# Probing Neural Topology of Large Language Models

**Yu Zheng**[1] **Yuan Yuan**[2] **Yong Li**[2] **Paolo Santi**[1]
[1]Masssachusetts Institute of Technology, Cambridge, MA USA
[2]Tsinghua University, Beijing, China
`yu_zheng@mit.edu`

## Abstract

Probing large language models (LLMs) has yielded valuable insights into their internal mechanisms by linking neural representations to interpretable semantics. However, how neurons functionally co-activate with each other to give rise to emergent capabilities remains largely unknown, hindering a deeper understanding and safer development of LLMs. In this work, we introduce graph probing, a method for uncovering the functional connectivity topology of LLM neurons and relating it to language generation performance. By analyzing internal neural graphs across diverse LLM families and scales, we discover a universal predictability of next-token prediction performance using only neural topology. This predictability is robust even when retaining just 1% of neuron connections or probing models after only 8 pretraining steps, highlighting the sparsity and early emergence of topological patterns. Further graph matching analysis suggests that, despite significant distinctions in architectures, parameters, and training data, different LLMs develop intricate and consistent neural topological structures that may form the foundation for their language generation abilities.

## 1 Introduction

Large language models (LLMs) exhibit remarkable generative capabilities [53, 50, 19, 47, 22], yet our understanding of how they succeed and what they have learned remains limited [45]. *Probing*, which extract interpretable features from neural activations [1], has emerged as a powerful approach for reverse-engineering LLMs [5, 24]. For instance, Gurnee *et al.*[24] showed that LLMs encode a compact world model of space and time using linear regression probes. Unsupervised probing, such as sparse auto-encoders [13, 18, 41, 33, 39], have further revealed dictionaries of interpretable, mono-semantic concepts [25] and even causal circuits [36], corresponding to directions in neural latent space. While these advances shed light on the semantics of individual neurons and representations [45], much less is known about how neurons are functionally connected, *i.e.* the neural topology, which is believed to play an essential role in the emergence of intelligence [42, 3].

Recent studies have drawn compelling parallels between neurons in LLMs and those in the human brain [49, 44, 11, 31, 42, 38, 51, 8, 46, 34], revealing shared properties such as spatial-functional organization [31, 42] and left lateralization [8]. Neural activations at internal layers of LLMs have also been shown to reliably predict human brain responses given the same linguistic stimuli [44, 51, 35]. However, these efforts primarily focus on static neural representations of LLMs, while overlooking the key aspect of temporal and functional neural topology that has been studied in neuroscience for decades [4, 3, 15]. Moreover, although analogies between LLMs and human brains are insightful [49, 21], few works explicitly connect these findings to LLMs' language generation performance, which is one of the primary indicators of an LLM's intelligence.

In this work, we introduce *graph probing*, a novel approach for investigating the functional connectivity of neurons in LLMs and its relationship to language generation performance. By analyzing neural activity time series as LLMs process text token by token, we compute temporal and functional

Preprint. Under review.

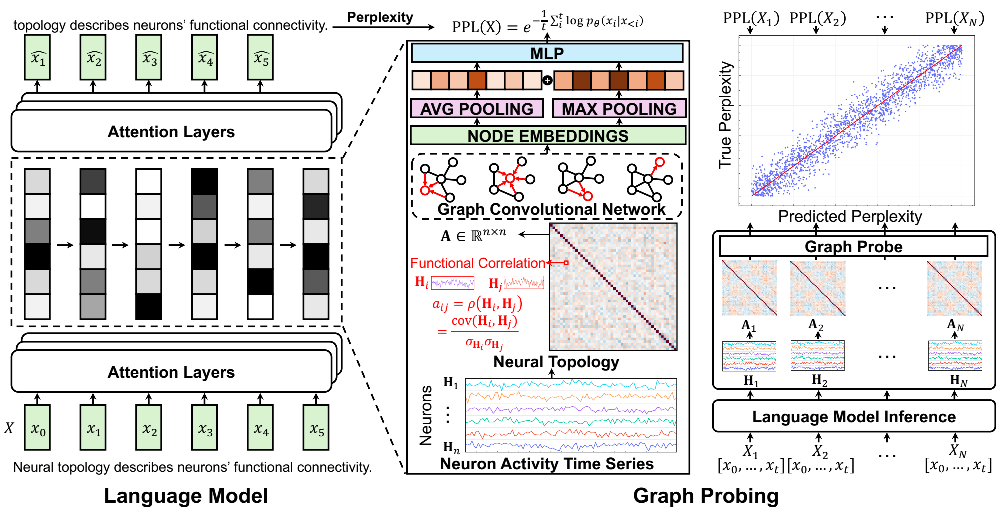
topology describes neurons' functional connectivity.
Perplexity
PPL(X) = $e^{-\frac{1}{t}\sum_i^t \log p_\theta(x_i|x_{<i})}$
Attention Layers
MLP
AVG POOLING
MAX POOLING
NODE EMBEDDINGS
Graph Convolutional Network
A ∈ ℝ^{n×n}
Functional Correlation
$a_{ij} = \rho(H_i, H_j) = \frac{cov(H_i, H_j)}{\sigma_{H_i}\sigma_{H_j}}$
Neural Topology
Neurons
Neuron Activity Time Series
Neural topology describes neurons' functional connectivity.
Language Model
PPL($X_1$) PPL($X_2$) ⋯ PPL($X_N$)
True Perplexity
Predicted Perplexity
Graph Probe
Language Model Inference
Graph Probing

Figure 1: An overview of our graph probing method. We extract the time series of neuron activities for each attention layer in an LLM as it processes text token by token. We then compute temporal and functional correlations between neural activations to obtain topological connectivity graphs of neurons. GNN-based probes are trained to predict the perplexity of the auto-regressive prediction for the input token sequence.

correlations between neurons to construct dynamic neural graphs. Using this large-scale dataset of text-induced neural topology, we train graph neural network (GNN) [29, 14] as probes to predict LLMs' accuracy in auto-regressively generating the corresponding text. In essence, graph probing connects the micro-level topology of how neurons are connected given a token sequence, to the macro-level performance of how well LLMs predict these tokens, offering a new lens to study the emergent capabilities of LLMs. Our method is summarized in Figure 1 and described in Section 2.

We then apply our graph probing framework to comprehensively analyze the neural topology of LLMs through extensive experiments. First, we demonstrate that auto-regressive language generation performance can be reliably predicted using only the neural connectivity graph. This predictability holds universally across LLM families and scales, with empirical results spanning GPT [40], Pythia [7], and Qwen [54], ranging from millions to billions of parameters (Section 3.1). Next, we show that these neural topologies are (1) *sparse*, given strong predictive performance when preserving only 1% of neuron connections (Section 3.2), (2) *non-linearly* related to language generation performance, as non-linear graph probing significantly outperforms linear baselines (Section 3.3), and (3) *early emerging* with predictive topological structures arising within just 8 steps of LLM pretraining (Section 3.4). Finally, we use our probes to perform cross-model contrastive graph matching, revealing that distinct LLMs converge toward similar internal neural topologies, suggesting shared underlying principles despite large discrepancies in architectures, parameters, and training data (Section 4). While not without limitations, we expect graph probing to provide valuable insights into the inner workings of LLMs and to guide their future development in an interpretable and safe manner.

