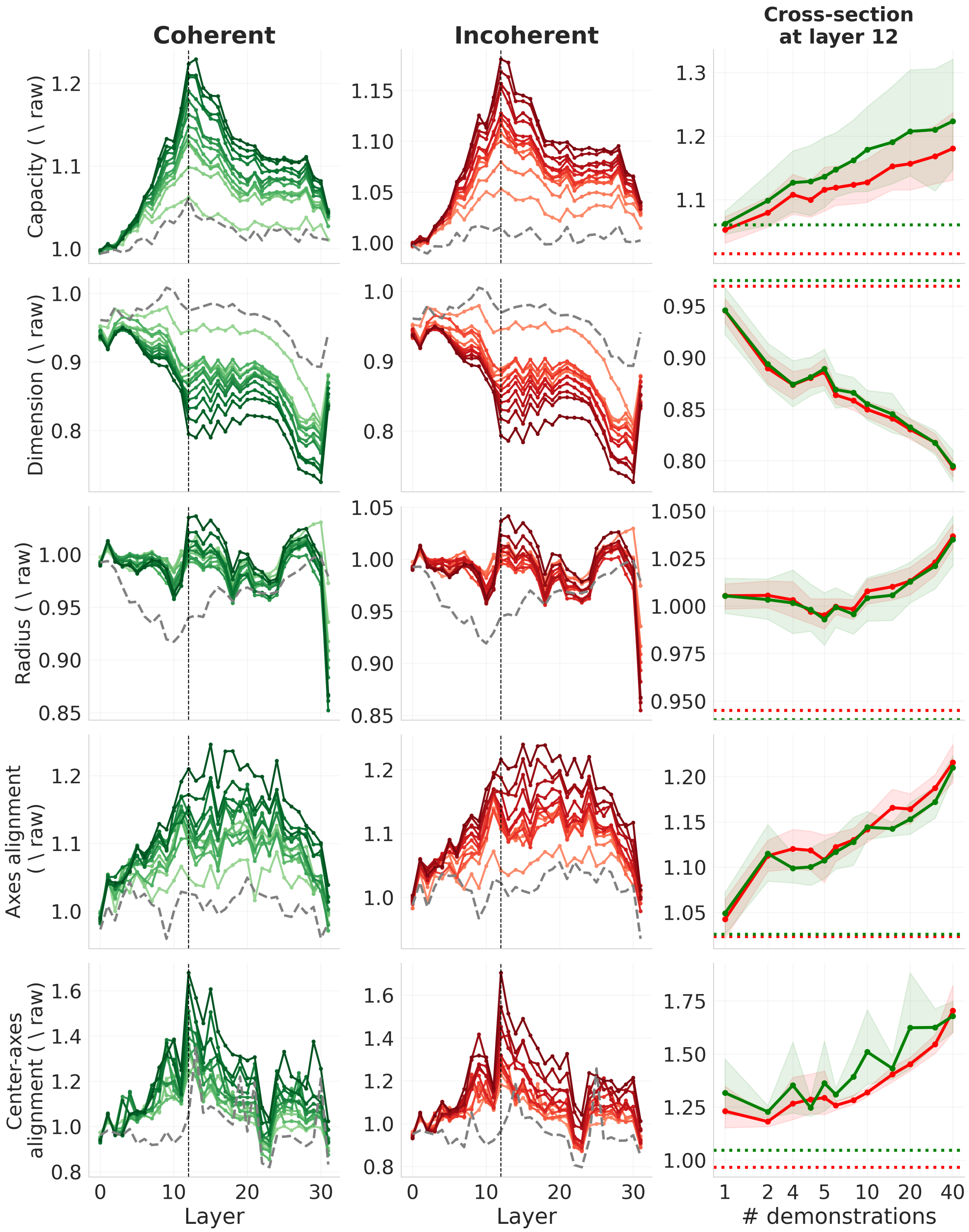

Figure 21: Geometric measures of **sentence-level** representation during coherent and incoherent task-prompting of **Llama3.1-8b**.Gradient color shows number of demonstration examples (darker — more examples). Dashed lines — instruction prompt.

