Less is More: Local Intrinsic Dimensions of Contextual Language Models

Benjamin Matthias Ruppik[1] **Julius von Rohrscheidt**[2,3] **Carel van Niekerk**[1] **Michael Heck**[1]
Renato Vukovic[1] **Shutong Feng**[1] **Hsien-chin Lin**[1] **Nurul Lubis**[1]
Bastian Rieck[2,4] **Marcus Zibrowius**[1] **Milica Gašić**[1]
[1] Faculty of Mathematics and Natural Sciences, Heinrich Heine University Düsseldorf, Germany
[2] Institute of AI for Health, Helmholtz Munich, Germany
[3] Technical University of Munich, Germany
[4] University of Fribourg, Switzerland
{ruppik,niekerk,heckmi,revuk100,fengs,
linh,lubis,marcus.zibrowius,gasic}@hhu.de
julius.rohrscheidt@helmholtz-munich.de
bastian.grossenbacher@unifr.ch

Abstract

Understanding the internal mechanisms of large language models (LLMs) remains a challenging and complex endeavor. Even fundamental questions, such as how fine-tuning affects model behavior, often require extensive empirical evaluation. In this paper, we introduce a novel perspective based on the geometric properties of contextual latent embeddings to study the effects of training and fine-tuning. To that end, we measure the local dimensions of a contextual language model's latent space and analyze their shifts during training and fine-tuning. We show that the local dimensions provide insights into the model's training dynamics and generalization ability. Specifically, the mean of the local dimensions predicts when the model's training capabilities are exhausted, as exemplified in a dialogue state tracking task, overfitting, as demonstrated in an emotion recognition task, and grokking, as illustrated with an arithmetic task. Furthermore, our experiments suggest a practical heuristic: reductions in the mean local dimension tend to accompany and predict subsequent performance gains. Through this exploration, we aim to provide practitioners with a deeper understanding of the implications of fine-tuning on embedding spaces, facilitating informed decisions when configuring models for specific applications. The results of this work contribute to the ongoing discourse on the interpretability, adaptability, and generalizability of LLMs by bridging the gap between intrinsic model mechanisms and geometric properties in embeddings.

1 Introduction

Large language models (LLMs) have transformed natural language processing in recent years, achieving impressive performance across a variety of tasks (Brown et al., 2020; Devlin et al., 2019; Jiang et al., 2023; Radford et al., 2018; Touvron et al., 2023). These models learn contextual token embeddings in high-dimensional latent spaces, whose structure governs how information is represented and processed. However, most performance diagnostics in LLMs rely on supervised validation or task-specific probes, with few attempting to understand the geometry of the LLM embedding spaces. Our paper is motivated by a central question: *Can structural changes in the embedding space yield unsupervised insights into model behavior across language modeling tasks?* We address this question by applying a localized version of the TwoNN estimator (Facco et al., 2017) to quantify the *local intrinsic dimension* of contextual token embeddings. While the ambient

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

dimension of these embeddings is typically large (ranging from hundreds to thousands of dimensions), our observations show that the local intrinsic dimension reflects a lower-dimensional manifold structure that varies across data space regions. This heterogeneity allows us to derive model- and data-specific geometric signatures without relying on labeled validation data or task-specific supervision.

Contributions We present a unified framework for analyzing LLM training dynamics through dimensionality. Based on this framework, we (i) demonstrate that fine-tuning reshapes the local intrinsic dimension in a *dataset-specific* manner, allowing us to infer overlap between training data used for fine-tuning with validation (or test) data from geometric shifts alone, without requiring labels; (ii) show that dimensionality anticipates the onset of *grokking* in synthetic arithmetic tasks, identifying generalization beyond training data from the dimensionality of the model's embeddings of the training data alone; (iii) demonstrate that local intrinsic dimensions can indicate *training convergence*, as evidenced in a sequence-tagging-based dialogue state tracking task where stabilizing dimension correlates with model stabilization; and (iv) show that local intrinsic dimensions detect *overfitting* in a sequence classification setting, where an initial drop followed by a rise in dimension reflects the model's tradeoff between generalization and memorization. In total, our results suggest that local intrinsic dimensions serve as a valuable unsupervised signal for practitioners seeking to interpret and monitor LLM behavior. Beyond offering empirical insights, our work thus highlights the potential of geometric descriptors to complement traditional evaluation methods and potentially serve to inform future model design.[1]

2 Related Work

Our work lies at the intersection of geometry-aware language model analysis and dataset-level representation diagnostics. While prior efforts focus on the intrinsic dimensions of single sequences or model parameters, our contribution is quantifying local intrinsic dimensions across datasets and training stages, enabling unsupervised inspection of generalization dynamics and fine-tuning effects.

Notions of intrinsic dimensions of language models In a first qualitative analysis of the internal representations of transformer models, Ethayarajh (2019) and Cai et al. (2021) identify clusters and low-dimensional manifold structures in contextual embedding spaces. Recent work has started studying relationships between geometric properties of LLMs and corresponding semantics. In Tulchinskii et al. (2023), the authors discover that the average intrinsic global dimension of artificially generated texts is lower than that of human-written texts. In that work, the points in the latent space are taken from a single text paragraph, so their work applies to single data input sequences, not to entire datasets like our approach. Valeriani et al. (2023) investigate how the global intrinsic dimension estimated is altered as data is passed through an LLM. Aghajanyan et al. (2021) define a notion of intrinsic dimension based on restricting the model's parameter space and its effect on the objective function, and show that larger LLMs tend to have smaller intrinsic dimensionality. That work studies the dimensionality of the model's parameter space, and not the latent space created from specific datasets. In Viswanathan et al. (2025), the authors analyze token-level intrinsic dimensions, linking token geometry to model next-token-prediction loss. The difference between their method and ours is that they always consider the contextual token embeddings of a single, sufficiently long prompt as making up the embedding space. In contrast, we sub-sample from an entire dataset split. Lee et al. (2025a) identify structural similarities between the token embedding spaces of different language models, particularly reporting correlations in local intrinsic dimension measures. However, their analysis is limited to the non-contextual token embedding and unembedding layers, without considering contextual representations.

Topology-based analysis of models The topological descriptors in Kushnareva et al. (2021); Lee et al. (2025b); Perez and Reinauer (2022); Tulchinskii et al. (2023) correspond to entire sequences of input data, while our proposed estimates are defined on the token level. In Durrani et al. (2022), pre-trained and fine-tuned models are compared using hierarchical clustering and alignment functions. The results indicate that the latent space in the higher layers adapts to task-specific concepts, while the lower layers preserve the general concepts learned in the pre-trained model. Ed-dib et al. (2024) introduces a method for fine-tuning large language models by dynamically adjusting LoRA

[1] Our code is available at `https://github.com/aidos-lab/Topo_LLM_public` and `https://github.com/aidos-lab/grokking-via-lid`.

ranks based on the intrinsic dimensionality of hidden state representations. Here, the intrinsic dimension is measured as the rank of an information matrix, and not from the data space as in our work. In Chang et al. (2022), the representational geometry of multilingual language models is explored, focusing on how they balance encoding language-specific and language-agnostic features within a shared multilingual space. Ruppik et al. (2024) utilize topological features derived from neighborhoods in a contextual embedding space to improve performance in a sequence tagging task. The definition of their features requires a fixed ambient data corpus, whereas our methods give an intrinsic measure of the dataset under consideration; thus, our measures are comparable between different model checkpoints of the feature generation model. Both works are situated in the nascent field of *topological deep learning* (Papamarkou et al., 2024), employing concepts from geometry and topology in an *observational manner* to study machine learning models. In contrast to mechanistic interpretability (see, for example, the survey by Ferrando et al. 2024), in which one tries to explain how a language model's internal representations fit together into higher-level abstractions, our approach is more atomic. We study the smallest non-divisible representation in the model (the contextual embedding of single tokens) and the space created by collections of these.

3 Methods

3.1 Large Language Models

Modern contextual language models are typically either masked language models (MLMs) (Devlin et al., 2019; Liu et al., 2019) or autoregressive language models (ALMs) (Radford et al., 2019). MLMs predict masked tokens based on bidirectional context, while ALMs generate tokens sequentially, conditioned on preceding tokens. In both cases, the representation of a token in context is given by a vector at each model layer, enabling geometric analysis of a data corpus on a layer-wise basis. We now give an overview of this construction; the whole procedure is summarized in Algorithm 1.

3.2 Latent Space Modeling

Let $\mathcal{D} = (s_0, s_1, \ldots, s_D)$ be a text corpus and $\mathcal{M}$ a language model (MLM or ALM) of depth l with tokenizer $\mathcal{T}$. Each sequence s_m yields tokenized input:

$$\mathcal{T}(s_m) = \left(t_0^m, \ldots, t_{n_m}^m\right), \tag{1}$$

with corresponding contextualized token embeddings at layer i:

$$\mathcal{M}_i(s_m) = \left(\mathcal{M}_i(t_0^m), \ldots, \mathcal{M}_i(t_{n_m}^m)\right). \tag{2}$$

We distinguish between *regular* embeddings (tokens embedded as-is during a forward-pass) and *masked* embeddings (specific tokens replaced by `[MASK]`, requiring the model to infer their representations from the surrounding context). Despite the simplified notation, embeddings are context-sensitive. The token embedding point cloud at layer i is:

$$\mathbb{T}_i = \{\mathcal{M}_i(t_j^m)\}_{m=0,\ldots,D;\ j\in I_m}, \tag{3}$$

with I_m denoting token indices in the m-th sequence. Distances between points are measured in the ambient Euclidean space. In practice, $\mathbb{T}_i$ can contain millions of vectors, making subsequent neighborhood computation on the full dataset infeasible.

To obtain a representative subset of tokens $\mathbb{T}$, we take a two-step sampling approach: We first sample M sequences from $\mathcal{D}$, and after de-duplication, we sample N vectors from the resulting token embedding space. Finally, we compute neighborhoods $\mathcal{N}_L(t_j; \mathbb{T})$ for each token using a locality parameter L. We assume the dataset is drawn from an underlying text-generating distribution, and our sampling approximates its geometric structure. We confirm this assumption in our sensitivity analysis in Appendix A.

3.3 Comparing Latent Spaces Across Models

We compare a base model $\mathcal{M}$ with its fine-tuned variant $\mathcal{M}^\Delta$, trained on a corpus Δ. Since both models share the same architecture and tokenizer, there exists a canonical bijection between $\mathcal{M}_i(\mathcal{D})$ and $\mathcal{M}_i^\Delta(\mathcal{D})$, mapping each token's representation in the base model to its counterpart in the fine-tuned model. More generally, such a bijection arises whenever the underlying model architectures

Algorithm 1: COMPUTE LOCAL DIMENSION ESTIMATES

Input : Text corpus $\mathcal{D}$;
Embedding model $\mathcal{M}$ (with corresponding tokenizer $\mathcal{T}$);
Dimension estimator $\dim$ (default $\dim = \mathrm{TwoNN}$);
Three parameters:
– Size of text corpus sequence sub-sample $M \in \mathbb{N}$;
– Size of token sub-sample $N \in \mathbb{N}$;
– Local neighborhood size $L \in \mathbb{N}$.

Output : Token-level local estimates given as a vector in $\mathbb{R}^N_{\geq 0}$, i.e., one non-negative number for each of the sub-sampled tokens.

Take a random sub-sample $\mathcal{D}_{\sim M}$ of size M of the dataset sequences ;
Compute embeddings for each token in each sequence of $\mathcal{D}_{\sim M}$;
De-duplicate the embedding vectors ;
Take token vector sub-sample of size N, call the resulting set of token vectors $\mathbb{T}$;
for *all* $t_j \in \mathbb{T}$ **do**
    Compute the L nearest neighbors of t_j in $\mathbb{T}$, resulting in the neighborhood $\mathcal{N}_L(t_j; \mathbb{T})$;
    Compute the local estimate $\dim(\mathcal{N}_L(t_j; \mathbb{T})) \in \mathbb{R}_{\geq 0}$ of this neighborhood ;
return *The vector of token-level dimension estimates.*

and tokenizers are identical. After identical sub-sampling, this allows for point-wise comparison of the resulting latent spaces to evaluate geometric changes. Next, we define the local estimates used for this comparison.