## 2 Graph Probing

**Neural Topology.** To construct neural graphs from LLMs, we draw inspiration from neuroscience where functional brain networks are derived from temporal correlations in fMRI or EEG activation signals [3, 52, 10], as shown in Figure 1. Formally, given an LLM composed of stacked attention layers, the neural topology is constructed as follows:

$$\textbf{Neural Activity:}\quad \mathbf{H} = \texttt{HIDDEN\_STATE}(\texttt{LLM}(X)) = [\mathbf{h_0}, \mathbf{h_1}, \ldots, \mathbf{h_t}] \in \mathbb{R}^{n\times t}, \tag{1}$$

$$\textbf{Neural Topology:}\quad \mathbf{A} = \big(a_{ij}\big) \in \mathbb{R}^{n\times n}, \tag{2}$$

$$a_{ij} = \rho(\mathbf{H}_{i,:}, \mathbf{H}_{j,:}) = \frac{\sum_{k=0}^{t}(\mathbf{H}_{i,k} - \overline{\mathbf{H}_{i,:}})(\mathbf{H}_{j,k} - \overline{\mathbf{H}_{j,:}})}{\sqrt{\sum_{k=0}^{t}(\mathbf{H}_{i,k} - \overline{\mathbf{H}_{i,:}})^2}\sqrt{\sum_{k=0}^{t}(\mathbf{H}_{j,k} - \overline{\mathbf{H}_{j,:}})^2}}, \tag{3}$$

where neurons at each layer produce a time series of hidden states $\mathbf{H}$ as the model processes a token sequence $X = [x_0, x_1, \ldots, x_t]$, and the temporal co-activation patterns among neurons define their

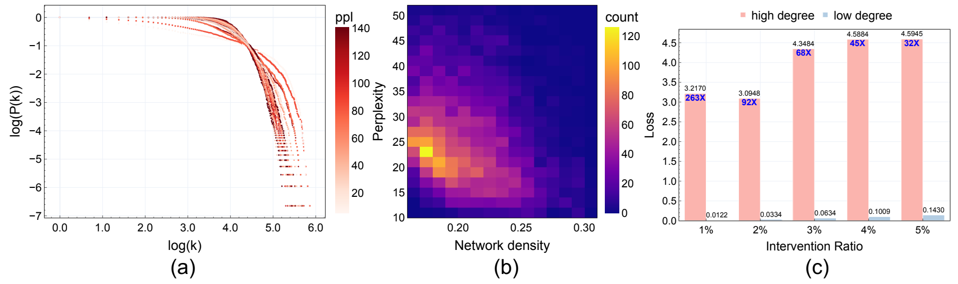

Figure 2: Analysis of topological properties and their relationship to language generation performance. (a) Degree distributions of neural connectivity graphs induced by 10 different text inputs, with perplexity indicated by color darkness. (b) 2D histogram showing the relationship between token prediction perplexity and network density across a text dataset. (c) Perplexity increase resulting from interventions on neurons with high- vs. low-degree. Specifically, we disable the top/bottom $k\%$ of neurons by setting their activations to zero during inference.

*functional connectivity*. We capture this through a complete $n \times n$ weighted connectivity matrix $\mathbf{A}$, where each node corresponds to a neuron and each edge weight $a_{ij}$ represents the Pearson correlation coefficient between the activation time series of two neurons [3, 15]. Meanwhile, the LLM is trained for auto-regressive next-token prediction, with performance commonly measured by perplexity [6] which corresponds to the exponentiated average negative log-likelihood over the token sequence:

$$\textbf{Perplexity:}\quad \texttt{PPL}(X) = \exp\left\{-\frac{1}{t}\sum_{i=1}^{t}\log p_\theta(x_i \mid x_{<i})\right\}. \tag{4}$$

The graph is dynamically induced by the specific token sequence, and our goal is to investigate whether the text-responsive neural topology is linked to how well the model predicts the text.

A trivial approach to characterizing graphs is to compute heuristic topological properties commonly used in network analysis [2], such as degree distribution[1] and network density[2]. However, these basic properties tend to exhibit no clear or intuitive relationship with the model's performance–samples with drastically different perplexities can display similar degree distributions or network densities, as shown in Figure 2(a-b) obtained by feeding a text dataset (details in Section 3) into the GPT-2 model [40]. Nevertheless, when we intervene on neurons with high degree by forcing their activations to zero during inference, we observe a substantial increase in perplexity (up to 263$\times$), compared to interventions on low-degree neurons (Figure 2(c)). This suggests that while neural topology indeed plays a crucial role in determining next-token prediction performance, the relationship is highly non-trivial and cannot be adequately explained by simple, handcrafted topological metrics.

**Probing with GNN.** To better characterize the complex interplay between neural topology and perplexity, we propose *graph probing*, a method that learns representations of neural connectivity graphs to predict corresponding language generation performance, as illustrated in Figure 1. Specifically, we adopt a GNN-based probe that encodes each node by aggregating neighborhood information through convolutional message passing on the graph [29, 14]. We employ the ReLU activation function [16] between graph convolution layers and use both average and maximum pooling to summarize node-level embeddings into a graph-level representation. Given a connectivity matrix $\mathbf{A}$ induced by feeding a tokenized sequence $X$ to an LLM, where each element $a_{ij}$ denotes the functional connectivity (Pearson correlation coefficient) between neurons $i$ and $j$, our probe produces the graph representation $\mathbf{z}$ as follows:

$$\mathbf{\Phi}^L = \texttt{ReLU}(\mathbf{A}\mathbf{\Phi}^{L-1}\mathbf{\Theta}^L), \ldots, \mathbf{\Phi}^0 \in \mathbb{R}^{n\times d}, \tag{5}$$

$$\mathbf{z} = \texttt{AVG\_POOLING}\{\mathbf{\Phi}^L_{1,:}, \ldots, \mathbf{\Phi}^L_{n,:}\} \parallel \texttt{MAX\_POOLING}\{\mathbf{\Phi}^L_{1,:}, \ldots, \mathbf{\Phi}^L_{n,:}\}, \tag{6}$$

where $\mathbf{\Phi}^0 \in \mathbb{R}^{n\times d}$ denotes the initial learnable node embeddings, $\mathbf{\Theta}^l \in \mathbb{R}^{d\times d}$ is the weight matrix of the $l$-th layer in the GNN with $L$ total layers, and $d$ is a hidden dimensionality hyperparameter. We

[1]The degree of a node is the sum of its connectivity strengths: $d_i = \sum_j \|a_{ij}\|$, where we take the absolute value since correlations can be either positive or negative within the range $[-1, 1]$.

[2]Network density is defined as the total connectivity strength: $\sum_{i,j} \|a_{i,j}\|$, again using the absolute value of Pearson correlation coefficients.

then feed the graph representation $\mathbf{z} \in \mathbb{R}^{2d}$ into a multi-layer perceptron (MLP) [43] to predict the perplexity associated with the input tokenized sequence $X$:

$$\hat{p} = \mathbf{W}_2 \cdot \texttt{ReLU}(\mathbf{W}_1 \cdot \mathbf{z}^T), \tag{7}$$

where $\hat{p}$ is the predicted perplexity, and $\mathbf{W_1} \in \mathbb{R}^{2d \times d}, \mathbf{W_2} \in \mathbb{R}^{d \times 1}$ are learnable weights of the MLP. Our graph probe is trained to minimize the mean squared error (MSE) between predicted and true perplexities over a dataset of tokenized sequences $\mathbf{X} = \{X_1, \ldots, X_N\}$:

$$\mathcal{L}(\mathbf{X}) = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{p}_i - \texttt{PPL}(X_i)\right)^2 . \tag{8}$$

For details of graph probing including hyper-parameter and computer resources, see Appendix A.

## 3 Results

With graph probing, we aim to understand whether the auto-regressive token-prediction performance of LLMs is associated with their probed internal neural topologies. If such a relationship exists, does it generalize across different model families and scales? Furthermore, how and when is the dependence between these two seemingly distant aspects established?