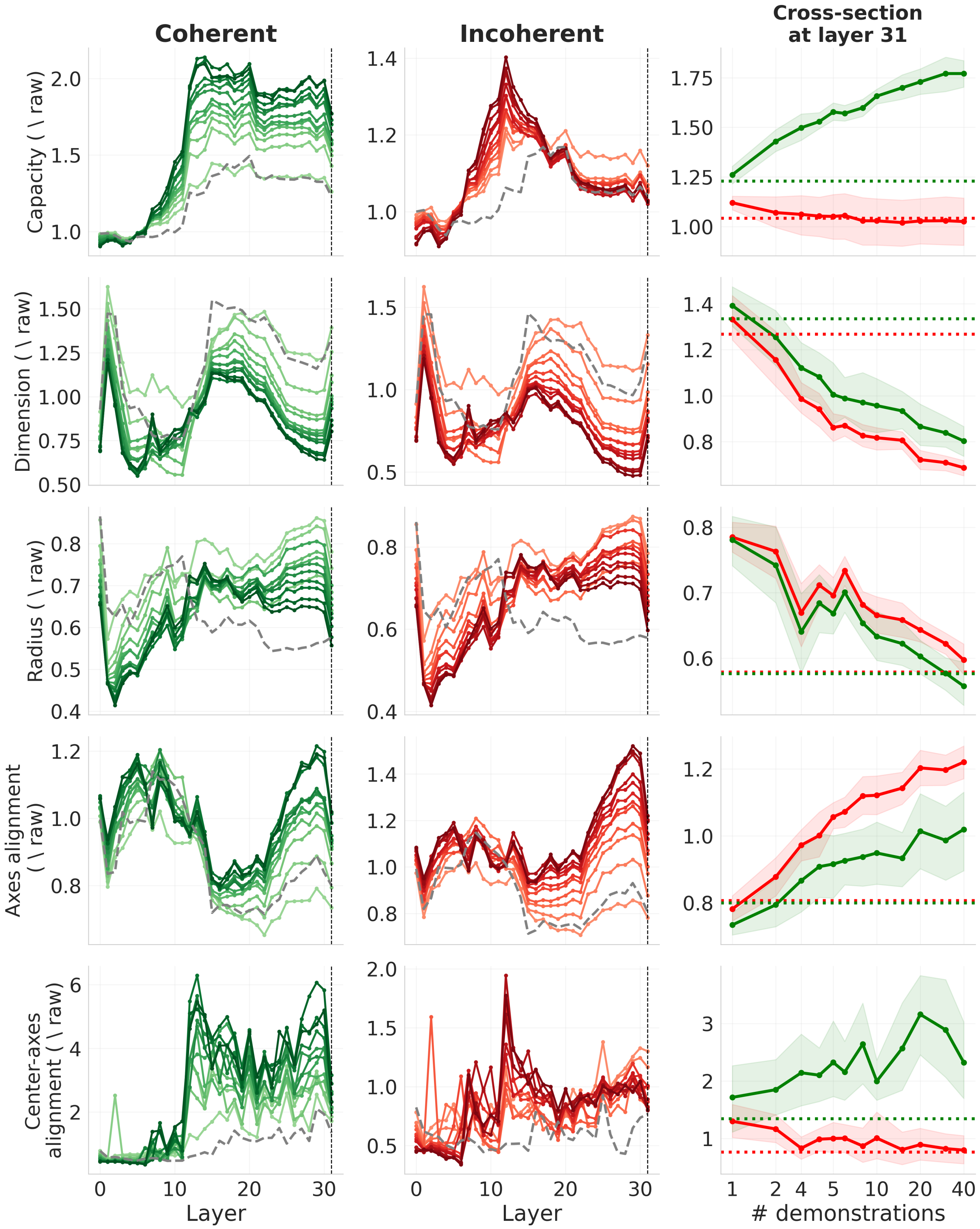

Figure 22: Geometric measures of **last-token** representation during coherent and incoherent task-prompting of **Llama3.1-8b**. Gradient color shows number of demonstration examples (darker — more examples). Dashed lines — instruction prompt.