3.4 Local Dimension Estimates

The TwoNN estimator (Facco et al., 2017) approximates the intrinsic dimension of a point cloud as a positive real number using only the distances r_1 and r_2 of each point to its nearest and second-nearest neighbors. It turns out that under weak assumptions, the ratio $\frac{r_2}{r_1}$ of these distances is Pareto-distributed (see Denti et al. (2021) for details). Then, the intrinsic dimension can be estimated from parameters describing this distribution, as long as the density around the points is locally approximately constant on the scale defined by the distance of the second nearest neighbor.

We turn this into a *local measure* as follows: Let X be a point cloud and $v \in X$. For $L \in \mathbb{N}_{>0}$, we define the L-local TwoNN estimate of X at v as $\mathrm{TwoNN}(\mathcal{N}_L(v; X))$, i.e., as the TwoNN estimate of the local neighborhood $\mathcal{N}_L(v; X)$ of v of size L in X. In this way, the parameter L controls the locality scale of the dimension estimation. In the context of our model $\mathcal{M}$ and corpus $\mathcal{D}$, we take $X = \mathbb{T}$ and compute a vector of positive real-valued numbers $\mathrm{TwoNN}(\mathcal{N}_L(v; X))_{v \in \mathbb{T}} \in \mathbb{R}^N_{\geq 0}$, one for each token $v \in \mathbb{T}$. We subsequently aggregate this to a mean estimate per corpus/model setup. While sensitive to hyperparameters such as L and sample sizes, we show in Appendix A that estimates remain stable under reasonable settings, enabling consistent comparisons across datasets and models.

4 Experiments

We now apply our method to analyze how embedding spaces evolve during LLM-related learning tasks. Unless stated otherwise, all results pertain to the model's final hidden layer, i.e., $\mathbb{T}_{-1}$ as defined in Section 3. We refer the reader to Appendix C.2 for results on layers other than the last. Subsequently, we will focus on four central questions:

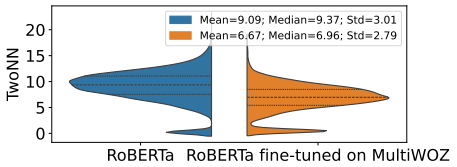
(Q1) How does fine-tuning on different datasets alter latent space geometry?
(Q2) How can local dimension estimates detect *grokking*?
(Q3) How can local dimension estimates detect the limit of *training capabilities*?
(Q4) How can local dimension estimates detect *overfitting*?

4.1 Fine-Tuning Induces Dataset-Specific Shifts in Heterogeneous Local Dimensions

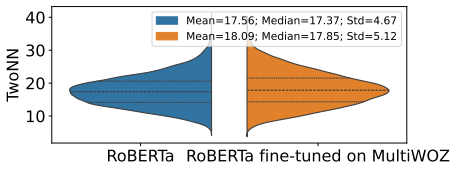
To understand how fine-tuning alters model representations, we investigate the distribution of local intrinsic dimensions across token embeddings. These dimensions, estimated using the method introduced in Section 3, reveal nuanced changes in the geometry of the embedding space.

Setup We evaluate models on datasets with varying domains and familiarity to the base and fine-tuned models: `MultiWOZ2.1` (Eric et al., 2020): Human-human multi-domain dialogues; `Schema-Guided Dialogue SGD` (Rastogi et al., 2020): Human–virtual assistant dialogues; `Reddit`: Reddit comments from the year 2022 mentioning Tesla, Inc.; `ICLR 2024` Submissions: Titles and abstracts of ICLR 2024 papers collected by us; and `Wikipedia`: The Hugging Face `wikitext-103-v1` corpus.

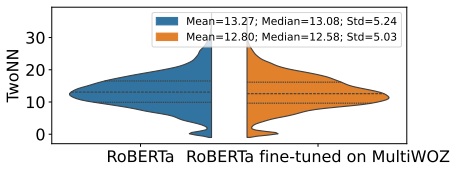
As can be seen from the selection of pre-training datasets of the respective models (Liu et al., 2019; Radford et al., 2019), this choice allows us to compare: (i) distributions not seen during pre-training or fine-tuning (e.g., `Reddit`, `ICLR 2024`, which were released after the pre-training of the model concluded), (ii) distributions seen during pre-training (`Wikipedia`), and (iii) distributions used for fine-tuning (e.g., `MultiWOZ`, `SGD`). This contrast allows us to probe how local intrinsic dimensions behave in both seen and unseen data regimes. We split `Reddit` and `Wikipedia` into training (80%), validation (10%), and test (10%) subsets. Chronologically-ordered datasets (e.g., `Reddit`) are shuffled before splitting to avoid temporal bias. `Wikipedia` was pre-processed by removing headings, empty lines, and stripping leading and trailing whitespace. Further dataset statistics are available in Table 1.



(a) TwoNN estimates on `MultiWOZ` validation



(b) TwoNN estimates on `Wikipedia` validation



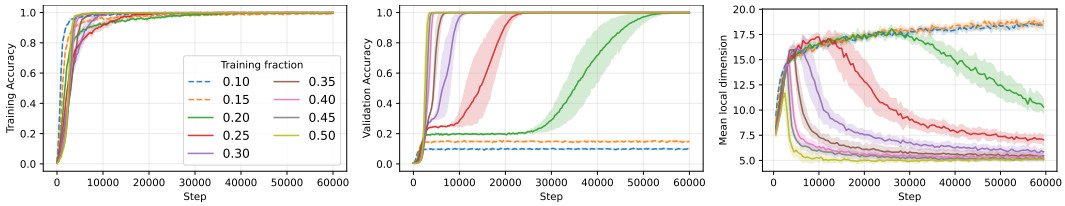
(c) TwoNN estimates on `Reddit` validation

Figure 1: Comparison of local intrinsic dimensions (LIDs) across three data modalities. The distribution of the local estimates over tokens is shown in the violin plot, together with their means and quartiles. The LID of embeddings originating from the fine-tuning distribution (`MultiWOZ`) differs markedly between models, whereas the LIDs for the out-of-distribution corpora (`Wikipedia`, `Reddit`) are almost indistinguishable.

Fine-tuning of the RoBERTa-base models (Liu et al., 2019) is performed using masked language modeling with a masking probability of 0.15. Each model is trained for 5 epochs on $10\,000$ training examples using a batch size of 8, a learning rate peaking at $5 \cdot 10^{-5}$ with 500 warmup steps, and linear decay thereafter. Weight decay of 0.01 is applied throughout. For evaluation, we embed a subset of the *validation* split of each dataset using both the base and fine-tuned model, ensuring the embeddings are out-of-sample relative to the fine-tuning set.

Our discussion in this section focuses on the masked language model RoBERTa, whose architectures remain widely used in practice where encoder-based models are needed (Warner et al., 2025) and autoregressive alternatives are less suitable, such as in information retrieval. Results for additional fine-tuning of the autoregressive models GPT-2-medium, Phi-3.5-mini-instruct, and Llama-3.1-8B can be found in Appendix C.1, where similar observations hold.

Results We find that local intrinsic dimensions vary significantly across tokens, as illustrated by the wide spread of the TwoNN distributions in Figure 1. Here, we choose the parameters $M = 7000$, $N = 60\,000$, $L = 128$. This heterogeneity is consistent across different models and datasets. It aligns with prior observations that numerical data is often not confined to a single manifold of uniform dimensionality, but is instead composed of multiple regions with varying local geometry (Brown et al., 2023). These findings reinforce our decision to favor local over global estimates of embedding dimensionality.



(a) **Training** accuracy (b) Validation accuracy (c) **Training** mean local estimates

Figure 2: Training a model on addition mod $p = 197$ with different training data fraction selected from $\{0.1; 0.15; 0.2; 0.25; 0.3; 0.4; 0.5\}$. The plots show the development for 60 000 batches, with mean and 95% confidence interval over 5 training seeds per configuration (plots per seed are in Appendix C.3). The mean local estimates are computed on the **training split** for the parameters $N = 3000$; $L = 64$. Dashed lines highlight the runs where grokking did *not* occur.

Our central observation is that fine-tuning systematically lowers the local intrinsic dimension *only* on the dataset used during the fine-tuning process. As shown in Figure 1a, the TwoNN estimates for the `MultiWOZ`-validation embeddings are markedly lower for the fine-tuned RoBERTa model than for the base model. By contrast, embeddings of unrelated datasets—such as `Wikipedia` (Figure 1b) or `Reddit` comments (Figure 1c)—show no or only minimal changes in their local dimension distributions. Quantitatively, this observation can be validated by the standardized mean difference between the respective cohorts, with a value of 1.19 for `MultiWOZ`, and 0.08 (resp. 0.1) for `Reddit` (resp. `Wikipedia`).

We find this shift to be stable across training, validation, and test splits of the same dataset, suggesting that it reflects robust structural changes in the latent space rather than artifacts (see Appendix A for a thorough sensitivity analysis showing the method's robustness to the choice of sampling parameters). This behavior highlights strong dataset-specificity in how local geometry adapts under fine-tuning. The effect is absent when evaluating on out-of-distribution data, supporting the view that local dimension reductions correspond to improved fit and specialization on the fine-tuned task, while leaving unrelated regions of the embedding space unchanged.

4.2 Local Dimensions Detect Grokking

Grokking is the phenomenon that a machine learning model acquires specific skills only after extended training far beyond overfitting on the training set, which was first discovered by Power et al. (2022). In such cases, from the model's performance on the training set alone, it is a complex problem to predict under which choices of hyperparameters grokking will occur (Junior et al., 2024).

Setup We here consider the task of learning an arithmetic operation on a small group (addition modulo p). For example, for $p = 197$ such an input sequence would be `[155, 'o', 88, '=']`, where the model should predict the result `46`. The tokenization is constructed so that every operand in the group is encoded as a single token, with the addition of the operation token `'o'` and the equality token `'='`. A certain fraction of all valid expressions in the group, ranging from 10 percent to 50 percent, is taken as the training set, and the remainder will be taken as the validation set.

We use a tiny decoder-only transformer model with two layers, 128 hidden dimensions, and four attention heads trained from scratch. Optimization is performed by AdamW (Loshchilov and Hutter, 2017) with learning rate 0.001 (with linear schedule over 400k steps, warmup of 10 steps), batch size 512, weight decay 0.01. Models are trained for many steps beyond the point where training performance saturates, keeping the same setup as in the original paper (Power et al., 2022).

For the local estimates, we set the number of sampled tokens N as the minimum of 3000 and the maximum number of tokens available, and a neighborhood size L of 64. Since the number of input sequences is comparatively small, we choose M to encompass everything, and only subsample in the token selection step. Still, we observe a similar qualitative behavior for other reasonable choices of these hyperparameters. The hidden states at the last layer are sampled from all available tokens, which could come from the operands and the operation or equality symbol.

Results Performance measures versus local estimates are shown in Figure 2 for the first $60\,000$ training steps averaged over seeds (we present plots for individual seeds in Figure 16 in the appendix). The training accuracy quickly reaches an almost perfect score in all cases, with the validation accuracy lagging behind. In the time frame under consideration, the model can generalize and reach almost perfect validation accuracy only in those runs for which the training data portion exceeds 20%. Based on the loss and accuracy of the training data alone, one would not be able to predict which configurations can break out and generalize to the validation data.

A typical pattern for the mean local dimension computed on the training data in all these runs is that it increases in the first few thousand global steps (see Figure 2c). But subsequently, the training mean local dimension starts dropping significantly for those runs that exhibit grokking. Observe that the timing of this drop coincides with the start of the increasing validation accuracy in Figure 2b. We can conclude that in this setting, a drop in the mean local dimension on the training set strongly indicates a successful generalization to unseen validation data.

For those runs in which the validation accuracy does not increase beyond the fraction of training data during the selected training time (with 10% and 15% of training data, highlighted via dashed lines), the local dimension increases and then stays mostly flat. Quantitatively, the Spearman rank correlation between the training/validation accuracy and mean local dimension measured over the training split is positive for those training fractions where the model does not grok ($0.880/0.076$ and $0.922/0.257$ for training fractions 10% and 15%). This behavior suggests that the model only tries to learn the training examples by heart and fails to generalize to unseen data. For all runs grouped by training fraction $\geq 20\%$, which all exhibit grokking, the Spearman rank correlation between both training/validation accuracy and training mean local dimension is negative.