**LLMs.** In our experiments, we train graph probes on neural topology derived from three families of LLMs, each spanning across different sizes. Specifically, we evaluate GPT2 [40] (GPT2, GPT2-medium, GPT2-large), Pythia [7] (160M, 410M, 1.4B, 2.8B, 6.9B, 12B), and Qwen2.5 [54] (0.5B, 3B, 7B, 14B). Details of the experimented LLMs are provided in Appendix B.

**Datasets.** To enable our study, we construct neural connectivity graphs using the text corpora on which LLMs were pretrained. Specifically, we use the Pile dataset [17] for Pythia models, and the OpenWebText dataset [20] for GPT2 and Qwen2.5 models. To ensure consistent temporal resolution, we control the length of neural activity time series to fall between 256 and 1024 tokens by merging consecutive sentences as needed. For each token sequence, we perform LLM inference to compute its perplexity and simultaneously extract hidden state time series to generate the corresponding neural topology. For each model, we construct a probing dataset comprising about 10,000 graph–perplexity pairs. Further details on dataset construction are provided in Appendix C.

**Evaluation.** We split the dataset into training and test sets using an 8:2 ratio. Having learned graph probes on the training set, we evaluate their out-of-sample graph regression performance on the test set, which reveals the extent to which micro-level neural topology is predictive of macro-level language generation ability. To quantify the effectiveness of graph probing, we report standard regression metrics on our test data, including mean squared error (MSE), mean absolute error (MAE), coefficient of determination ($R^2$), Pearson correlation ($\rho_p$), and Spearman rank correlation ($\rho_s$).

### 3.1 Predictability

We show our graph probing results in Figure 3, which exhibit consistently strong predictability across all three LLM families. Particularly, graph probing achieves 0.95 in $\rho_p$ and $\rho_s$, and 0.90 in $R^2$ on

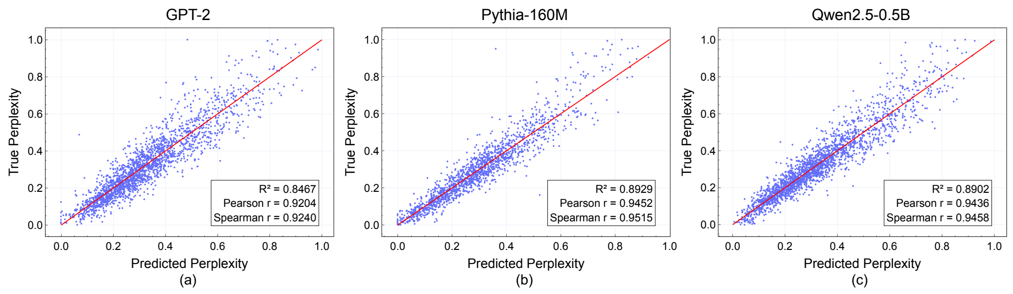

Figure 3: Out-of-sample performance of graph probing on the test set for (a) GPT-2 (b) Pythia-160M (c) Qwen2.5-0.5B. The correlation between the perplexity predicted by graph probing and the ground-truth perplexity reflects how well LLM performance can be inferred from neural topology.

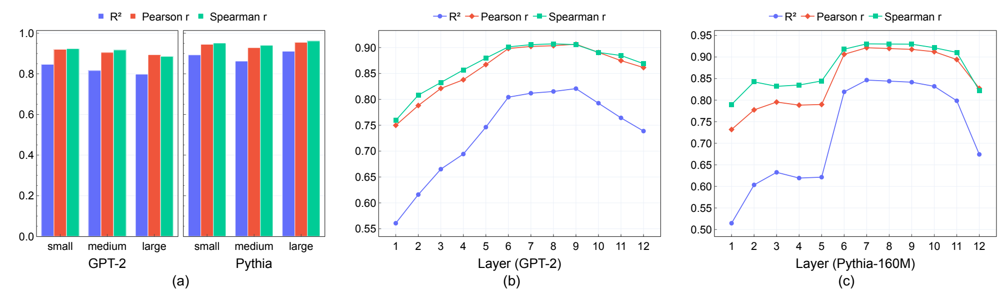

Figure 4: (a) Out-of-sample performance of graph probing on LLMs of different sizes including GPT2, GPT2-medium, GPT2-large, Pythia-160M, Pythia-410M, and Pythia-1.4B. (b-c) Out-of-sample performance of graph probing on each of the 12 layers in (b) GPT-2 and (c) Pythia-160M.

both Pythia-160M and Qwen2.5-0.5B. For GPT-2, graph probing reliably predicts perplexity with $\rho_p$ and $\rho_s$ exceeding 0.92 and an $R^2$ score of 0.85. The high predictability is observed across LLMs of varying scales. As shown in Figure 4(a), models ranging from 124M to 1.4B parameters in the GPT-2 and Pythia families consistently achieve strong topology-perplexity correlations, with $\rho_p$ and $\rho_s$ ranging from 0.89 to 0.96.

We further train probes on neural connectivity graphs derived from different layers of GPT-2 and Pythia-160M. Figures 4(b–c) show that neural topology at every layer is predictive of language generation performance, with the strongest predictability ($\rho_p$ and $\rho_s$ exceeding 0.91 and 0.93) found in the middle layers of both models. This observation is also aligned with previous probing studies that identify the middle layers of LLMs as particularly informative and semantically rich [24, 51]. The above graph probing experiments reveal that neural topology is a universal and strong predictor of LLMs' language generation capabilities.

## 3.2 Sparsity

The universal predictability observed so far is based on complete graphs, which are dense $n \times n$ connectivity matrices that capture pairwise functional correlations between all neurons. Yet not all connections contribute equally to language generation performance, as suggested by our earlier intervention analysis comparing high- and low-degree nodes (Figure 2(c)). To explore this further, we examine the distribution of edge weights in Figure 5(a), which reveals that the majority of connections are weak, with absolute correlation values near zero. This motivates us to investigate

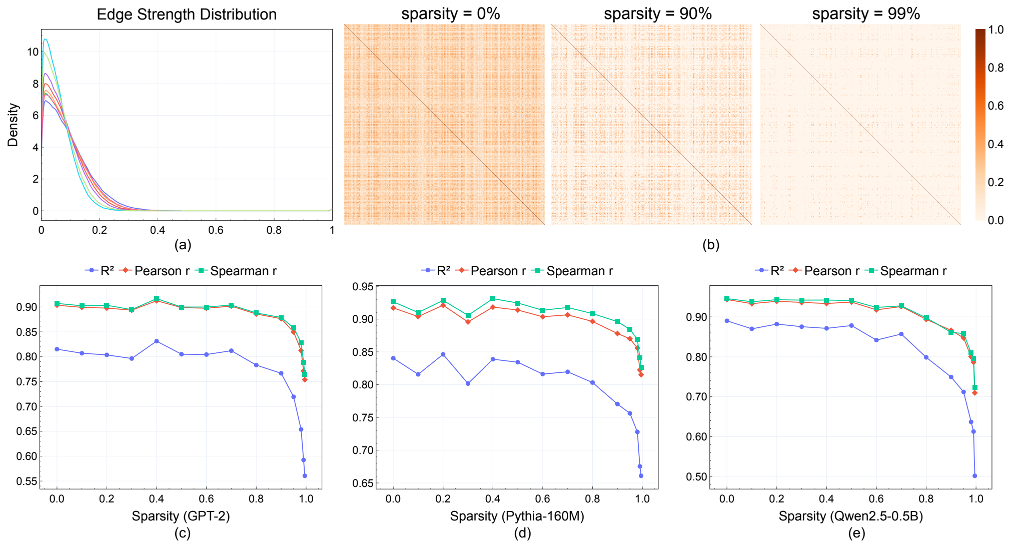

Figure 5: (a) Edge weight (absolute value) distribution for 10 neural graphs of GPT-2. (b) Neural topology under different levels of sparsity, where weak connections are pruned by thresholding on the absolute value of edge weight. (c-e) Out-of-sample graph probing performance on neural connectivity graphs of different sparsity levels for (c) GPT-2 (d) Pythia-160M and (e) Qwen2.5-0.5B.

whether perplexity can still be predicted from sparse graphs with weakly correlated edges pruned out by thresholding, which is commonly employed in human brain network construction [3], as illustrated in Figure 5(b). To evaluate this, we train graph probes on neural topology with varying levels of sparsification (Figures 5(c-e)). Surprisingly, the predictive performance remains remarkably stable even after removing up to 90% of the edges, with minimal degradation. Notably, even under extreme sparsity where only 1% of the original edges are retained, the neural topology still enables effective prediction of perplexity, achieving above 0.71 correlations in $\rho_p$ and $\rho_s$.