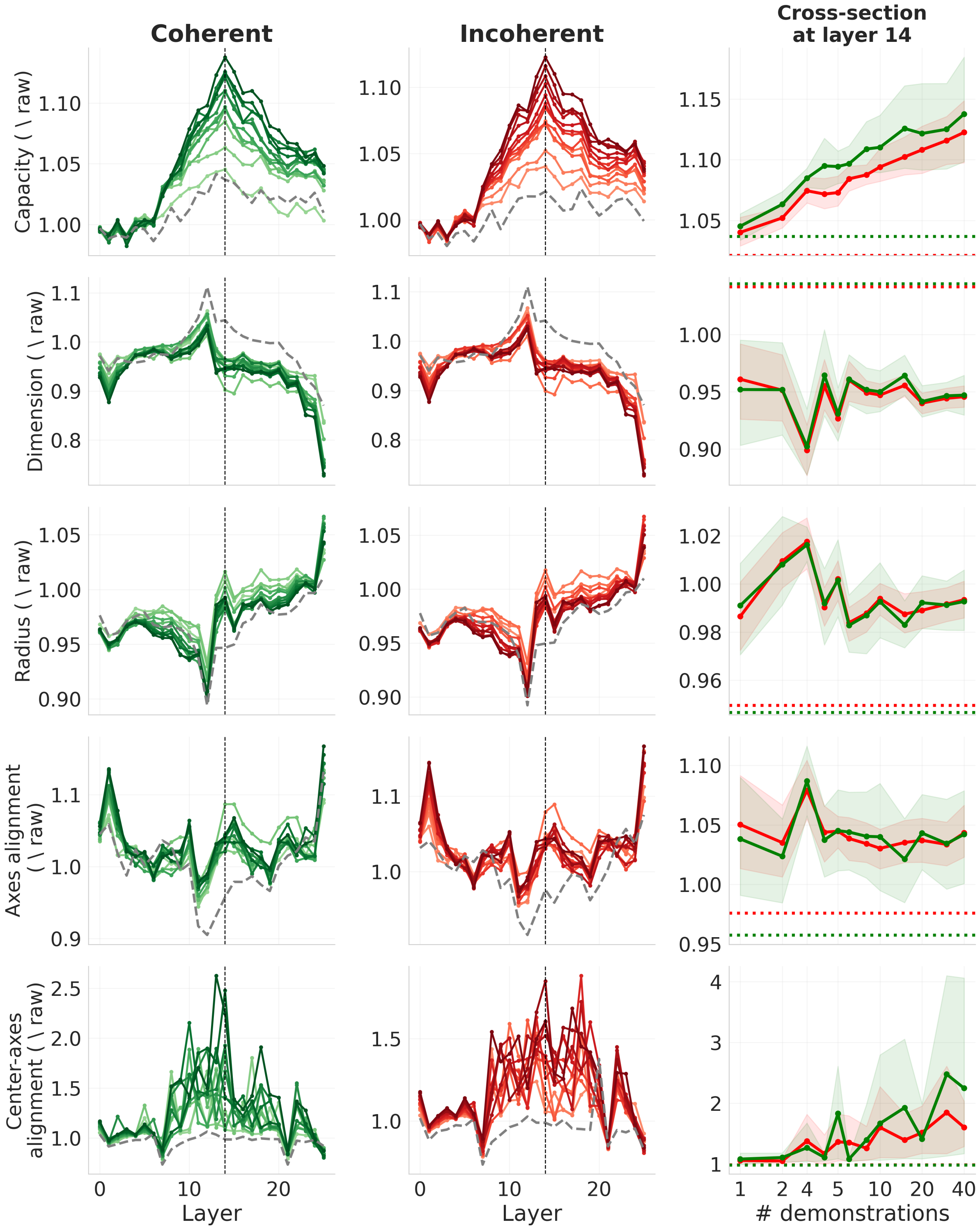

Figure 23: Geometric measures of **sentence-level** representation during coherent and incoherent task-prompting of **Gemma2-2b**. Gradient color shows number of demonstration examples (darker — more examples). Dashed lines — instruction prompt