4.3 Local Dimensions Detect Exhaustion of Training Capabilities

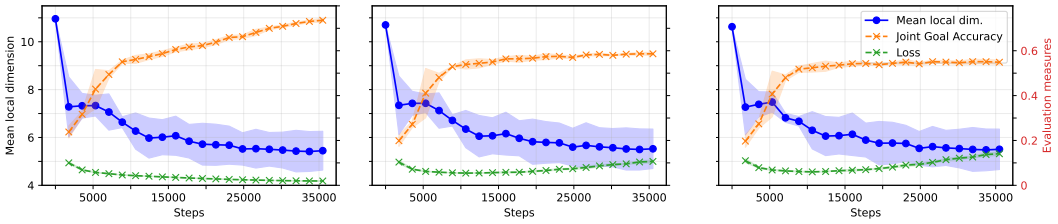
Finding relevant text segments in response to a query, also called *span prediction* or *sequence tagging*, is a highly relevant problem in various natural language processing settings (Jurafsky and Martin, 2025). One such application is retrieval augmented generation (RAG), where, during the information retrieval step, relevant passages from a large corpus need to be selected (Fan et al., 2024) before generating an answer. Here we study it in the context of dialogue state tracking—a critical task of dialogue modeling.

Setup To demonstrate the application of our local dimension estimates in this setting, we compute them for the sequence-tagging-based dialogue state tracking model *TripPy-R* (Heck et al., 2022) trained on the `MultiWOZ2.1` dataset (Eric et al., 2020). In dialogue state tracking, the task is to predict the user's intent from the natural language input utterances, and keep track of the user's goal described throughout the conversation by updating the dialogue state (Young et al., 2010). The contextualized hidden states for computing our measure are taken from the last layer of the unified encoder component, which has the RoBERTa architecture. The actual state tracking is performed based on classifier outputs derived from this encoder's output. The encoder is fine-tuned during the TripPy-R training via the loss signal derived from predicting the current dialogue turn's state update. We train the models for 20 epochs with Adam, with a linear learning rate schedule up to $5 \cdot 10^{-5}$ that starts with one warm-up epoch.

The input data for the local dimension estimates is formatted in the same way as it is used during training and inference. A single input sequence consists of a dialogue turn, followed by the dialogue history, where the different sequence components are separated via special tokens. For the local dimension estimates, we sample $M = 7000$ sequences from the training, validation, and test split, with a token subsample size $N = 60\,000$, and neighborhood size $L = 128$.

Results Figure 3 shows the development of two model performance measures: One is the differentiable model loss, which is used for backpropagation. The other is the non-differentiable *joint-goal accuracy* (JGA), which is used to judge the downstream performance of the state tracking model.

Note that the mean local dimension estimates behave similarly when comparing the training, validation, and test splits. This is in contrast to the loss and JGA: On the training set, the loss (green) is monotonically decreasing, while the JGA (orange) is increasing. On the other hand, the loss reaches a minimum on the validation set after 7500 batches, long before the JGA converges. In this case, the differentiable model validation loss would give the wrong impression that the model has finished



(a) MultiWOZ Training split (b) MultiWOZ Validation split (c) MultiWOZ Test split

Figure 3: Development of TripPy-R performance measures (model loss in green; joint goal accuracy in orange) compared with mean local dimension estimates (blue) evaluated on the training, validation, and test split of the MultiWOZ dataset. We show the mean and standard deviation of the measures evaluated at the end of each epoch over six different model training seeds.

learning generalizable representations. However, from the dimension estimate, we conclude that the model has not converged at that point. The mean local dimension is still decreasing, and one should continue training. Notably, we can make this observation on the training data alone and would not need a separate labeled validation set when relying upon our estimates.

Quantitatively, taken over the whole time series progression and step-wise averaged over six seeds, the mean local dimension exhibits a strong negative Spearman rank correlation with the JGA of -0.982/-0.958/-0.905 for the training/validation/test split, respectively. This confirms that progressive compression of the embedding space directly reflects improvements in generalizable task performance. Correlation between the mean local dimension and the loss is positive (0.982) on the training set, and negative $(-0.612/-0.734)$ on the validation and test set, mirroring the fact that the loss does not align consistently with JGA performance over the different splits.

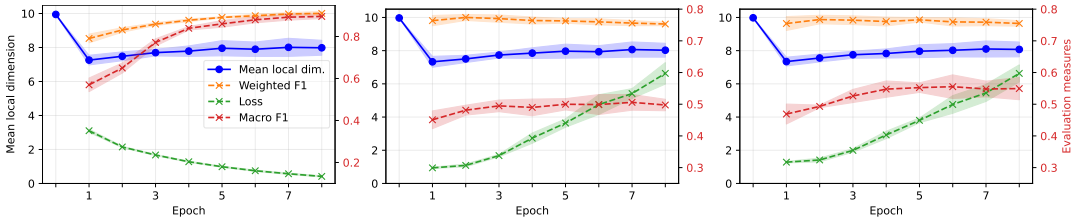
In another direction, we can see that later in the training, after roughly $25\,000$ batches, the stabilization of the local dimension estimates coincides with the convergence of the model performance regarding JGA on the validation and test sets. This effect of stabilization of the local estimates, indicating that the training capabilities have been exhausted, can be seen even more clearly for longer training runs (50 epochs) in Figure 17 in Appendix C.4.

4.4 Local Dimensions Detect Overfitting

We now examine the mean local dimension as an indicator of overfitting in a challenging sequence classification setting.

Setup We use the EmoWOZ dataset (Feng et al., 2022), with the task of classifying dialogue utterances into one of seven emotion classes. Our experiments employ the ERToD model (Feng et al., 2023) in its baseline configuration, a linear classifier built upon a `bert-base-uncased` (Devlin et al., 2019) model feature extractor. We intentionally decide on the setup *without* dialogue history, data augmentation, or auxiliary predictions, which are modifications proposed by the authors to solve the problem of classifying rare emotions. The model is trained for 8 epochs using a linear learning rate schedule with warmup for AdamW and peak learning rate of $2 \cdot 10^{-5}$. At the end of each training epoch, local dimension estimates are computed from hidden states extracted at the last BERT-encoder layer. Thus, this is the latent space that the model has tuned to aid its subsequent classification. We select representative subsets of $M = 7000$ sequences from the training, validation, and test split, with a token subsample size $N = 60\,000$, and neighborhood size $L = 128$. Model performance is evaluated using cross-entropy loss and weighted F1 and macro F1 classification scores. To better capture effects on minority classes, the majority "neutral" class is excluded from the F1-scores.

Results Figure 4 depicts the model's performance measures plotted against the mean local estimates. After the first training epoch, the model's mean local dimension drops notably to ≈ 7.25 from the base model's initial dimension of ≈ 9.94. Over the subsequent eight training epochs, this dimension increases to a value of about 8. Note that this behavior of the mean local estimates can be observed on all three data splits. As expected, the training loss monotonically decreases, while both weighted and macro F1-scores increase on the train set.



(a) EmoWOZ Training split (b) EmoWOZ Validation split (c) EmoWOZ Test split

Figure 4: Development of emotion recognition model performance measures (loss in green; weighted F1 in orange; macro F1 in red) compared with mean local dimension estimates (blue) evaluated on the training, validation, and test split of the EmoWOZ dataset. We show the mean and standard deviation of the measures evaluated at the end of each epoch over four different training seeds.

However, the minimum of the local dimension, followed by an increase, suggests that the model has already found an efficient representation after a single training epoch. We previously observed a connection between rising dimension and memorization in the grokking experiments in Section 4.2. This suspicion is confirmed when considering the performance measures on the validation set in Figure 4b. After the first epoch, the validation loss increases strongly over the following epochs; thus, the model is clearly *overfitting* on the training examples.

In terms of how this influences the classification accuracy, the effect is more nuanced. While the macro F1-score increases and plateaus after about three epochs, the weighted F1-score declines from the end of epoch two onward. This combined behavior implies that the model improves on minority classes at the expense of performance on the majority classes, potentially memorizing rare instances and diminishing its generalization capability. This interplay between macro and weighted F1 suggests that the model is learning a decision boundary that trades off the performance of majority and minority classes, a phenomenon that may be overlooked when looking at the loss curve alone. Practitioners monitoring the shifts of local dimensionality on the training set could, therefore, identify subtler model behavior *without* requiring labeled validation or test data.

Note that in general, the evaluation criteria are task-specific: In contrast to the observations in Section 4.3, where the validation loss is not a reliable stopping criterion when aiming for the target metric JGA, here, the validation loss and weighted F1 are tightly aligned. Hence, the loss serves as a valid stopping signal here. The local dimension measures support this point, since they are negatively correlated with the training loss (Spearman rank -0.952), but positively correlated with the validation (0.952) and test loss (0.976).

4.5 Computational Resources

Fine-tuning of the RoBERTa-base and GPT-2-medium models (in Appendix C.1) on one of our selected datasets can be performed efficiently on a single NVIDIA V100 or GTX1080TI-12GB GPU within a few hours. Additionally, computing the embeddings for a single layer of these models requires only forward passes, which takes approximately 10 minutes on the same hardware.

For the TwoNN computations, which are CPU-intensive, the computational requirements depend on the size of the dataset and the dimensionality of the embeddings. The computation is feasible in 20 minutes on an E5-2640v4 (Broadwell) 2.40GHz dual-core machine with 32GB of RAM using the `scikit-dimension` package (Bac et al., 2021) for a typical dataset with tens of thousands of points in high-dimensional space (ambient dimension in the hundreds). For the grokking experiments, where we evaluate the local dimension frequently, this amounts to a total of 48 CPU core hours per run in the depicted range. However, precise timing may vary depending on implementation optimizations and the chosen neighborhood sizes. For a constant and small neighborhood size L and ambient embedding dimension d, the computation of the neighborhoods for a subsample of N using exact search (Johnson et al., 2019) takes $\mathcal{O}(dN^2)$. Then, since we already have the ratios of closest distances available for each neighborhood, the TwoNN local dimension estimation is a constant overhead of performing a linear fit on values of a transformed cumulative distribution.

4.6 Limitations

While versatile, our framework has several limitations. One drawback is its comparatively high computational complexity; to get a reliable dimension estimate, we need to build a sufficiently large sub-sample of the latent space by performing forward passes on the input data and subsequently computing the neighborhoods and estimates. Being *independent* of the model training procedure, these computations can be performed in parallel to any potential fine-tuning, thus not slowing down the training process. Further work could focus on making the local estimate computation more efficient and developing local, stable, and computationally efficient descriptors.

An intrinsic limitation arises from the assumptions of our dimension estimates. The TwoNN estimate, which we apply locally, is known to accurately reflect the true local dimension only under strong assumptions on the local point sampling distribution: The point sample should come from a Poisson point process on a subspace of uniform dimension. Note that these assumptions and locally constant density for the hidden states of transformer models have been empirically demonstrated (Valeriani et al., 2023; Viswanathan et al., 2025). While the TwoNN method is non-parametric, the precise absolute value of the mean TwoNN estimate depends on the hyperparameter choices in selecting the latent space sub-sample and neighborhood size. This implies that our mean of local dimensions is of a "relative" nature, because we cannot directly compare the dimensions' values between different model architectures. However, this is only of secondary importance for our method, as we are more interested in *changes* of dimension than in absolute values. Moreover, our framework is general, permitting the use of other estimation methods. Our working hypothesis is that high-dimensional neighborhoods reflect over-parameterized and poorly localized behavior, whereas very low LID indicates over-compression or memorization. Between these extremes lies a "sweet spot" in LID, where representations retain the essential degrees of freedom required for effective generalization. Empirically, this intermediate range correlates with peak generalization performance. While this correlation supports the interpretability of our analysis, establishing a causal link between LID reduction and improved generalization remains an open question for future work.