Graph probing on complete neural topology becomes computationally prohibitive as the LLM size increases, due to the quadratic number of edges that directly impacts the computational cost in both time and memory. For instance, while complete graph probing is feasible for Pythia-160M with 768 neurons and 0.6M edges per layer, the number of edges in Pythia-12B–comprising 5,120 neurons per layer–explodes to over 26M per graph. Fortunately, the above experiments suggest that most of the predictive signal resides in a small subset of strong connections, making it possible to significantly reduce the number of edges while preserving nearly all critical topological information. Leveraging this insight, we scale up graph probing to much larger models by operating on sparsified neural topology. While our earlier results focused on models with fewer than 1.4B parameters, we now train probes on sparse graphs derived from LLMs with up to 14B parameters. As shown in Table 1, graph probing continues to exhibit strong regression performance across all six models, achieving a maximum accuracy of over 0.92 $R^2$ and over 0.96 $\rho_s$, providing compelling evidence that the relationship between neural topology and language modeling performance is universal across model sizes. Complete results of the predicted and groundtruth perplexities for all models are provided in Appendix D.

Table 1: Out-of-sample graph probing performance on sparse neural topologies derived from LLMs containing 2.8B to 14B parameters.

| **LLM (Sparsity)** | $\mathbf{R^2}$ | $\rho_p$ | $\rho_s$ |
|---|---|---|---|
| Pythia-2.8B (90%) | 0.8995 | 0.9484 | 0.9592 |
| Pythia-6.9B (90%) | 0.9210 | 0.9599 | 0.9670 |
| Pythia-12B (99%) | 0.8974 | 0.9480 | 0.9527 |
| Qwen2.5-3B (90%) | 0.7051 | 0.8426 | 0.8372 |
| Qwen2.5-7B (90%) | 0.7699 | 0.8789 | 0.8823 |
| Qwen2.5-14B (95%) | 0.8249 | 0.9086 | 0.9167 |

## 3.3 Non-linearity

We next examine the complexity of the relationship between neural topology and language generation performance. Specifically, we train graph probes with varying capabilities, under the hypothesis that more expressive models can capture deeper and more nuanced topological patterns. We focus on two key factors: (1) the *linearity* of the probe, controlled by enabling or disabling the non-linear ReLU activation function, and (2) the *receptive field* of the probe, determined by the number of graph convolutional layers, $L$. Tables 2 and 3 show the out-of-sample regression performance of different probe configurations on complete and sparse neural topology, respectively. We find that linear probes, though still retaining considerable predictive power, consistently underperform compared to non-linear probes, with a 31.4% increase in MSE and a drop of more than 0.05 in $R^2$. This suggests that the relationship between neural topology and language generation performance is inherently non-linear. We also find that 1-hop GNNs perform best on complete graphs, whereas 2-hop GNNs outperform on sparse graphs, which is reasonable since the loss of local connectivity by sparsification can be partially compensated by incorporating information from more distant neighbors, making a larger receptive field beneficial. Results on more probe configurations can be found in Appendix E.

Table 2: Out-of-sample performance using different probes on complete neural topologies.

| **LLM** | **Graph Probing** | **MSE** $\downarrow$ | **MAE** $\downarrow$ | $\mathbf{R^2}$ $\uparrow$ | $\rho_p$ $\uparrow$ | $\rho_s$ $\uparrow$ |
|---|---|---|---|---|---|---|
| GPT-2 | 1-hop linear | 0.0067 | 0.0629 | 0.7966 | 0.8930 | 0.8926 |
| | 1-hop non-linear | **0.0051** | **0.0529** | **0.8467** | **0.9204** | **0.9240** |
| | 2-hop non-linear | 0.0061 | 0.0563 | 0.8152 | 0.9036 | 0.9055 |
| Pythia-160M | 1-hop linear | 0.0036 | 0.0449 | 0.8894 | 0.9431 | 0.9475 |
| | 1-hop non-linear | **0.0035** | **0.0421** | **0.8929** | **0.9452** | **0.9515** |
| | 2-hop non-linear | 0.0050 | 0.0498 | 0.8465 | 0.9215 | 0.9305 |
| Qwen2.5-0.5B | 1-hop linear | 0.0046 | 0.0506 | 0.8596 | 0.9272 | 0.9264 |
| | 1-hop non-linear | **0.0036** | **0.0438** | **0.8902** | **0.9436** | **0.9458** |
| | 2-hop non-linear | 0.0051 | 0.0512 | 0.8447 | 0.9193 | 0.9221 |

Table 3: Out-of-sample performance using different probes on sparse neural topologies where top 10% functional connectivity are reserved (90% sparsity).

| LLM | Graph Probing | MSE ↓ | MAE ↓ | $R^2$ ↑ | $\rho_p$ ↑ | $\rho_s$ ↑ |
|---|---|---|---|---|---|---|
| GPT-2 | 1-hop linear | 0.0098 | 0.0756 | 0.7034 | 0.8443 | 0.8439 |
|  | 1-hop non-linear | 0.0085 | 0.0690 | 0.7419 | 0.8627 | 0.8666 |
|  | 2-hop non-linear | **0.0077** | **0.0640** | **0.7667** | **0.8765** | **0.8789** |
| Pythia-160M | 1-hop linear | 0.0096 | 0.0741 | 0.7061 | 0.8431 | 0.8627 |
|  | 1-hop non-linear | 0.0087 | 0.0673 | 0.7329 | 0.8566 | 0.8833 |
|  | 2-hop non-linear | **0.0075** | **0.0612** | **0.7704** | **0.8780** | **0.8961** |
| Qwen2.5-0.5B | 1-hop linear | 0.0084 | 0.0706 | 0.7462 | 0.8652 | 0.8657 |
|  | 1-hop non-linear | 0.0076 | 0.0644 | 0.7695 | 0.8788 | 0.8786 |
|  | 2-hop non-linear | **0.0074** | **0.0623** | **0.7764** | **0.8849** | **0.8922** |

## 3.4 Early Emergence

Having empirically validated that the relationship between neural topology and language generation performance is universal, sparse, and non-linear, there still remains an open question: Is this dependence an inherent consequence of the LLM's architectural design, or does it emerge during pretraining? While definitively answering this question is challenging, in this section we offer an exploratory analysis from the perspective of LLMs' learning process.

To inspect how the dependence between neural topology and perplexity evolves during pretraining, we perform graph probing on intermediate checkpoints of the Pythia models at various pretraining steps[3]. Figure 6 presents the probing performance throughout pretraining for Pythia-160M and Pythia-410M, alongside the corresponding average perplexity values across the dataset as an indicator of language generation capability. We find that, although it takes more than 143,000 pretraining steps for the models to reduce their average perplexity from over 60,000 to 21.02 (Pythia-160M) and 14.35 (Pythia-410M), the predictability of perplexity from neural topology emerges much earlier. Notably, graph probing detects meaningful predictability after only 8 pretraining steps, when the average perplexity remains as high as 43,000 and 29,000, respectively. This suggests that LLMs may first establish informative neural topological structures which are indirectly optimized through parameter updates, before developing strong next-token prediction capabilities built upon that topology. In addition, the early emergence of this topology–performance relationship opens up promising avenues for monitoring the learning trajectory of LLMs, detecting training failure at an early stage, designing effective early stopping strategies, and advancing theoretical understanding of LLMs' learning dynamics.