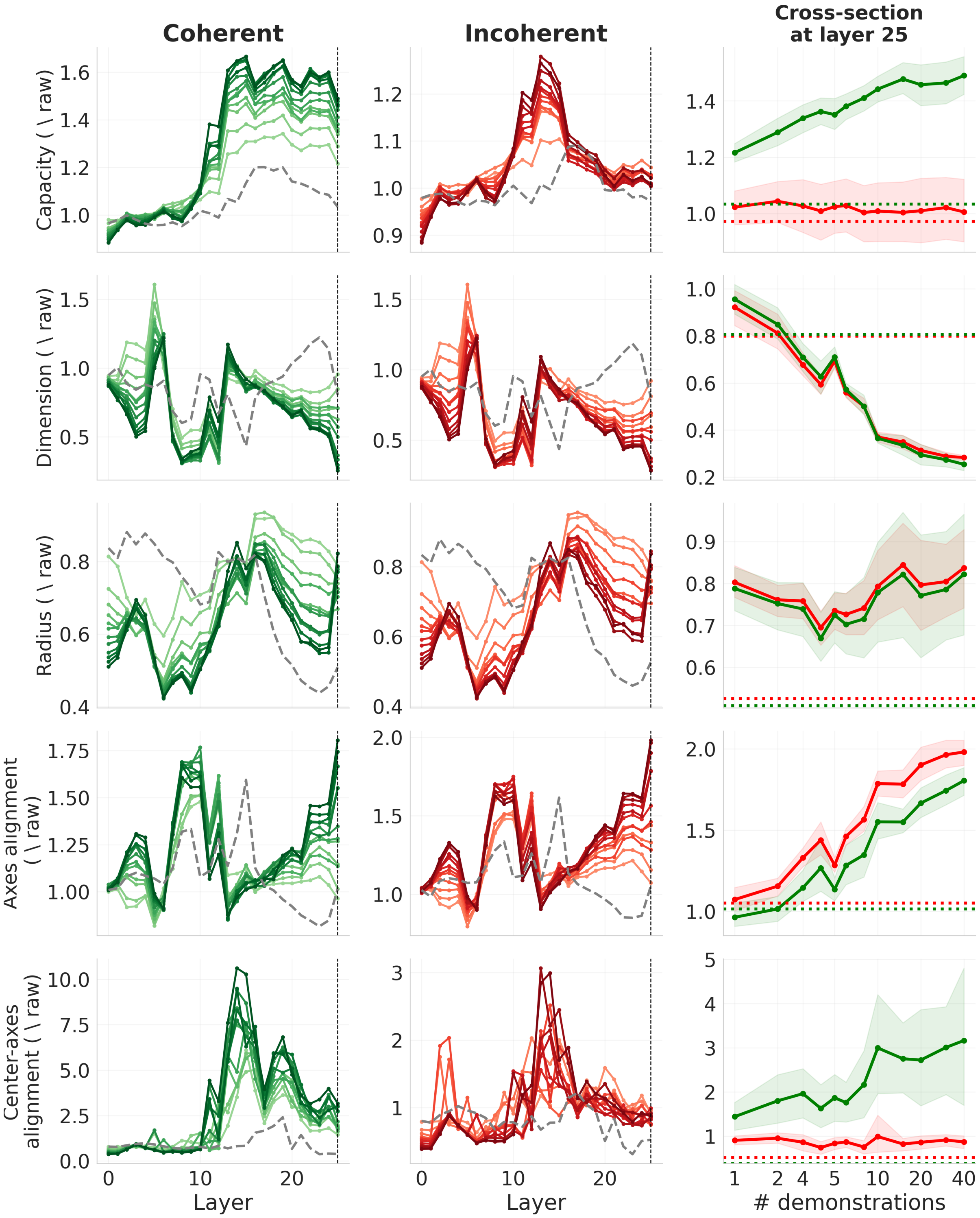

Figure 24: Geometric measures of **last-token** representation during coherent and incoherent task-prompting of **Gemma2-2b**. Gradient color shows number of demonstration examples (darker — more examples). Dashed lines — instruction prompt.