5 Conclusion

We introduce a novel geometric perspective on LLM training dynamics by measuring the local intrinsic dimension of contextual token embeddings. While the latent spaces of contextual language models exhibit regional variation in dimensionality, the mean local dimension provides a stable, interpretable summary across dataset splits. Across diverse tasks, a sustained drop in mean local dimension reliably suggests improved generalization, offering a practical, unsupervised diagnostic signal that, among other things, enables the detection of grokking. Such a marked reduction in intrinsic dimension in supervised downstream tasks highlights their strong compressive effect on the latent space. On the other hand, rising dimensions can point to a tradeoff between generalization and specialization, typically observed during overfitting. Our approach enables monitoring without supervision by labels, which is particularly valuable in low-resource settings where validation labels are limited or unavailable. Thus, our findings highlight the utility of geometric descriptors for monitoring and interpreting LLM behavior beyond traditional label-based metrics.

Impact and Future Work To our knowledge, we are the first to investigate local dimensional shifts and their implications in training and fine-tuning contextual language models for different downstream tasks. The findings of this work enhance the ongoing discussion regarding the interpretability, adaptability, and generalizability of LLMs. Our work provides a foundation for designing better model architectures and developing interventions that utilize the insight that lower intrinsic dimensions benefit machine learning problems. Our measures could inform parameter choices such as LoRA-ranks in the spirit of (Ed-dib et al., 2024), and are broadly applicable for other (post)-training phases, for example, those which involve RL-tuning. While our current dimension estimation method is not differentiable (due to nearest-neighbor graph construction and the subsequent TwoNN estimator), we see designing a differentiable surrogate or proxy loss to encourage local compression during training as an exciting opportunity for future work. The locality of the dimension estimates promises further applications beyond the scope of the current work. One immediate avenue for investigation is to see to what extent the dimension can be used to detect which data the model has been trained on. Another important direction would be to find connections between the underlying meaning that the tokens carry and their corresponding dimension estimates.

Acknowledgments and Disclosure of Funding

This work was made possible through the support of the European Research Council (ERC) under the Horizon 2020 research and innovation program (Grant No. STG2018 804636), the Ministry of Culture and Science of North Rhine-Westphalia within the Lamarr Fellow Network, and the Swiss State Secretariat for Education, Research, and Innovation (SERI). Computational resources were provided by the Centre for Information and Media Technology at Heinrich Heine University Düsseldorf, and Google Cloud.

References

Abdin, M., Aneja, J., Awadalla, H., Awadallah, A., Awan, A. A., Bach, N., Bahree, A., Bakhtiari, A., Bao, J., Behl, H., Benhaim, A., Bilenko, M., Bjorck, J., Bubeck, S., Cai, M., Cai, Q., Chaudhary, V., Chen, D., Chen, D., Chen, W., Chen, Y.-C., Chen, Y.-L., Cheng, H., Chopra, P., Dai, X., Dixon, M., Eldan, R., Fragoso, V., Gao, J., Gao, M., Gao, M., Garg, A., Giorno, A. D., Goswami, A., Gunasekar, S., Haider, E., Hao, J., Hewett, R. J., Hu, W., Huynh, J., Iter, D., Jacobs, S. A., Javaheripi, M., Jin, X., Karampatziakis, N., Kauffmann, P., Khademi, M., Kim, D., Kim, Y. J., Kurilenko, L., Lee, J. R., Lee, Y. T., Li, Y., Li, Y., Liang, C., Liden, L., Lin, X., Lin, Z., Liu, C., Liu, L., Liu, M., Liu, W., Liu, X., Luo, C., Madan, P., Mahmoudzadeh, A., Majercak, D., Mazzola, M., Mendes, C. C. T., Mitra, A., Modi, H., Nguyen, A., Norick, B., Patra, B., Perez-Becker, D., Portet, T., Pryzant, R., Qin, H., Radmilac, M., Ren, L., de Rosa, G., Rosset, C., Roy, S., Ruwase, O., Saarikivi, O., Saied, A., Salim, A., Santacroce, M., Shah, S., Shang, N., Sharma, H., Shen, Y., Shukla, S., Song, X., Tanaka, M., Tupini, A., Vaddamanu, P., Wang, C., Wang, G., Wang, L., Wang, S., Wang, X., Wang, Y., Ward, R., Wen, W., Witte, P., Wu, H., Wu, X., Wyatt, M., Xiao, B., Xu, C., Xu, J., Xu, W., Xue, J., Yadav, S., Yang, F., Yang, J., Yang, Y., Yang, Z., Yu, D., Yuan, L., Zhang, C., Zhang, C., Zhang, J., Zhang, L. L., Zhang, Y., Zhang, Y., Zhang, Y., and Zhou, X. (2024). Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. *arXiv preprint*. arXiv:2404.14219.

Aghajanyan, A., Gupta, S., and Zettlemoyer, L. (2021). Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *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 7319–7328, Online. Association for Computational Linguistics. doi:10.18653/v1/2021.acl-long.568;arXiv:2012.13255.

AI@Meta (2024). Llama 3 Model Card. `https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md`.

Bac, J., Mirkes, E. M., Gorban, A. N., Tyukin, I., and Zinovyev, A. (2021). Scikit-Dimension: A Python Package for Intrinsic Dimension Estimation. *Entropy*, 23(10). doi:10.3390/e23101368;arXiv:2109.02596.

Brown, B. C., Caterini, A. L., Ross, B. L., Cresswell, J. C., and Loaiza-Ganem, G. (2023). Verifying the Union of Manifolds Hypothesis for Image Data. In *The Eleventh International Conference on Learning Representations (ICLR)*. OpenReview:Rvee9CAX4fi;arXiv:2207.02862.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc. arXiv:2005.14165.

Cai, X., Huang, J., Bian, Y., and Church, K. (2021). Isotropy in the Contextual Embedding Space: Clusters and Manifolds. In *International Conference on Learning Representations (ICLR)*. OpenReview:xYGNO86OWDH.

Chang, T., Tu, Z., and Bergen, B. (2022). The Geometry of Multilingual Language Model Representations. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022*

Conference on Empirical Methods in Natural Language Processing, pages 119–136, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-main.9;arXiv:2205.10964.

Denti, F., Doimo, D., Laio, A., and Mira, A. (2021). Distributional Results for Model-Based Intrinsic Dimension Estimators. *arXiv preprint*. arXiv:2104.13832.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *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. doi:10.18653/v1/N19-1423;arXiv:1810.04805.

Durrani, N., Sajjad, H., Dalvi, F., and Alam, F. (2022). On the Transformation of Latent Space in Fine-Tuned NLP Models. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1495–1516, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-main.97;arXiv:2210.12696.

Ed-dib, A., Datbayev, Z., and Aboussalah, A. M. (2024). GeLoRA: Geometric Adaptive Ranks For Efficient LoRA Fine-tuning. *arXiv preprint*. arXiv:2412.09250.

Eric, M., Goel, R., Paul, S., Sethi, A., Agarwal, S., Gao, S., Kumar, A., Goyal, A., Ku, P., and Hakkani-Tur, D. (2020). MultiWOZ 2.1: A Consolidated Multi-Domain Dialogue Dataset with State Corrections and State Tracking Baselines. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 422–428, Marseille, France. European Language Resources Association. ACL Anthology:2020.lrec-1.53;arXiv:1907.01669.

Ethayarajh, K. (2019). How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics. doi:10.18653/v1/D19-1006,arXiv:1909.00512.

Facco, E., d'Errico, M., Rodriguez, A., and Laio, A. (2017). Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific reports*, 7(1):12140. doi:10.1038/s41598-017-11873-y.

Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., and Li, Q. (2024). A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 6491–6501, New York, NY, USA. Association for Computing Machinery. doi:10.1145/3637528.3671470;arXiv:2405.06211.

Feng, S., Lubis, N., Geishauser, C., Lin, H.-c., Heck, M., van Niekerk, C., and Gasic, M. (2022). EmoWOZ: A Large-Scale Corpus and Labelling Scheme for Emotion Recognition in Task-Oriented Dialogue Systems. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4096–4113, Marseille, France. European Language Resources Association. ACL Anthology:2022.lrec-1.436;doi:10.5281/zenodo.6506504;arXiv:2109.04919.

Feng, S., Lubis, N., Ruppik, B., Geishauser, C., Heck, M., Lin, H.-c., van Niekerk, C., Vukovic, R., and Gasic, M. (2023). From Chatter to Matter: Addressing Critical Steps of Emotion Recognition Learning in Task-oriented Dialogue. In Stoyanchev, S., Joty, S., Schlangen, D., Dusek, O., Kennington, C., and Alikhani, M., editors, *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 85–103, Prague, Czechia. Association for Computational Linguistics. doi:10.18653/v1/2023.sigdial-1.8;arXiv:2308.12648.

Ferrando, J., Sarti, G., Bisazza, A., and Costa-jussà, M. R. (2024). A Primer on the Inner Workings of Transformer-based Language Models. *arXiv preprint*. arXiv:2405.00208.

Heck, M., Lubis, N., van Niekerk, C., Feng, S., Geishauser, C., Lin, H.-C., and Gašić, M. (2022). Robust Dialogue State Tracking with Weak Supervision and Sparse Data. *Transactions of the Association for Computational Linguistics*, 10:1175–1192. doi:10.1162/tacl_a_00513;arXiv:2202.03354.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. (2023). Mistral 7B. *arXiv preprint*. arXiv:2310.06825.

Johnson, J., Douze, M., and Jégou, H. (2019). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547. doi:10.1109/TBDATA.2019.2921572;arXiv:1702.08734.

Junior, T. N. P., Zhou, H., Pezeshki, M., Rish, I., and Dumas, G. (2024). Predicting Grokking Long Before it Happens: A look into the loss landscape of models which grok. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*. OpenReview:AUjCSjxhGj;arXiv:2306.13253.

Jurafsky, D. and Martin, J. H. (2025). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. Online manuscript, 3rd edition. Online manuscript released August 24, 2025.

Kushnareva, L., Cherniavskii, D., Mikhailov, V., Artemova, E., Barannikov, S., Bernstein, A., Piontkovskaya, I., Piontkovski, D., and Burnaev, E. (2021). Artificial Text Detection via Examining the Topology of Attention Maps. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 635–649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. doi:10.18653/v1/2021.emnlp-main.50;arXiv:2109.04825.

Lee, A., Weber, M., Viégas, F., and Wattenberg, M. (2025a). Shared Global and Local Geometry of Language Model Embeddings. In *Second Conference on Language Modeling*. OpenReview:aJDykpJAYF;arXiv:2503.21073.

Lee, J. H., Jiralerspong, T., Yu, L., Bengio, Y., and Cheng, E. (2025b). Geometric Signatures of Compositionality Across a Language Model's Lifetime. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T., editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5292–5320, Vienna, Austria. Association for Computational Linguistics. doi:10.18653/v1/2025.acl-long.265;arXiv:2410.01444.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint*. arXiv:1907.11692.

Loshchilov, I. and Hutter, F. (2017). Decoupled Weight Decay Regularization. *arXiv preprint*. arXiv:1711.05101.

Papamarkou, T., Birdal, T., Bronstein, M., Carlsson, G., Curry, J., Gao, Y., Hajij, M., Kwitt, R., Liò, P., Lorenzo, P. D., Maroulas, V., Miolane, N., Nasrin, F., Ramamurthy, K. N., Rieck, B., Scardapane, S., Schaub, M. T., Veličković, P., Wang, B., Wang, Y., Wei, G.-W., and Zamzmi, G. (2024). Position: Topological deep learning is the new frontier for relational learning. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F., editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 39529–39555. PMLR. OpenReview:Nl3RG5XWAt;arXiv:2402.08871.

Perez, I. and Reinauer, R. (2022). The Topological BERT: Transforming Attention into Topology for Natural Language Processing. *arXiv preprint*. arXiv:2206.15195.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. (2022). Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets. *arXiv preprint*. arXiv:2201.02177.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training. *OpenAI blog*. Link.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*. Link.

Rastogi, A., Zang, X., Sunkara, S., Gupta, R., and Khaitan, P. (2020). Towards Scalable Multi-Domain Conversational Agents: The Schema-Guided Dialogue Dataset. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8689–8696. doi:10.1609/aaai.v34i05.6394;arXiv:1909.05855.