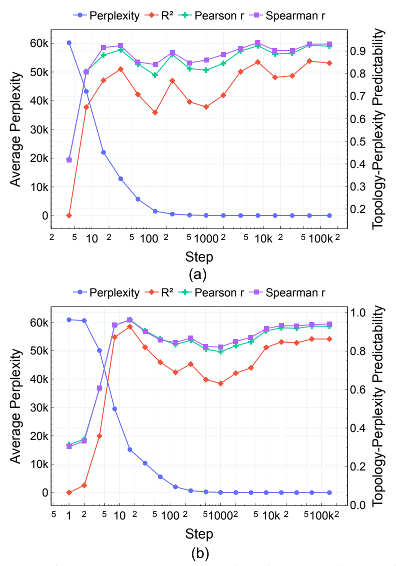

Figure 6: Average perplexity and probing performance throughout pretraining for (a) Pythia-160M (b) Pythia-410M.

# 4 Matching Neural Topology across LLMs

Despite significant discrepancies in architectures, parameters, and training data across different LLMs, they are all trained to optimize the same next-token prediction objective, whose performance, as we have shown, is closely tied to the internal neural topology. This raises a natural question: *do different LLMs develop similar neural topology patterns despite their differences*? To investigate potential structural similarity across LLMs, we extend graph probing with contrastive learning to perform *graph matching*, as illustrated in Figure 7. This extension encourages graph representations derived

[3]We do not include other LLMs in this analysis, as their intermediate checkpoints are not publicly available.

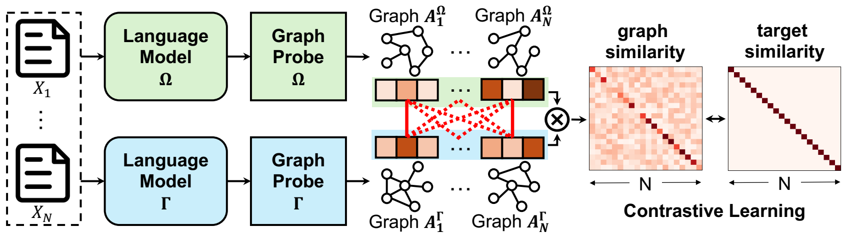

Figure 7: An overview of graph matching. We learn representations of neural topologies derived from two different LLMs processing the same text dataset. We then perform contrastive learning on the graph representations such that matching pairs are more similar by inner product.

from the same input text to be more similar than those from different texts. Specifically, suppose we feed a batch of $B$ token sequences into two LLMs, $\Omega$ and $\Gamma$. We compute the corresponding neural connectivity graphs and use two graph probes to encode them into representations $\mathbf{Z}_\Omega = [\mathbf{z}_1^\Omega, \dots, \mathbf{z}_B^\Omega]$ and $\mathbf{Z}_\Gamma = [\mathbf{z}_1^\Gamma, \dots, \mathbf{z}_B^\Gamma]$, as described in equations (5–6). Graph matching is implemented using a contrastive cross-entropy loss that encourages alignment between graph representations:

$$\mathcal{S} = \texttt{MAT\_MUL}(\mathbf{Z}_\Omega^T, \mathbf{Z}_\Gamma), \quad \mathcal{T} = \texttt{IDENTITY}(B), \tag{9}$$

$$\mathcal{L} = \sum_{i=1}^{B} \texttt{CROSS\_ENTROPY}(\mathcal{S}_{i,:}, \mathcal{T}_{i,:}) + \sum_{j=1}^{B} \texttt{CROSS\_ENTROPY}(\mathcal{S}_{:,j}, \mathcal{T}_{:,j}), \tag{10}$$

where $\mathcal{S}$ is the similarity matrix by taking inner product of graph representations and $\mathcal{T}$ is the target identity matrix for graph matching.

After training the graph probes contrastively on a shared set of training texts, the out-of-sample graph matching performance serves as an indicator of neural topology similarity between two LLMs. To evaluate this, we adopt the commonly used AUC and GAUC metrics [32], where AUC measures the global ranking quality across the entire test set, while GAUC computes a local ranking for each graph pair against all others, making AUC the more challenging metric (see Appendix F for details). Table 4 presents the graph matching results. As a sanity check, we first perform *self-matching* using the same LLM. Given that identical text inputs induce identical neural topologies, the results indeed show that both AUC and GAUC are close to 1.0, validating the rationality of our methodology. We then extend the matching experiments across multiple configurations, including: (1) LLMs within the same family but from different generations, (2) LLMs across different families, and (3) the same LLM trained with different random seeds. Surprisingly, all cross-model configurations yield high graph matching performance, with AUC (GAUC) scores ranging from 0.86 (0.87) to 0.95 (0.96). These results suggest that distinct LLMs develop strikingly similar neural topology patterns, implying the emergence of shared functional structures despite substantial differences in architectures, parameters, and training data. Moreover, we observe that cross-seed and cross-generation matching outperform cross-family matching, which is intuitive given the reduced architectural and training data differences within the same model family.

Table 4: Graph matching performance between different LLM configurations evaluated on AUC ($\times$100) and GAUC ($\times$100) at 80% connection sparsity levels (20% density).

| **Matching** | **LLM Ω** | **LLM Γ** | **AUC** | **GAUC** |
|---|---|---|---|---|
| **Self Matching** | GPT2 | GPT2 | 97.32 | 98.64 |
| | Pythia-160M | Pythia-160M | 95.95 | 96.92 |
| | Qwen2.5-0.5B | Qwen2.5-0.5B | 98.05 | 99.24 |
| **Cross Generation** | Qwen2.5-0.5B | Qwen2-0.5B | 91.45 | 93.27 |
| | Qwen2.5-0.5B | Qwen1.5-0.5B | 95.19 | 96.10 |
| | Qwen2-0.5B | Qwen1.5-0.5B | 92.92 | 94.21 |
| **Cross Family** | GPT2 | Pythia-160M | 90.96 | 92.00 |
| | GPT2 | Qwen2.5-0.5B | 90.30 | 91.11 |
| | Pythia-160M | Qwen2.5-0.5B | 86.17 | 87.39 |
| **Cross Seed** | Pythia-160M-seed1 | Pythia-160M-seed2 | 92.75 | 93.87 |
| | Pythia-160M-seed1 | Pythia-160M-seed3 | 92.48 | 93.59 |
| | Pythia-160M-seed2 | Pythia-160M-seed3 | 92.95 | 94.20 |

## 5 Related Work

**Probing LLMs.** Growing concerns over the transparency and steerability of LLMs have driven recent advances in reverse-engineering LLMs by extracting interpretable features from their neural activations through probes [45]. Supervised probing typically maps neuron activations to interpretable semantics through regression or classification [23, 24, 26, 27, 12, 30, 48, 5]. For example, Gurnee *et al.* [24] predicted the time and location of input entities from LLM activations. Unsupervised probing, by contrast, aims to learn a dictionary of disentangled features related to more abstract concepts [13, 18, 41, 33, 39, 13]. A famous example is the *Golden Gate Bridge* feature identified in the Claude 3 Sonnet model [48]. While prior work focused on connecting LLM activations to external semantics, our work studies the *functional topology* of neurons in LLMs, and relates this internal structure directly to language generation performance via *graph probing*.