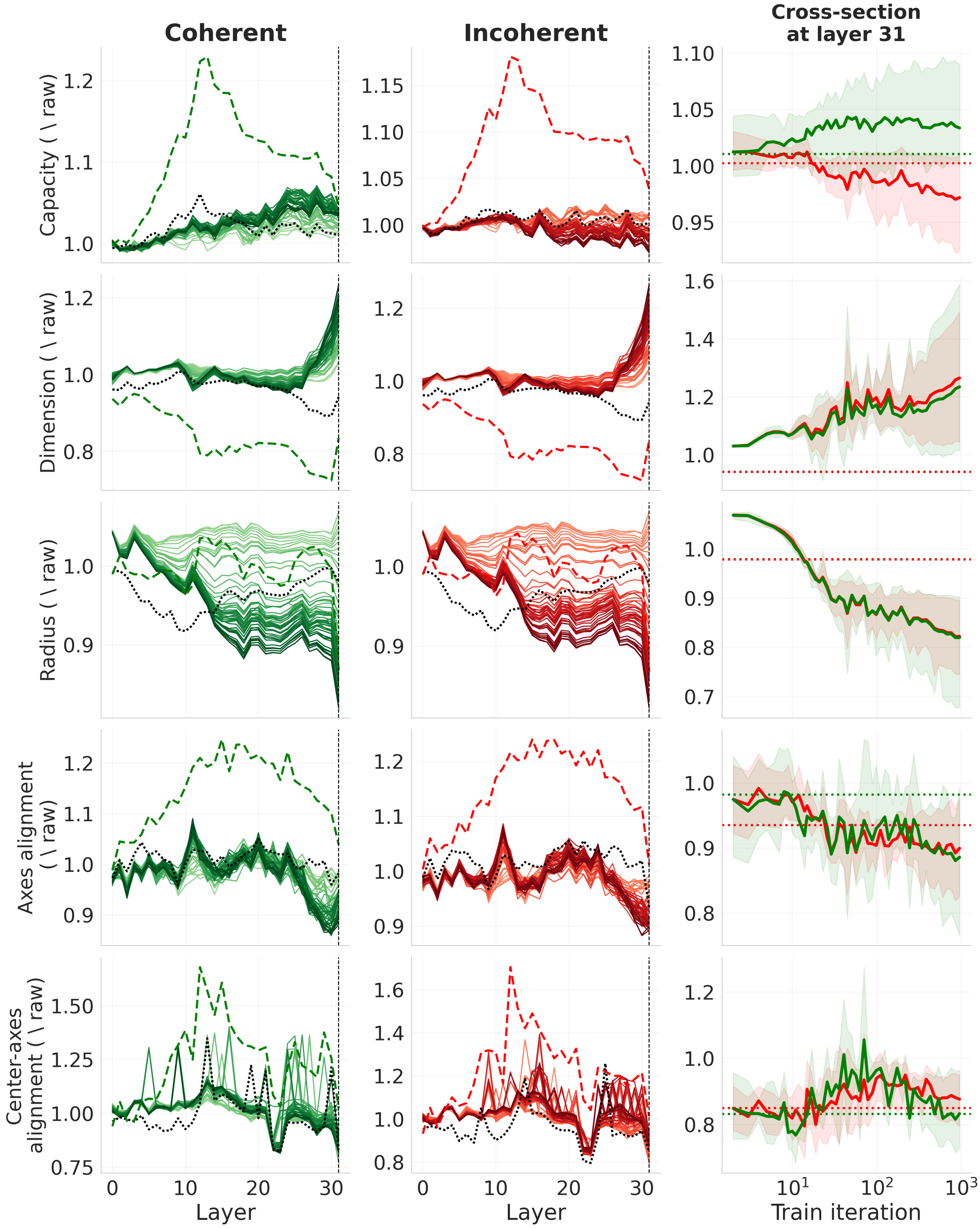

Figure 25: Manifold capacity and geometric measures of **sentence-level** embeddings during training soft-prompt of length 5 (Llama3.1-8b). Gradient color shows training iterations (darker — later epochs). Dashed lines — demonstrations prompt with 40 examples for reference. Dotted — instruction prompt.