Ruppik, B. M., Heck, M., van Niekerk, C., Vukovic, R., Lin, H.-c., Feng, S., Zibrowius, M., and Gasic, M. (2024). Local Topology Measures of Contextual Language Model Latent Spaces with Applications to Dialogue Term Extraction. In Kawahara, T., Demberg, V., Ultes, S., Inoue, K., Mehri, S., Howcroft, D., and Komatani, K., editors, *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 344–356, Kyoto, Japan. Association for Computational Linguistics. doi:10.18653/v1/2024.sigdial-1.31;arXiv:2408.03706.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint*. arXiv:2302.13971.

Tulchinskii, E., Kuznetsov, K., Kushnareva, L., Cherniavskii, D., Nikolenko, S., Burnaev, E., Barannikov, S., and Piontkovskaya, I. (2023). Intrinsic Dimension Estimation for Robust Detection of AI-Generated Texts. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 39257–39276. Curran Associates, Inc. arXiv:2306.04723.

Valeriani, L., Doimo, D., Cuturello, F., Laio, A., Ansuini, A., and Cazzaniga, A. (2023). The geometry of hidden representations of large transformer models. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 51234–51252. Curran Associates, Inc. arXiv:2302.00294.

Viswanathan, K., Gardinazzi, Y., Panerai, G., Cazzaniga, A., and Biagetti, M. (2025). The Geometry of Tokens in Internal Representations of Large Language Models. *arXiv preprint*. arXiv:2501.105739.

Warner, B., Chaffin, A., Clavié, B., Weller, O., Hallström, O., Taghadouini, S., Gallagher, A., Biswas, R., Ladhak, F., Aarsen, T., Adams, G. T., Howard, J., and Poli, I. (2025). Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T., editors, *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. doi:10.18653/v1/2025.acl-long.127;arXiv:2412.13663.

Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., and Yu, K. (2010). The hidden information state model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174. doi:10.1016/j.csl.2009.04.001.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper has four main claims, which are listed in the last paragraph of the Introduction Section 1. Each claim corresponds to a subsection of the Experiments Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In the methods section, we mention limitations whenever they lead to restrictions in our setup, for instance, in Section 3.4. We additionally dedicate Section 4.6 to a separate discussion of limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We do not provide new theoretical results in this work. For those local dimension estimation methods that we apply, in Section 3.4, we clearly state references to the original publications that contain a detailed discussion of the assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Our method is described in full detail in Section 3 and in Algorithm 1. Sections 4.1 to 4.4 contain all the details necessary to reproduce the model training runs, which form the basis of our main findings. Moreover, our code is released as part of the supplemental material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our code is submitted as part of the supplemental material and will be made public upon publication. All datasets are freely available under permissive licenses; more details can be found in the experiments in Section 4 and Appendix B.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and test details, including hyperparameters and details about splits are described in the experiment setup in Sections 4.1 to 4.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: To confirm the findings of our main results, we run every experimental setup for several seeds, and report confidence intervals or standard deviation in Figures 2 to 4. All standard deviations in this paper are sample standard deviations, computed with the numpy and pandas library functions `np.std(ddof=1)` and `pd.std()`. Our sensitivity analysis in Appendix A studies the effect of sampling size and locality parameters by reporting the results for multiple sub-sampling seeds per hyperparameter choice. The violin plots of the local estimates in Figure 1 and Appendix C.1 show the entire distribution and are reported together with mean and standard deviations.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We discuss the runtime and computational resources in Section 4.5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We reviewed the NeurIPS Code of Ethics and confirm that we have followed every aspect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work is foundational and provides general insights into the training dynamics of language models. While we acknowledge the general risks involved in developing and deploying machine learning models, our insights do not lead to new negative societal implications. Since we aim to geometrically understand the internal mechanism in a language model under different fine-tuning and training regimes, we hope that our work can help in safety and interpretability research in the future.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We believe our foundational work does not pose such safety risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original research papers introducing the models used in Sections 3.1 and 4.2 to 4.4 and acknowledge code that we used as the basis for our work in the submitted supplemental material. All datasets discussed are freely available under permissive licenses, and we provide links to the dataset cards in Appendix B.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our code, which is submitted as part of the supplemental material and will be made public upon publication, is documented in the repository. This includes example `hydra` configuration files for the datasets and model selection. The Python package management system `uv` is used to reproduce the virtual environment with all dependencies, and run commands for the most important entry points into the codebase are provided.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We employ LLMs as standard components of the machine learning models under consideration, and describe their usage in detail in Sections 3 and 4. Our development process did not involve LLMs in a non-standard way.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix (Supplementary Materials)

A Sensitivity Analysis

The following analysis gives insights into the stability of the sampling process in our pipeline of Algorithm 1. In particular, we discuss the influence of the following variations on our final mean local estimate:

- In Appendix A.1, we show that with increasing sample and neighborhood sizes, the TwoNN estimates tend to converge in our setting, and already a small portion of samples suffices for reasonable reliability.
- In Appendix A.2 we show that the mean local estimates are stable under applying noise to the embedding vectors.
- In Appendix A.3, we show that for masked language models, both masked and regular token embeddings lead to similar mean estimates.

All embeddings in the following analyses are computed using the regular token embeddings from the last layer of the RoBERTa-base model.

A.1 Dependence on Hyperparameter Choices

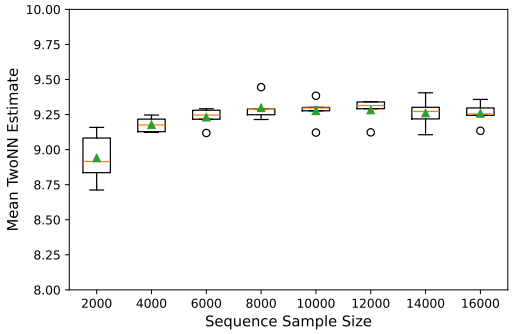
Dataset sequence sub-sampling (dependence on M) Text corpora data splits usually contain different numbers of text sequences, since, for example, the training dataset is larger than the validation or test dataset. Here, we present experiments to show that our method can be made stable in the size of text corpus sequence sub-sample $M \in \mathbb{N}$, and how it depends on the seed of the sequence sub-sample.

Note that in our sub-sampling, we shuffle the dataset split using a given seed, and then truncate to a beginning segment of the specified size. Thus, when comparing different data sequence sub-sampling sizes for a fixed seed, the smaller sequence sub-samples are subsets of the larger ones. In all the experiments in this section, the size of token sub-sample is set to $N = 60\,000$, and the local neighborhood size parameter $L = 128$. Note that for the smallest sequence sub-sample size $M = 2000$, not enough non-padding tokens are necessarily available to reach a token sample size of $N = 60\,000$, but with more sequences, we get this common token space size.

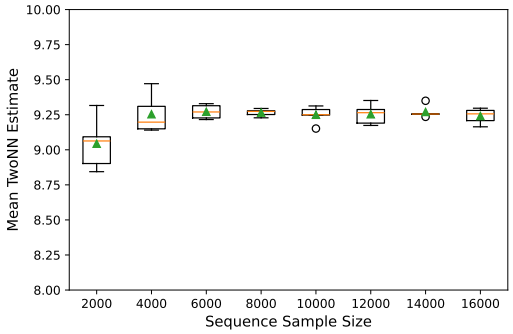
We take sequence sub-samples of each given size with 5 different seeds and select a random token sub-sample for each of these, on which the local TwoNN estimates are based. For the results, see Figure 5 and Figure 6.

For a given sequence sample size, the average of the mean local estimates, calculated across different sampling seeds, remains consistent between the various dataset splits of the same dataset. This

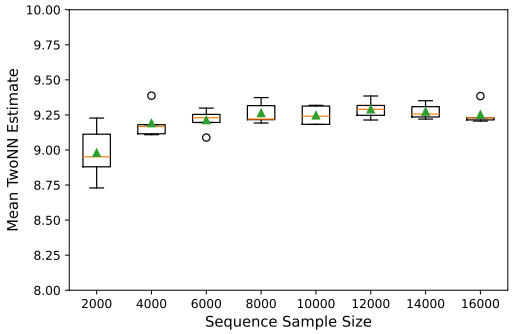
observation indicates that the sampling process is robust to differences in how the data is divided. Furthermore, the estimates stabilize as the sequence sample size increases, suggesting convergence of the local mean estimates with larger samples. Based on these observations, we set M to the values 7000 or 10 000 for the natural language-based tasks in Sections 4.1, 4.3 and 4.4, because it represents a practical compromise between computation effort and stability of the value at that scale. This value provides sufficient sampling density to capture the underlying data distribution comparably across datasets while maintaining computational efficiency.



(a) `MultiWOZ` Training split



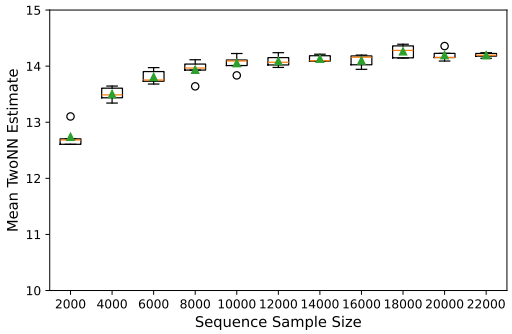
(b) `MultiWOZ` Validation split



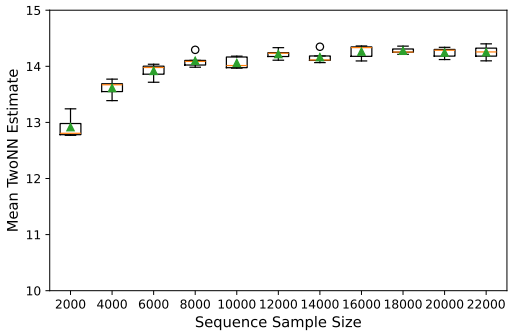
(c) `MultiWOZ` Test split

Figure 5: Boxplots of the mean local TwoNN estimates for different sequence sample sizes M ranging from 2000 to 16 000. We compute the mean estimates for 5 different sequence sub-sampling seeds for the three splits of the `MultiWOZ` dataset. Here and in all subsequent boxplots, the average of the mean estimates is depicted as green triangles, their median is the orange line; outliers are shown as circles.

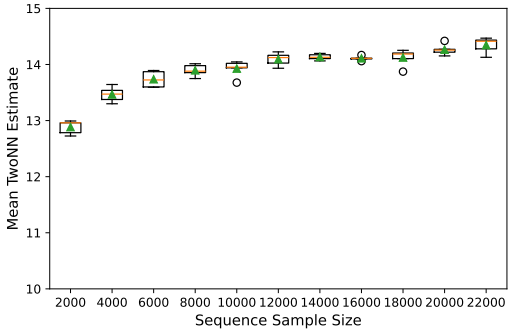
Token sub-sampling (dependence on N) Here, we want to investigate the influence of the token sub-sample size on the local TwoNN estimate. We conduct our analysis on the `MultiWOZ` dataset. Since here we are interested in the effect of the second sub-sampling process, we want to fix the first



(a) `Reddit` Training split



(b) `Reddit` Validation split

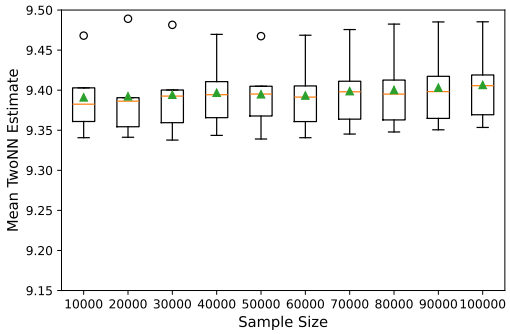


(c) `Reddit` Test split

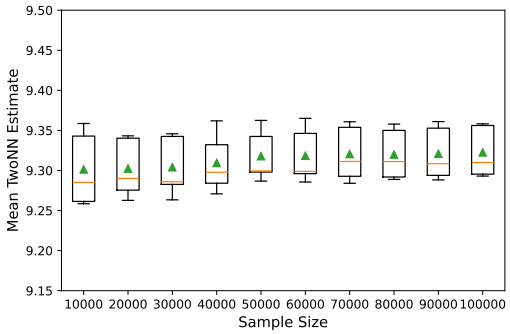
Figure 6: Mean local estimates for different sequence sample sizes M for splits of the `Reddit` dataset, shown for 5 sampling seeds.

sampling step for the sequences: We take the sub-sample of the first $M = 10\,000$ sequences from the `MultiWOZ` training, validation, and test split. From the resulting token embedding spaces, we then sample $100\,000$ points (ignoring padding and end-of-sequence (EOS) tokens). Finally, we compute the local estimates based on the first $N = 10\,000, 20\,000, \ldots, 90\,000, 100\,000$ points in the given sub-sample. Note that we are left with slightly fewer points in the local estimation step for the largest sub-sample size because of our de-duplication step in Algorithm 1. This means that in a given token sub-sample, the beginning segment of the first $10\,000$ points is the same even when increasing the sample size, and we compare the mean of the local estimates of this beginning segment.