**Network Neuroscience.** The study of functional networks in the human brain has been a central topic in neuroscience for decades [4, 3, 15, 37] which motivates this research. Brain networks are typically constructed by correlating fMRI or EEG signals across different neural regions, and then analyzed using tools from network science [2], which has revealed a range of structural and functional properties, such as small-worldness [4], economical wiring [9], and functional specialization [15]. More recently, several studies have drawn parallels between LLM activations and human brain activity [49, 11, 31, 42, 38, 51, 8, 46, 34]. For instance, Tuckute *et al.* [51] used GPT-2 activations to identify sentence stimuli that drive or suppress human brain responses. However, while these efforts focus on representational similarities, the functional *topology* of neurons within LLMs and its relationship to the model's language generation capabilities remain largely unexplored.

## 6 Discussion

Neurons in LLMs are connected both structurally through the model's architecture and functionally through their dynamic responses to input linguistic stimuli. In this work, we focus on the latter and demonstrate that the language generation performance of LLMs can be reliably predicted from their functional neural topologies using our proposed graph probing approach. Beyond the shared sparsity, non-linearity, and early emergence of this topology–performance relationship across models, we also find that different LLMs, despite substantial differences in architectures, parameters, and training data, exhibit highly similar topological patterns, as their neural topologies can be matched with near-perfect accuracy, suggesting a common underlying structure in how token sequences are processed. These findings imply that LLMs develop intricate and consistent topological structures among their neurons that are fundamental to their emergent ability to generate coherent language.

While we have empirically shown a strong dependence between next-token prediction perplexity and neural topology, we have not yet identified specific topological structures such as motifs, or physical metrics like small-worldness and modularity within these neural graphs. It remains an open question whether such properties exist in LLMs' neural topology and play a causal role in shaping their language generation capabilities. This valuable knowledge, now captured implicitly by the graph probes, may be further uncovered through careful analysis of the learned graph representations. Additionally, this paper evaluates LLMs with up to 14B parameters, while leaving graph probing on even larger models for future work due to the substantial computational costs of both LLM inference and connectivity graph construction at that scale.

Our graph probing results raise many interesting directions for future research. While we have linked neural topology to general next-token prediction ability, it remains unclear whether specific topologies emerge for specialized domains such as mathematical proofs or codes. We conjecture that certain neurons may become functionally specialized when processing text from particular domains, which might be identified via graph probing on datasets from diverse fields. Additionally, recent advances in enhancing LLMs' reasoning abilities [22] raise a natural question: does reasoning alter, or is it constrained by, neural topology? Moreover, the observed sparsity and early emergence of neural topology suggest potential applications in parameter pruning and early training failure detection, offering promising avenues for reducing the inference and training cost of LLMs. Finally, graph probing is model-agnostic and can be extended to models other than LLMs. In particular, applying graph probing to vision-language models may shed light on the neural topology underlying multi-modal generation capabilities. In all, we believe graph probing offers a promising lens for understanding AI models and ultimately guiding their improvement in an reliable and safe way.

## References

[1] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017.

[2] Albert-László Barabási. Network science. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1987):20120375, 2013.

[3] Danielle S Bassett and Olaf Sporns. Network neuroscience. *Nature neuroscience*, 20(3):353–364, 2017.

[4] Danielle Smith Bassett and ED Bullmore. Small-world brain networks. *The neuroscientist*, 12(6):512–523, 2006.

[5] Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

[6] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[7] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

[8] Laurent Bonnasse-Gahot and Christophe Pallier. fmri predictors based on language models of increasing complexity recover brain left lateralization. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[9] Ed Bullmore and Olaf Sporns. The economy of brain network organization. *Nature reviews neuroscience*, 13(5):336–349, 2012.

[10] Edward T Bullmore and Danielle S Bassett. Brain graphs: graphical models of the human brain connectome. *Annual review of clinical psychology*, 7(1):113–140, 2011.

[11] Charlotte Caucheteux, Alexandre Gramfort, and Jean-Rémi King. Evidence of a predictive coding hierarchy in the human brain listening to speech. *Nature human behaviour*, 7(3):430–441, 2023.

[12] Xiangjue Dong, Yibo Wang, Philip S Yu, and James Caverlee. Probing explicit and implicit gender bias through llm conditional text generation. *arXiv preprint arXiv:2311.00306*, 2023.

[13] Joshua Engels, Isaac Liao, Eric J Michaud, Wes Gurnee, and Max Tegmark. Not all language model features are linear. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, Apr 24-28, 2025*. OpenReview.net, 2025.

[14] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[15] Panagiotis Fotiadis, Linden Parkes, Kathryn A Davis, Theodore D Satterthwaite, Russell T Shinohara, and Dani S Bassett. Structure–function coupling in macroscale human brain networks. *Nature Reviews Neuroscience*, 25(10):688–704, 2024.

[16] Kunihiko Fukushima. Cognitron: A self-organizing multilayered neural network. *Biological cybernetics*, 20(3):121–136, 1975.

[17] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[18] Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, Apr 24-28, 2025*. OpenReview.net, 2025.

[19] Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024.

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

[21] Ariel Goldstein, Zaid Zada, Eliav Buchnik, Mariano Schain, Amy Price, Bobbi Aubrey, Samuel A Nastase, Amir Feder, Dotan Emanuel, Alon Cohen, et al. Shared computational principles for language processing in humans and deep language models. *Nature neuroscience*, 25(3):369–380, 2022.

[22] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[23] Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *Trans. Mach. Learn. Res.*, 2023, 2023.

[24] Wes Gurnee and Max Tegmark. Language models represent space and time. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[25] Robert Huben, Hoagy Cunningham, Logan Riggs, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[26] Charles Jin and Martin Rinard. Emergent representations of program semantics in language models trained on programs. In *Forty-first International Conference on Machine Learning*, 2024.

[27] Tianjie Ju, Weiwei Sun, Wei Du, Xinwei Yuan, Zhaochun Ren, and Gongshen Liu. How large language models encode context knowledge? a layer-wise probing study. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8235–8246, 2024.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[29] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[30] Connor Kissane, Robert Krzyzanowski, Joseph Isaac Bloom, Arthur Conmy, and Neel Nanda. Interpreting attention layer outputs with sparse autoencoders. *arXiv preprint arXiv:2406.17759*, 2024.

[31] Sreejan Kumar, Theodore R Sumers, Takateru Yamakoshi, Ariel Goldstein, Uri Hasson, Kenneth A Norman, Thomas L Griffiths, Robert D Hawkins, and Samuel A Nastase. Shared functional specialization in transformer-based language models and the human brain. *Nature communications*, 15(1):5523, 2024.

[32] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects. In *International conference on machine learning*, pages 3835–3845. PMLR, 2019.

[33] Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300, 2024.

[34] Yiheng Liu, Xiaohui Gao, Haiyang Sun, Bao Ge, Tianming Liu, Junwei Han, and Xintao Hu. Brain-inspired exploration of functional networks and key neurons in large language models. *arXiv preprint arXiv:2502.20408*, 2025.

[35] Zixiang Luo, Kaining Peng, Zhichao Liang, Shengyuan Cai, Chenyu Xu, Dan Li, Yu Hu, Changsong Zhou, and Quanying Liu. Mapping effective connectivity by virtually perturbing a surrogate brain. *arXiv preprint arXiv:2301.00148*, 2022.

[36] Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, Apr 24-28, 2025*. OpenReview.net, 2025.

[37] John D Medaglia, Mary-Ellen Lynall, and Danielle S Bassett. Cognitive network neuroscience. *Journal of cognitive neuroscience*, 27(8):1471–1491, 2015.

[38] Gavin Mischler, Yinghao Aaron Li, Stephan Bickel, Ashesh D Mehta, and Nima Mesgarani. Contextual feature extraction hierarchies converge in large language models and the brain. *Nature Machine Intelligence*, pages 1–11, 2024.