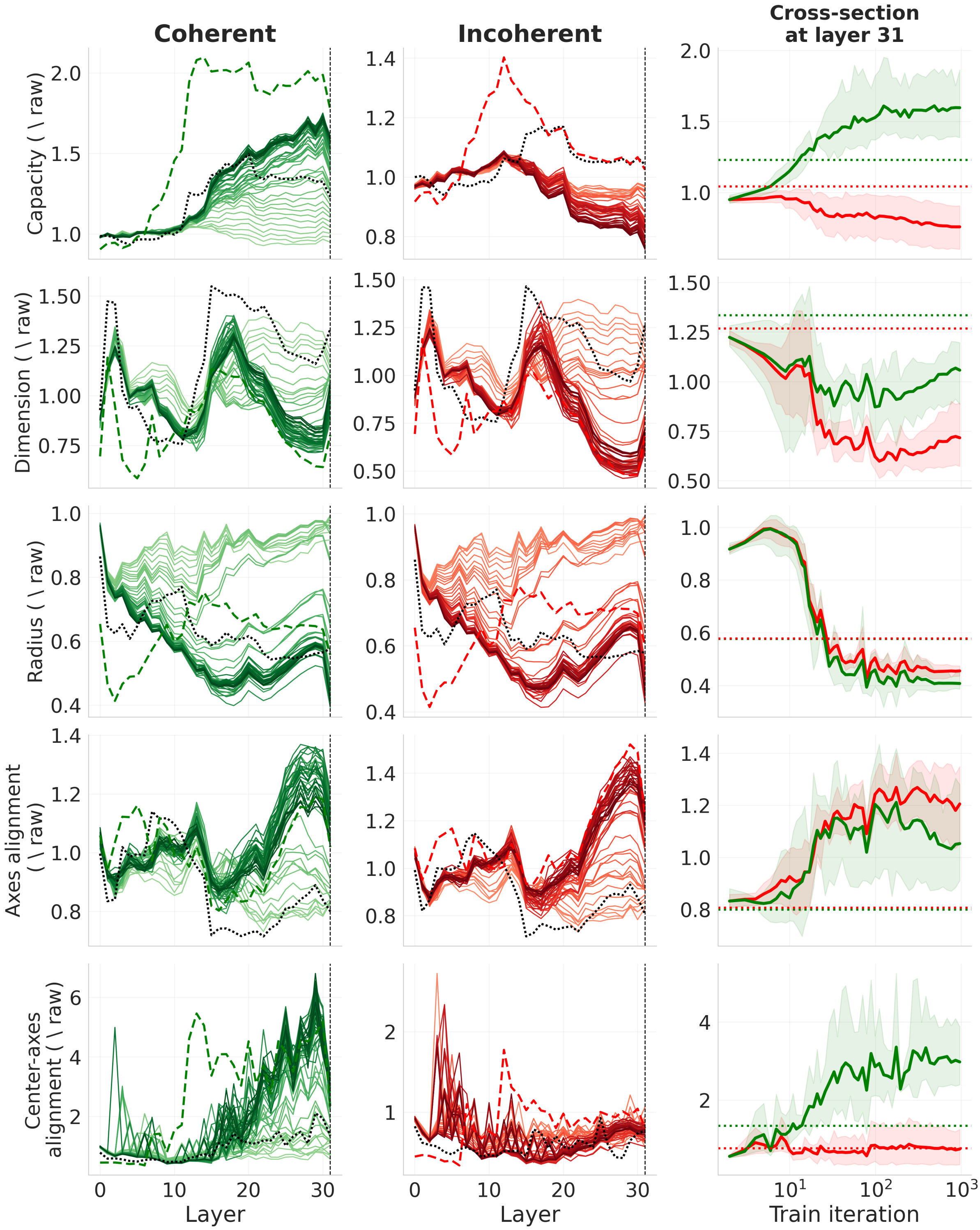

Figure 26: Manifold capacity and geometric measures of **last token** embeddings during training soft-prompt of length 5 (Llama3.1-8b). Gradient color shows training iterations (darker — later epochs). Dashed lines — demonstrations prompt with 40 examples for reference. Dotted — instruction prompt.