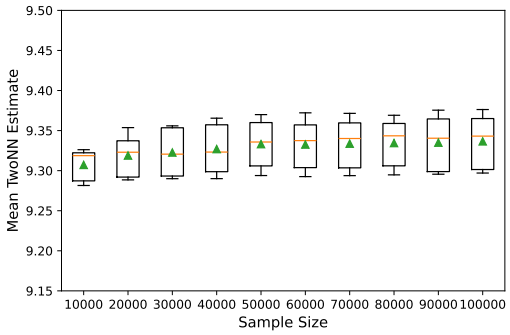
The analysis of the datasets reveals a clear trend in the behavior of the truncated mean values and their associated standard deviations as the sample size increases, see Figure 7. Here, the mean stabilizes quickly, and the standard deviation is comparatively small. This justifies our choice of $N = 60\,000$ in the natural language dataset settings.



(a) `MultiWOZ` Training split



(b) `MultiWOZ` Validation split



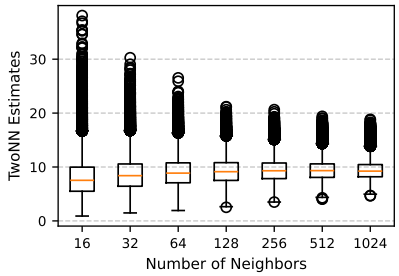
(c) `MultiWOZ` Test split

Figure 7: Distribution of the mean local TwoNN estimates for different cardinalities of token subsamples. We compute the mean estimates for 5 different token sub-sampling seeds and show their distribution for each fixed subsampling cardinality.

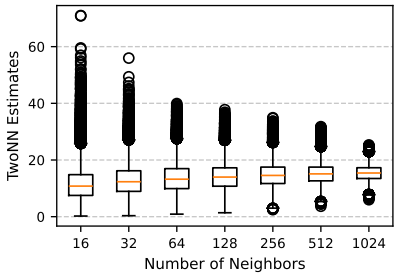
Neighborhood size for local estimates (dependence on L) Having fixed sample sizes M and N, Figure 8 shows the distribution of TwoNN dimension estimates across varying locality scale parameters, represented by the number of neighbors L. A key observation is the relative stability of the estimates as the number of neighbors increases, with smaller locality scales (fewer neighbors) yielding more diverse estimate distributions. This indicates that local properties of the data significantly influence dimension estimates.

The increasing stability with larger neighborhoods suggests that the embedding space exhibits more homogeneity at broader locality scales. However, the variability at smaller scales aligns with the observation in Brown et al. (2023) that in many cases, numerical data is not confined to a single manifold of uniform dimensionality but rather consists of multiple manifolds or regions with varying local dimensions.

Our findings provide evidence that token embedding spaces of this nature do not conform to a single, unique dimension. Instead, they exhibit a complex structure, where local estimates diverge, reflecting the diverse geometric and topological properties within the space. This emphasizes the importance of considering locality when evaluating dimension estimates in such embedding spaces.



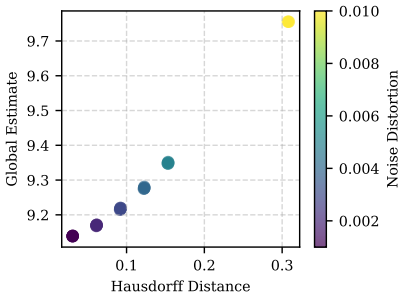
(a) `MultiWOZ` Validation split



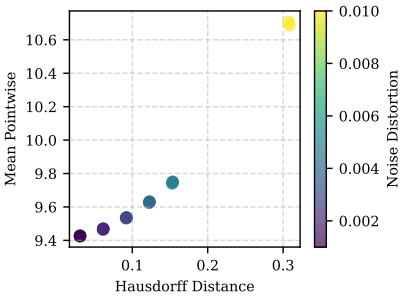
(b) `Reddit` Validation split

Figure 8: Distribution of the local token-wise TwoNN estimates over different locality scales L (number of neighbors). The mean and quartiles of the resulting measure varying over the different tokens in a fixed subsample are shown here. The more global the estimates, the smaller the standard deviation in the resulting distribution.

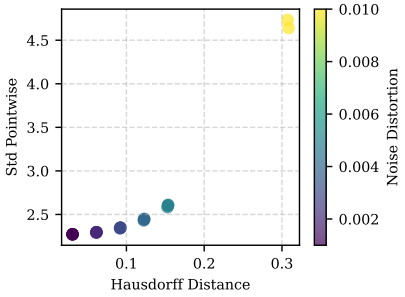
A.2 Noise Analysis



(a) Global TwoNN estimate



(b) Mean of local TwoNN estimates



(c) Standard deviation of local TwoNN estimates

Figure 9: Effect of artificial Gaussian noise on masked token embeddings. On the x-axis, we mark the Hausdorff distance of the noisy embeddings to the clean embeddings, with color denoting the distortion σ in the artificial Gaussian noise.

The following experiments investigate the stability of the TwoNN dimension estimates under small perturbations of point clouds in a realistic setting of high-dimensional language model embedding spaces. The original work in Facco et al. (2017) studies the influence of artificial noise on the

global TwoNN estimator for specific toy datasets (points sampled from a 2-dimensional plane in 20-dimensional space, and points sampled from a 2-dimensional Gaussian distribution wrapped around a Swiss roll). Here, we confirm the stability under noise for our mean local estimates and in the realistic setting of high-dimensional contextual language model embeddings.

We analyze the effect of Gaussian noise on the embeddings from the `MultiWOZ` test set using a fixed set of parameters and varying noise distortions. Specifically, we use $M = 10\,000$ random sequences from the `MultiWOZ` test set and extract the masked token embeddings from the last layer of the RoBERTa base model, which produces 768-dimensional vectors. To compute local estimates, we sub-sample $N = 60\,000$ points from the embeddings and set the neighborhood size to $L = 128$.

Gaussian noise is applied to all dimensions of the embeddings with varying distortion parameters, $\sigma \in \{0.001; 0.002; 0.003; 0.004; 0.01\}$, and multiple random seeds to introduce variability. For each noise level, we compute the approximate Hausdorff distance between the point cloud of the $60\,000$ clean embedding vectors and the $60\,000$ noisy embedding vectors, using the Euclidean distance as the distance metric.

The results are visualized in Figure 9, where we plot the Hausdorff distance against three measures: the global TwoNN-estimate of the noisy point cloud, the mean of the local TwoNN estimates, and the standard deviation of the local TwoNN estimates.

From the plots, we observe the following trends. The Hausdorff distances between the clean and noisy embeddings and the global and the mean local TwoNN estimates increase with larger noise levels, and there is a clear correlation between the two. Notably, for a distortion parameter of $\sigma = 0.01$, the difference in the dimension estimates exceeds one. As expected, the standard deviation of the local TwoNN estimates also increases with higher noise levels, reflecting the increased variability introduced by the noise. Nevertheless, under small noise levels, the mean local estimate remains relatively stable and has lower variability than the global TwoNN estimate.

A.3 Masked Token Embeddings versus Regular Token Embeddings

The plots in Figure 10 compare the TwoNN estimates for regular tokens (left) and masked tokens (right) in the `MultiWOZ` dataset at the last layer. Each box plot represents the distribution of token-wise TwoNN estimates.

The estimates for regular tokens have an interquartile range from approximately 5.5 to 8, with a median near 7. This distribution closely resembles that of masked tokens, which exhibit a similar range and median. However, the broader spread for regular tokens suggests distinct token-level geometric behaviors influenced by their role in the learned representation space. In any case, this justifies the computation of the mean local estimates for the regular token embeddings, which are cheaper to obtain: Every sequence token must be masked individually to retrieve masked embeddings with a corresponding model forward pass. In contrast, for the regular embeddings, a single forward pass of the entire sequence automatically produces embeddings for every input token.

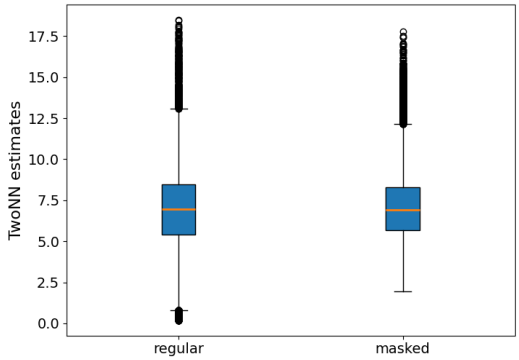


Figure 10: TwoNN estimates for regular and masked tokens in the `MultiWOZ` dataset at the last layer. Both token types show similar distributions.

B Additional Details on Datasets Used

See Table 1 for details on the dataset cards and corresponding split sizes.

Table 1: Dataset sizes for various datasets used in the experiments. Some datasets use pre-determined splits, while others involve random splits.

Dataset	Training Size	Validation Size	Test Size
`MultiWOZ2.1` `https://huggingface.co/datasets/ConvLab/multiwoz21`	113 556	14 748	14 744
`EmoWOZ` `https://paperswithcode.com/dataset/emowoz-1`	66 474	8509	8634
`SGD` `https://huggingface.co/datasets/google-research-datasets/schema_guided_dstc8`	329 964	48 726	84 594
`ICLR 2024` submissions (titles + abstracts)	5764	721	720

`Reddit`
(random 80%-10%-10% split)
`https://huggingface.co/datasets/SocialGrep/one-year-of-tsla-on-reddit`
	174 996	21 874	21 874

`Wikipedia`
(random 80%-10%-10% split)
`https://huggingface.co/datasets/Salesforce/wikitext/viewer/wikitext-103-v1`
	1 441 080	180 135	180 135

C Additional Experimental Results

C.1 Additional Fine-Tuning Results

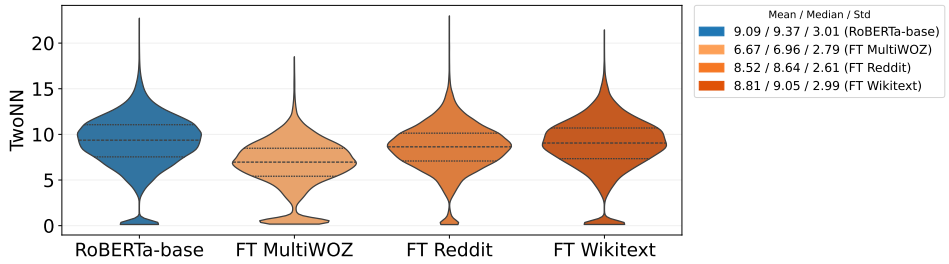
Fine-tuning on unseen data Continuing the discussion in Section 4.1, in Figure 11, we provide additional distribution plots of the local estimates for fine-tuned RoBERTa-base models. Here and in the following, the plots labeled with `FT` correspond to embedding spaces created from the respective base model fine-tuned on the given dataset (`MultiWOZ`, `Reddit`, `Wikipedia`).

Fine-tuning an autoregressive language model Moreover, in Figure 12 we show the distribution of local estimates for the auto-regressive GPT-2-medium model, and their shift under fine-tuning on a causal language modeling task. The GPT-2-medium fine-tuning here is performed with standard parameters on the first $10\,000$ sequences of the given dataset's train portion, and we show the local estimates at the checkpoint after 1200 batches.