[39] Anish Mudide, Joshua Engels, Eric J Michaud, Max Tegmark, and Christian Schroeder de Witt. Efficient dictionary learning with switch sparse autoencoders. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

[40] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[41] Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders. *arXiv preprint arXiv:2404.16014*, 2024.

[42] Neil Rathi, Johannes Mehrer, Badr AlKhamissi, Taha Osama A Binhuraib, Nicholas Blauch, and Martin Schrimpf. Topolm: brain-like spatio-functional organization in a topographic language model. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, Apr 24-28, 2025*. OpenReview.net, 2025.

[43] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[44] Martin Schrimpf, Idan Asher Blank, Greta Tuckute, Carina Kauf, Eghbal A Hosseini, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. The neural architecture of language: Integrative modeling converges on predictive processing. *Proceedings of the National Academy of Sciences*, 118(45):e2105646118, 2021.

[45] Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, et al. Open problems in mechanistic interpretability. *arXiv preprint arXiv:2501.16496*, 2025.

[46] Haiyang Sun, Lin Zhao, Zihao Wu, Xiaohui Gao, Yutao Hu, Mengfei Zuo, Wei Zhang, Junwei Han, Tianming Liu, and Xintao Hu. Brain-like functional organization within large language models. *arXiv preprint arXiv:2410.19542*, 2024.

[47] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[48] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024.

[49] Mariya Toneva and Leila Wehbe. Interpreting and improving natural-language processing (in machines) with natural language-processing (in the brain). *Advances in neural information processing systems*, 32, 2019.

[50] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[51] Greta Tuckute, Aalok Sathe, Shashank Srikant, Maya Taliaferro, Mingye Wang, Martin Schrimpf, Kendrick Kay, and Evelina Fedorenko. Driving and suppressing the human language network using large language models. *Nature Human Behaviour*, 8(3):544–561, 2024.

[52] Petra E Vértes, Aaron F Alexander-Bloch, Nitin Gogtay, Jay N Giedd, Judith L Rapoport, and Edward T Bullmore. Simple models of human brain functional networks. *Proceedings of the National Academy of Sciences*, 109(15):5868–5873, 2012.

[53] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022, 2022.

[54] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

## A Graph Probing Configuration

**Hyperparameters.** We train graph probes using the Adam optimizer [28] with mean squared error (MSE) loss, as defined in Equation (8). The learning rate is set to 0.001, with a batch size of 16. We apply a learning rate decay strategy, reducing the rate by a factor of 0.1 if the loss does not improve for 5 consecutive epochs. Each model is trained for up to 100 epochs, with early stopping triggered if no improvement is observed for 20 epochs. Dropout is not used, as preliminary experiments showed no significant impact on regression performance.

**Computational Resources.** LLM inference for computing neural topologies and perplexity scores requires GPUs with large memory. All experiments were conducted on a Linux server equipped with 8 NVIDIA A100 GPUs (80GB memory each). In contrast, training graph probes is relatively lightweight and can be performed on a single GPU with 16GB memory in less than 1 hour.

## B Experimented LLMs

We run graph probing experiments on a diverse range of LLMs across three different families, with the numper of parameters ranging from 124M to 14B. Basic information of these experimented LLMs is summarized in Table 5.

Table 5: Basic information of the experimented LLMs.

| LLM family | #params | #layers | #neurons per layer | experimented layer id |
|---|---|---|---|---|
| GPT-2 | 124M | 12 | 768 | 1-12 |
| | 355M | 24 | 1024 | 12 |
| | 774M | 36 | 1280 | 18 |
| Pythia | 160M | 12 | 768 | 1-12 |
| | 410M | 24 | 1024 | 12 |
| | 1.4B | 24 | 2048 | 12 |
| | 2.8B | 32 | 2560 | 16 |
| | 6.9B | 32 | 4096 | 16 |
| | 12B | 36 | 5120 | 18 |
| Qwen2.5 | 0.5B | 24 | 896 | 12 |
| | 3B | 36 | 2048 | 18 |
| | 7B | 28 | 3584 | 14 |
| | 14B | 48 | 5120 | 24 |

## C Datasets

We conduct graph probing experiments using the same text datasets on which the LLMs were originally pretrained. Specifically, we adopt the Pile dataset [17] for models in the Pythia family, and the OpenWebText dataset [20] for GPT-2 and Qwen2.5 models. For each dataset, we randomly sample 10,000 text sequences to construct neural connectivity graphs. Each sample is generated by merging and tokenizing raw text until it reaches a length between 256 and 1024 tokens, which defines the length of the corresponding neural activity time series used for computing pairwise correlations. We then construct a text-responsive neural connectivity graph for each sample and compute its associated perplexity score. To remove outliers that distort the distribution, we filter out the top 1% and bottom 1% of samples based on perplexity. Finally, we normalize all perplexity values to the range $[0, 1]$ by subtracting the minimum perplexity and dividing by the observed range. Summary statistics for the constructed datasets are provided in Table 6.

Table 6: Basic information of constructed graph probing datasets.

| LLM family | Dataset | #tokens | #graphs | #training graphs | #test graphs |
|---|---|---|---|---|---|
| GPT-2 | OpenWebText | 7,020,215 | 10,384 | 8,308 | 2,076 |
| Pythia | Pile | 5,216,371 | 8,011 | 6,409 | 1,602 |
| Qwen2.5 | OpenWebText | 7,935,555 | 11,452 | 9162 | 2,290 |