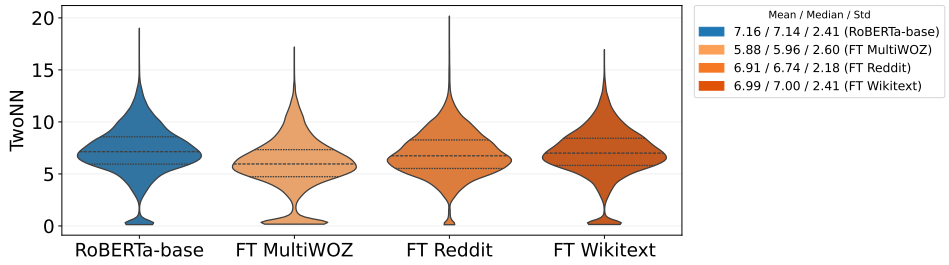
Interesting cases to note are the RoBERTa and GPT-2-medium models fine-tuned on the `Reddit` data: Since the text from the dataset was created in the year 2022, long after the training of RoBERTa and GPT-2-medium finished, this combination illustrates the behavior of a model fine-tuned on data that it has not seen during pre-training. The general observations from Section 4.1 hold for the autoregressive models as well: The LIDs of embeddings originating from the fine-tuning distribution decrease markedly between models, whereas the LIDs for the out-of-distribution corpora remain largely unaffected or have increased.

Local estimates for larger models The geometric perspective and methodology we propose are broadly applicable across model architectures. Local estimates computations for the Phi-3.5-mini-instruct model in Figure 13 and for Llama-3.1-8B in Figure 14 demonstrate that our sub-sampling procedure yields meaningful estimates even for latent spaces whose ambient dimension is in the thousands, a typical setting for modern LLM architectures.

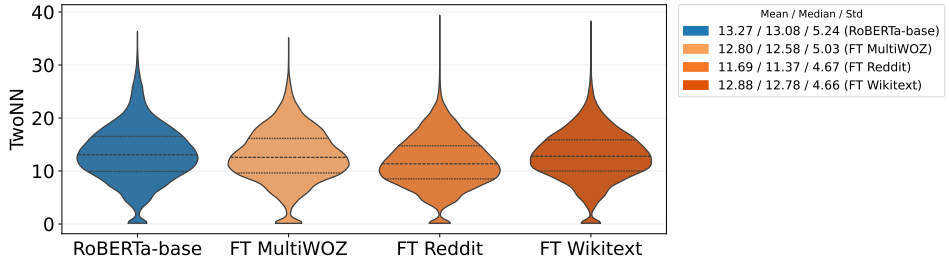
Phi-3.5-mini-instruct is a decoder model with 3.82B parameters and a hidden dimension of 3072 (Abdin et al., 2024), while Llama-3.1-8B is even larger with 8.03B parameters and a hidden dimension of 4096 (AI@Meta, 2024). We fine-tune via LoRA with rank constraint $r = 16$ on a random subsample of $10\,000$ sequences from the `MultiWOZ` and `Reddit` training splits, with other hyperparameters as described in Section 4.1. The violin plots in Figures 13 and 14 show the distribution of the LIDs of the validation split for the model checkpoint after 800 batches.



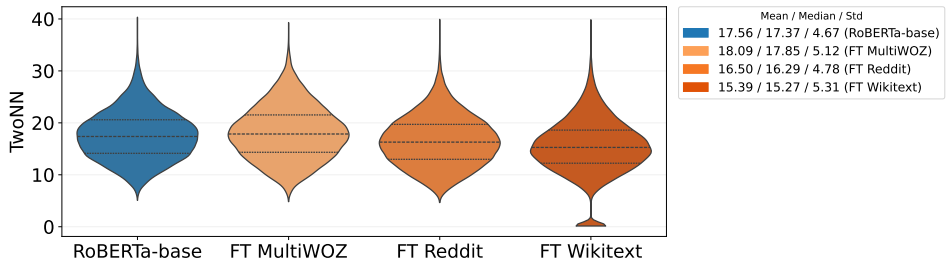
(a) TwoNN estimates on `MultiWOZ` Validation embeddings



(b) TwoNN estimates on `SGD` Validation embeddings



(c) TwoNN estimates on `Reddit` Validation embeddings



(d) TwoNN estimates on `Wikipedia` Validation embeddings

Figure 11: Distribution of LIDs of the masked language model RoBERTa-base and fine-tunes (`FT`).

C.2 Layer-Wise Computation of Local Estimates

Our setup from Section 3 and Algorithm 1 naturally applies to embeddings derived from arbitrary layers of the language model. In particular, the mean local estimates can be compared layer-wise between different checkpoints of a model, as we show in Figure 15. The main observations here are that the drop in mean local dimension on the dataset used for fine-tuning is visible over all the model layers. It appears to be most pronounced in the intermediate and last layers of the model.

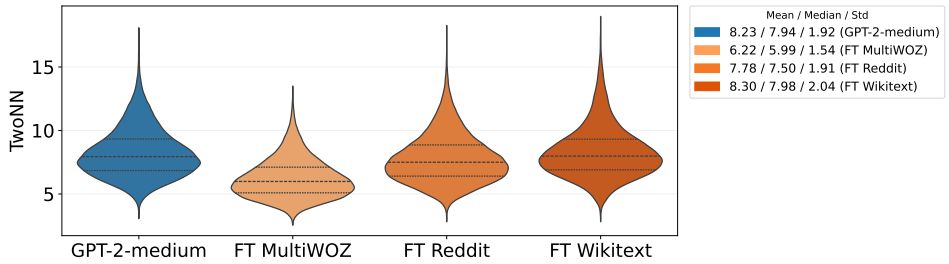
C.3 Additional Grokking Results

This section supplements the grokking experiments in Section 4.2. See Figure 16 for a plot comparing model performance and mean local estimates on the training set, shown separately for the training

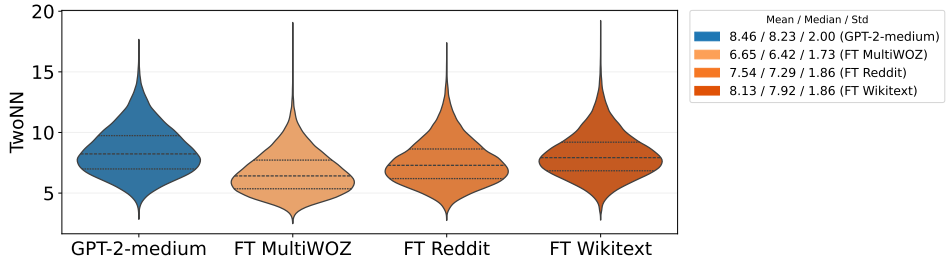
run seeds. This representation is useful, as the timing of the onset of grokking can be very dependent on the model and dataset shuffling seed. It remains true that for individual seeds, the drop in mean local dimension on the training set coincides with the increase in validation accuracy.

C.4 Additional TripPy-R Dialogue State Tracking Results

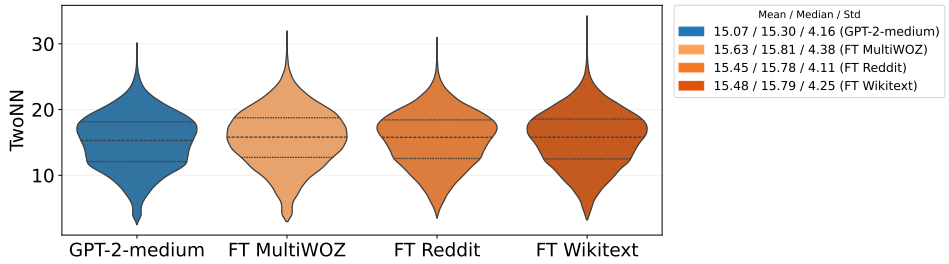
We here show additional results for Section 4.3, where we discuss exhausting model training capabilities. See Figure 17c for longer TripPy-R dialog state tracking runs over 50 epochs. After about 20 training epochs, the mean local estimates stabilize and suggest the convergence of the model performance on the downstream task.



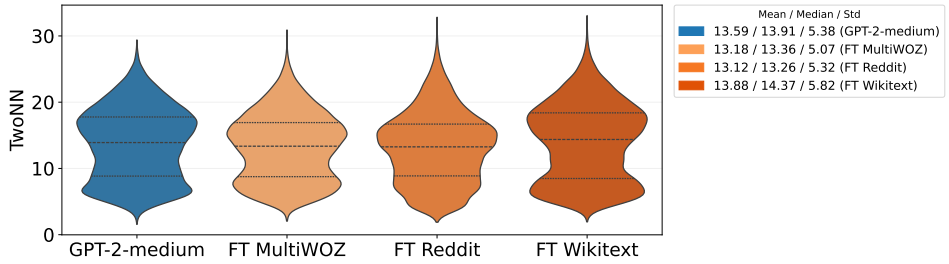
(a) TwoNN estimates on `MultiWOZ` Validation embeddings



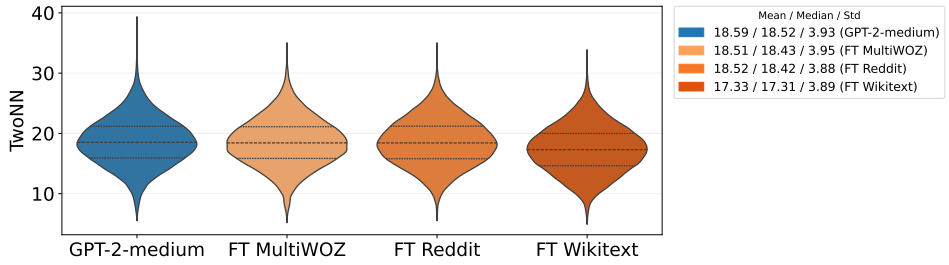
(b) TwoNN estimates on `SGD` Validation embeddings



(c) TwoNN estimates on `ICLR 2024` Validation embeddings

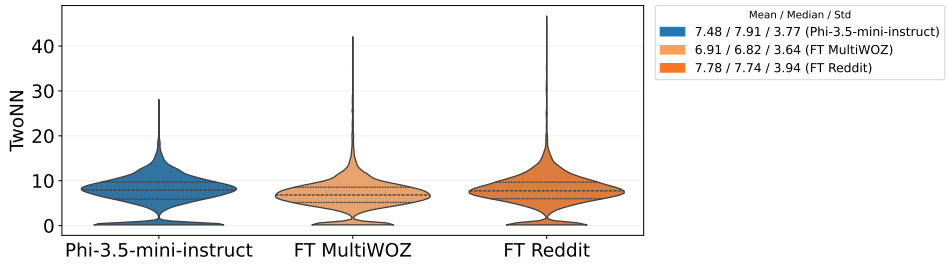


(d) TwoNN estimates on `Reddit` Validation embeddings

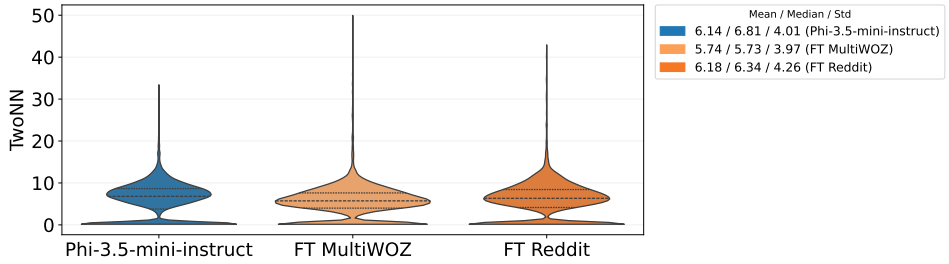


(e) TwoNN estimates on `Wikipedia` Validation embeddings

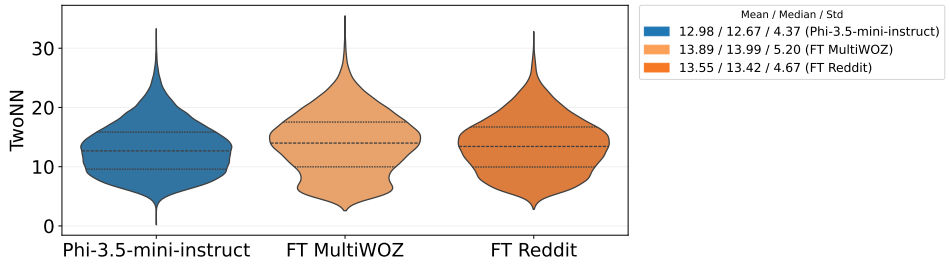
Figure 12: Distribution of LIDs of the autoregressive GPT-2-medium model and fine-tunes (`FT`).



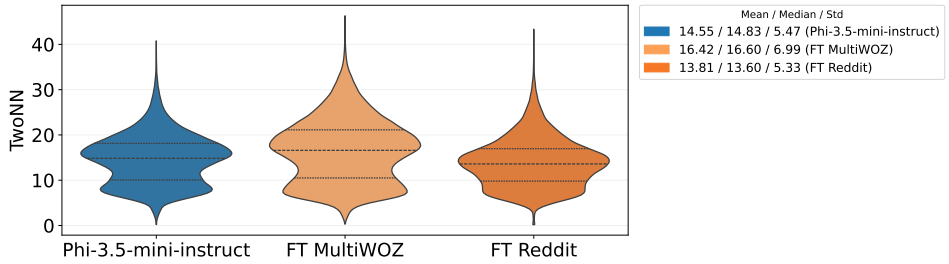
(a) TwoNN estimates on `MultiWOZ` Validation embeddings



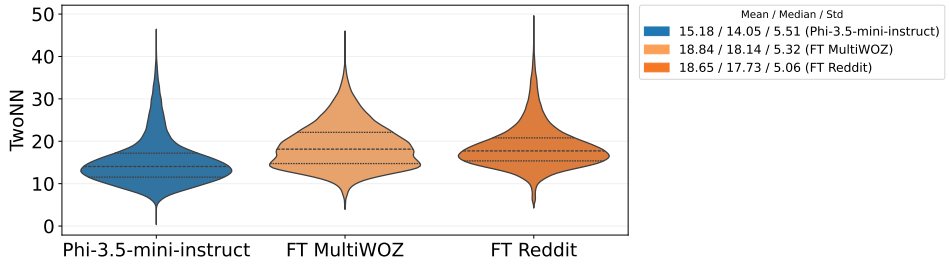
(b) TwoNN estimates on `SGD` Validation embeddings



(c) TwoNN estimates on `ICLR 2024` Validation embeddings

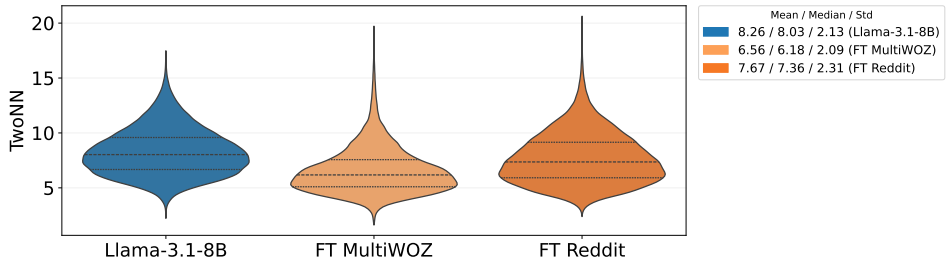


(d) TwoNN estimates on `Reddit` Validation embeddings

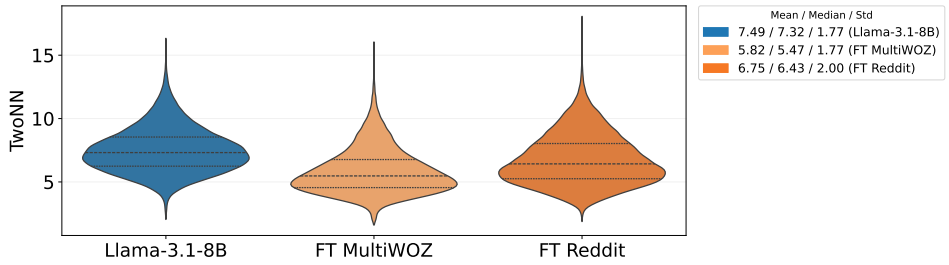


(e) TwoNN estimates on `Wikipedia` Validation embeddings

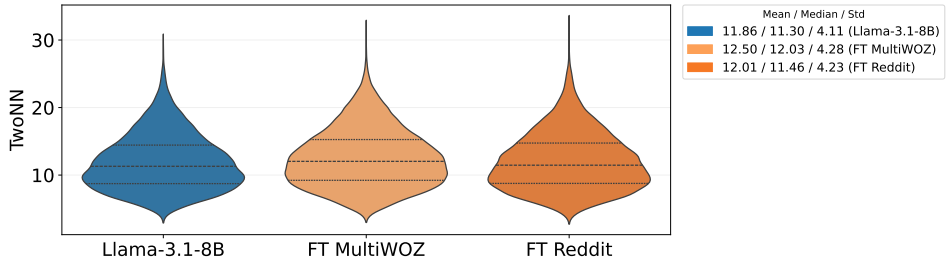
Figure 13: Distribution of LIDs of the autoregressive Phi-3.5-mini-instruct model and fine-tunes (FT).



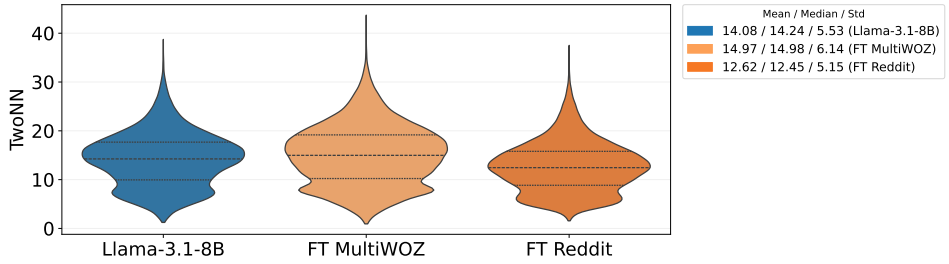
(a) TwoNN estimates on `MultiWOZ` Validation embeddings



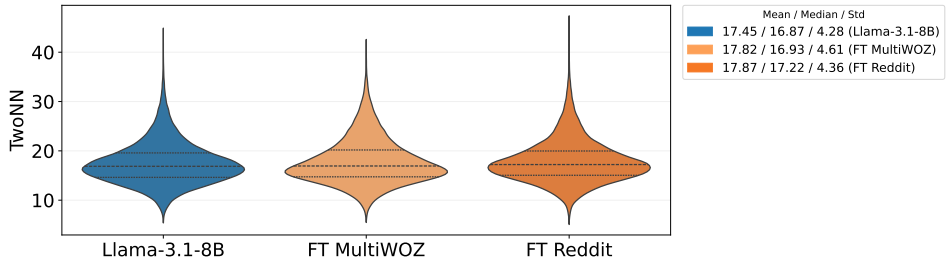
(b) TwoNN estimates on `SGD` Validation embeddings



(c) TwoNN estimates on `ICLR 2024` Validation embeddings



(d) TwoNN estimates on `Reddit` Validation embeddings



(e) TwoNN estimates on `Wikipedia` Validation embeddings

Figure 14: Distribution of LIDs of the autoregressive Llama-3.1-8B model and fine-tunes (`FT`).

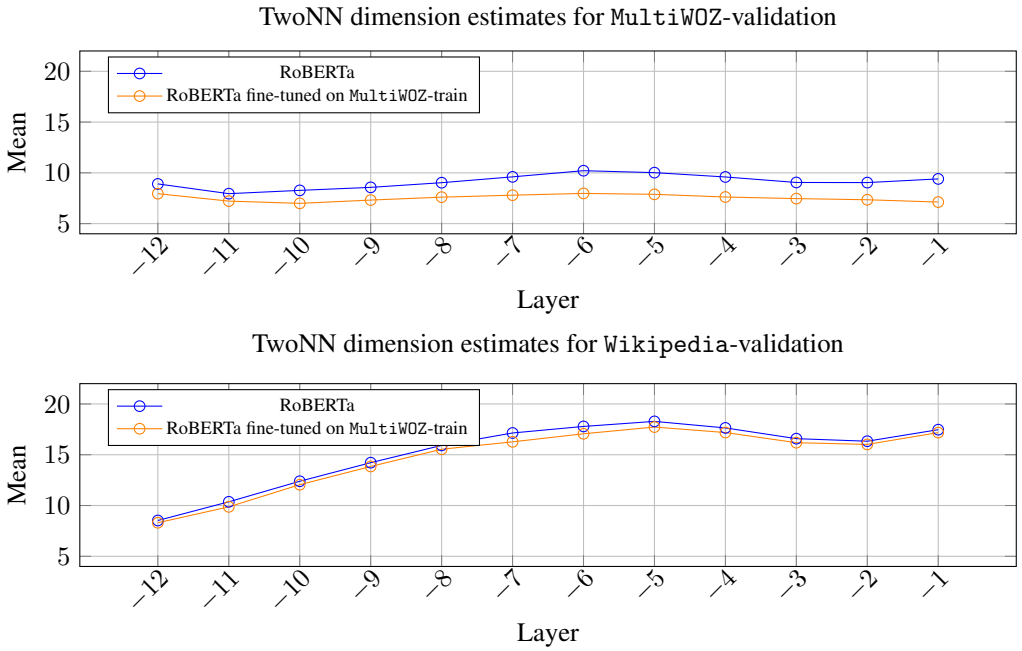
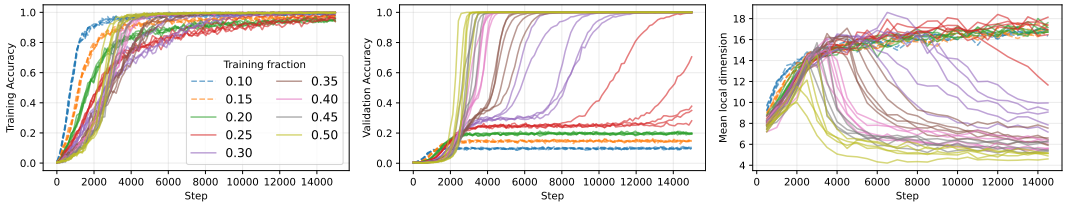
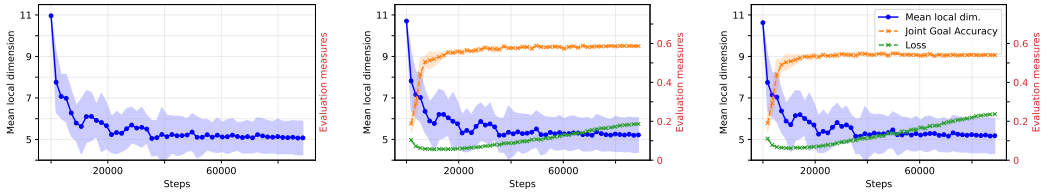


Figure 15: Comparison of the mean LID development through the different layers of RoBERTa-base and a fine-tuned variant. The LID of embeddings from the distribution used for fine-tuning (here `MultiWOZ`) differs significantly between the two models, whereas the LID of other data distributions (here `Wikipedia`) is indistinguishable.



(a) **Training** accuracy (b) Validation accuracy (c) **Training** mean local estimates

Figure 16: Training a model on addition mod $p = 197$ with different training data fraction selected from $\{0.1; 0.15; 0.2; 0.25; 0.3; 0.4; 0.5\}$. Five seeds per configuration are shown separately, demonstrating that the onset of grokking and the drop in mean local dimension vary greatly depending on the specific training run. The plots show the development over 15 000 batches. The mean local estimates are computed on the **training split** for $N = 3000$; $L = 64$.



(a) `MultiWOZ` Training split (b) `MultiWOZ` Validation split (c) `MultiWOZ` Test split

Figure 17: Longer TripPy-R training runs over 50 epochs with a linear learning rate schedule. Shown are the mean and standard deviation of the measures over 4 seeds of the training runs.