## D Perplexity Regression Results

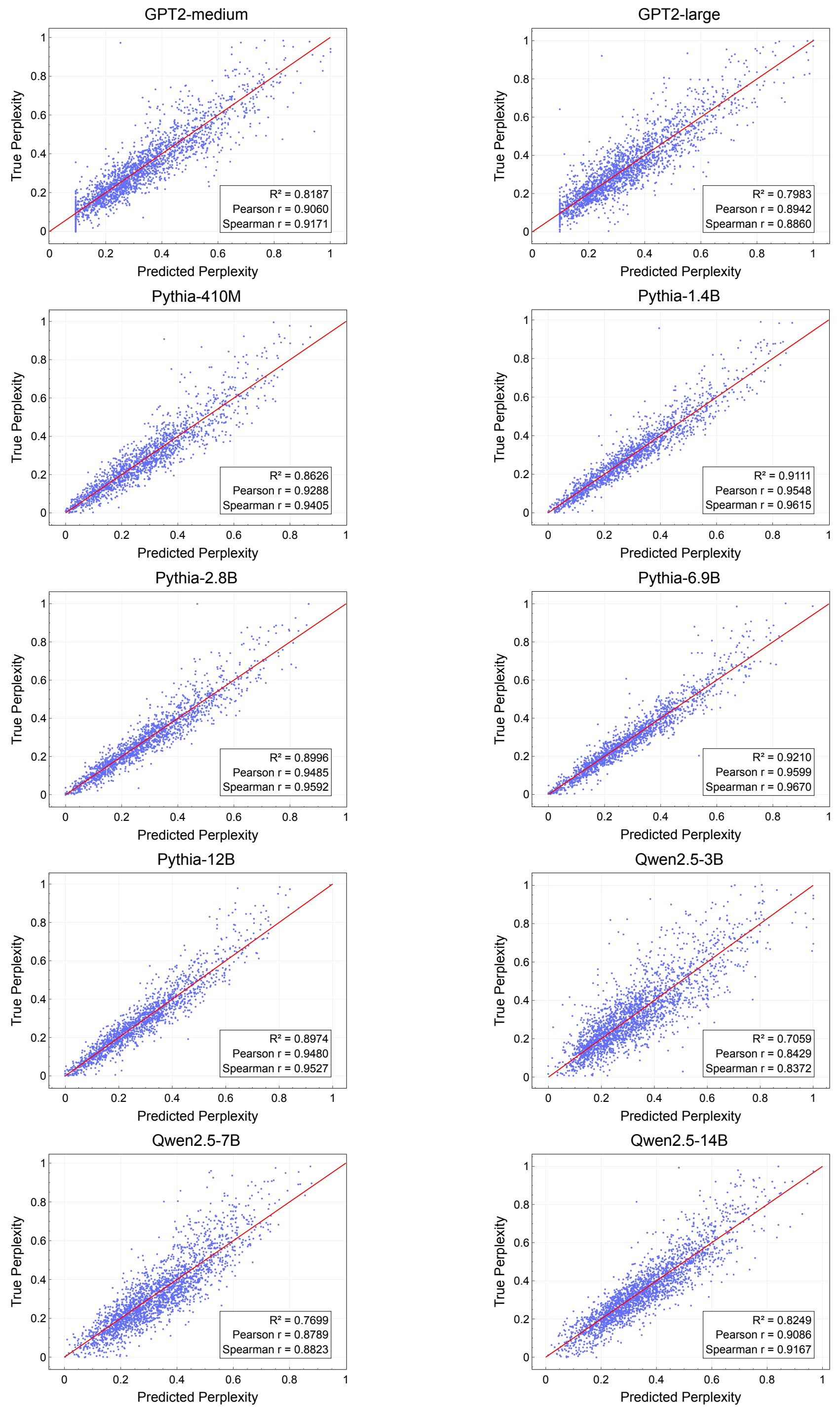

Figure 8: Out-of-sample performance of graph probing on different LLMs, including GPT2-medium, GPT2-large, Pythia-410M, Pythia-1.4B, Pythia-2.8B, Pythia-6.9B, Pythia-12B, Qwen2.5-3B, Qwen2.5-7B, and Qwen2.5-13B.

## E Results of Different Probe Dimensions

In addition to the number of graph convolutional layers ($L$), the hidden dimensionality ($d$) is a key factor influencing the expressive power of graph probes. We report out-of-sample graph probing performance across different values of $d$ in Tables 7–9. As expected, we observe a monotonic improvement in perplexity regression performance with increasing $d$, indicating that higher-dimensional probes are more capable of capturing fine-grained topological patterns in neural connectivity graphs. Notably, all experiments in the main paper were conducted with $d = 32$, which already yielded strong predictive accuracy despite its relatively small size.

Table 7: Out-of-sample perplexity regression performance for GPT-2 using different dimensions $d$ on sparse neural topologies where top 10% functional connections are reserved (90% sparsity).

| $d$ | **MSE** $\downarrow$ | **MAE** $\downarrow$ | $\mathbf{R^2}$ $\uparrow$ | $\rho_\mathbf{P}$ $\uparrow$ | $\rho_\mathbf{S}$ $\uparrow$ |
|---|---|---|---|---|---|
| 4 | 0.0098 | 0.0738 | 0.7041 | 0.8400 | 0.8525 |
| 8 | 0.0089 | 0.0707 | 0.7321 | 0.8566 | 0.8621 |
| 16 | 0.0085 | 0.0692 | 0.7431 | 0.8631 | 0.8726 |
| 32 | 0.0085 | 0.0690 | 0.7419 | 0.8627 | 0.8666 |
| 64 | 0.0062 | 0.0584 | 0.8112 | 0.9008 | 0.9071 |
| 128 | 0.0058 | 0.0562 | 0.8249 | 0.9090 | 0.9149 |
| 256 | 0.0050 | 0.0524 | 0.8496 | 0.9224 | 0.9274 |

Table 8: Out-of-sample perplexity regression performance for Pythia-160M using different dimensions $d$ on sparse neural topologies where top 10% functional connections are reserved (90% sparsity).

| $d$ | **MSE** $\downarrow$ | **MAE** $\downarrow$ | $\mathbf{R^2}$ $\uparrow$ | $\rho_\mathbf{P}$ $\uparrow$ | $\rho_\mathbf{S}$ $\uparrow$ |
|---|---|---|---|---|---|
| 4 | 0.0095 | 0.0705 | 0.7087 | 0.8422 | 0.8762 |
| 8 | 0.0089 | 0.0688 | 0.7276 | 0.8534 | 0.8761 |
| 16 | 0.0084 | 0.0660 | 0.7417 | 0.8622 | 0.8783 |
| 32 | 0.0074 | 0.0627 | 0.7736 | 0.8796 | 0.9004 |
| 64 | 0.0076 | 0.0630 | 0.7687 | 0.8779 | 0.8970 |
| 128 | 0.0058 | 0.0542 | 0.8227 | 0.9073 | 0.9264 |
| 256 | 0.0051 | 0.0515 | 0.8436 | 0.9185 | 0.9336 |

Table 9: Out-of-sample perplexity regression performance for Qwen2.5-0.5B using different dimensions $d$ on sparse neural topologies where top 10% functional connections are reserved (90% sparsity).

| $d$ | **MSE** $\downarrow$ | **MAE** $\downarrow$ | $\mathbf{R^2}$ $\uparrow$ | $\rho_\mathbf{P}$ $\uparrow$ | $\rho_\mathbf{S}$ $\uparrow$ |
|---|---|---|---|---|---|
| 4 | 0.0090 | 0.0725 | 0.7265 | 0.8525 | 0.8539 |
| 8 | 0.0084 | 0.0689 | 0.7433 | 0.8634 | 0.8686 |
| 16 | 0.0079 | 0.0659 | 0.7592 | 0.8733 | 0.8757 |
| 32 | 0.0078 | 0.0655 | 0.7633 | 0.8781 | 0.8839 |
| 64 | 0.0049 | 0.0517 | 0.8505 | 0.9224 | 0.9257 |
| 128 | 0.0047 | 0.0510 | 0.8569 | 0.9259 | 0.9311 |
| 256 | 0.0042 | 0.0471 | 0.8714 | 0.9336 | 0.9408 |

## F Graph Matching Metrics

Given the predicted similarity matrix $\mathcal{S} \in \mathbb{R}^{N \times N}$ and the target similarity matrix $\mathcal{T} = \texttt{IDENTITY}(N)$, we calculate the following metrics for graph matching [32]:

$$\texttt{AUC} = \texttt{AREA\_UNDER\_ROC}(\texttt{flatten}(\mathcal{S}), \texttt{flatten}(\mathcal{T})), \tag{11}$$

$$\texttt{GAUC} = \frac{1}{2N} \sum_{i=1}^{N} \left(\texttt{AREA\_UNDER\_ROC}(\mathcal{S}_{i,:}, \mathcal{T}_{i,:}) + \texttt{AREA\_UNDER\_ROC}(\mathcal{S}_{:,i}, \mathcal{T}_{:,i})\right). \tag{12}$$

## 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: See Abstract and Section 1.

   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: See Section 6.

   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: The paper does not include theoretical results.

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: We release all the code and data, as well as instructions for how to replicate the results. See Section 3.

   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: We have submitted code and data anonymously as supplementary materials.

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, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide sufficient information on experimental setting. See Section 3.

   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: We report the the statistical significance of the experiments suitably and correctly. See Section 3.

   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 provide sufficient information on the computer resources. See Section 3.

   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 make sure that the presented research conforms with the NeurIPS Code of Ethics.

   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: [Yes]

    Justification: We provide thorough discussion about broader impacts of this work. See Section 6.

    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: The paper poses no such 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: All assets used in the paper are properly credited. The license and terms of use are explicitly mentioned and properly respected.

    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: All new assets introduced in the paper are well documented and we provide the documentation alongside the assets.

    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: The 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: The 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: [Yes]

    Justification: We provide sufficient information on the usage of LLMs. See Section 2 and 3.

    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.