Transformers Represent Belief State Geometry in their Residual Stream

Adam S. Shai
Simplex
PIBBSS*

Sarah E. Marzen
Department of Natural Sciences
Pitzer and Scripps College

Lucas Teixeira
PIBBSS

Alexander Gietelink Oldenziel
University College London
Timaeus

Paul M. Riechers
Simplex
BITS

Abstract

What computational structure are we building into large language models when we train them on next-token prediction? Here, we present evidence that this structure is given by the meta-dynamics of belief updating over hidden states of the data-generating process. Leveraging the theory of optimal prediction, we anticipate and then find that belief states are linearly represented in the residual stream of transformers, even in cases where the predicted belief state geometry has highly nontrivial fractal structure. We investigate cases where the belief state geometry is represented in the final residual stream or distributed across the residual streams of multiple layers, providing a framework to explain these observations. Furthermore we demonstrate that the inferred belief states contain information about the entire future, beyond the local next-token prediction that the transformers are explicitly trained on. Our work provides a general framework connecting the structure of training data to the geometric structure of activations inside transformers.

1 Introduction

In this work, we present a rigorous and concrete theoretical framework that connects the structure of training data to the geometry of activations in trained transformer neural networks. Our framework is grounded in the theory of optimal prediction. It suggests that transformers pretrained on next-token prediction will develop internal structures characterized by the meta-dynamics of belief updating over hidden states of the data-generating process. In other words, pretrained models should learn *more* than the hidden structure of the data generating process—they must also learn how to update their beliefs about the hidden state of the world as they synchronize to it in context.

To test this framework, we conduct well-controlled experiments where we train transformers on data generated from processes with hidden ground truth structure, and then use our theory to make predictions about the geometry of internal activations. Even in cases where the framework predicts highly nontrivial fractal structure, our empirical results confirm these predictions (Figure 1). This provides a comprehensive explanation of how transformers encode information beyond the local next-token predictions they are explicitly trained on.

In Section 2, we review the relevant theory from computational mechanics which motivates our prediction of the geometry of activations in the residual stream. Core to our work is the *mixed-state presentation*, which describes the metadynamic of beliefs in the probability simplex over states of the

*Principles of Intelligent Behaviour in Biological and Social Systems

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

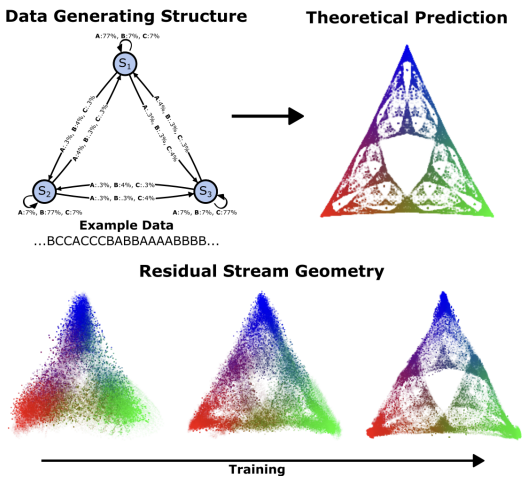

Figure 1: (Top) Given a hidden data-generating structure, our framework predicts a unique belief state geometry in a probability simplex. Often these have highly nontrivial fractal structure as shown in this example. (Bottom) Our main experimental result is that we find that the fractal geometry of optimal beliefs is linearly embedded in the residual stream, and emerges over the course of training.

data generating process. This formalism leads to a geometric structure that forms a natural hypothesis for the internal states of neural networks trained on prediction tasks.

In Section 3, we verify that this geometry is linearly represented in the residual stream of transformers. We implement two main experiments that control for different aspects of the hidden structure of the data generating process and make concrete predictions about the internal states of networks trained on these datasets. In some cases the geometry of belief state updating is found in the final residual stream, while in others it is spread out across multiple layers. We detail how our framework explains this phenomenon. After demonstrating the initial success of this framework, Section 4 discusses the theory and implications in more depth.

2 Theory and methods

2.1 Data generating processes

In this study, we assume that our training data is generated by an edge-emitting hidden Markov model (HMM)[2]. Paths through the HMM produce sequences of tokens from a predefined vocabulary. Tokens ($x \in \mathcal{X}$) are emitted as we transition between hidden states ($s \in \mathcal{S}$). These transitions are governed by token-labeled transition matrices $\{T^{(x)}\}$, where each $T^{(x)}_{i,j} = \Pr(x, s_j | s_i)$ represents the joint probability of emitting token x and transitioning to state s_j, given that the HMM was in state s_i.

As a simple example, consider the HMM shown in Figure 2, that we call the Zero-One-Random (Z1R) process. The Z1R process generates strings of tokens of the form `...01R01R...`, where `R` is a randomly generated `0` or `1`. This HMM has 3 states: $\mathtt{S}_0$, $\mathtt{S}_1$, and $\mathtt{S}_\mathtt{R}$. Arrows of the form $s \xrightarrow{x:p\%} s'$ denote the probability $\Pr(x, s' | s) = p\%$ of moving to state s' and emitting the token x, given that the process was in state s.

2.2 Mixed-state presentations

There are competing intuitions for what transformers should represent. Are they stochastic parrots [2]? Do they build a world model [13]? One natural intuition would be that transformers represent the hidden structure of the data-generating process—a world model—in order to predict the next token well. For instance, Ilya Sutskever has said: "Predicting the next token well means that you understand

[2]For arbitrarily large, possibly non-ergodic HMMs with arbitrary initial distribution over the latent states, this does not limit the sophistication of training data.

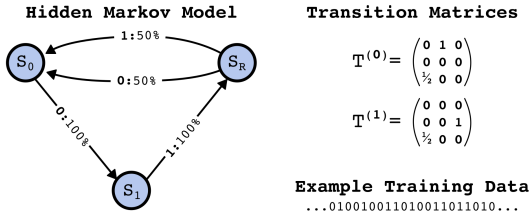

Figure 2: An illustration of a hidden Markov model (HMM) and its components. The left side shows the HMM with states $\mathtt{S}_0$, $\mathtt{S}_1$, and $\mathtt{S}_\mathtt{R}$, and their respective transition probabilities. The right side displays the transition matrices $T^{(\mathtt{0})}$ and $T^{(\mathtt{1})}$ corresponding to token emissions `0` and `1`. Example training data is provided at the bottom, demonstrating a sequence generated by the HMM.

the underlying reality that led to the creation of that token [32]." This type of intuition is natural, but not formal.

Computational mechanics is a formalism that was developed in order to study the limits of prediction in chaotic systems, and has since expanded to a deep and rigorous theory of computational structure for any process [4]. One of its contributions is in providing a rigorous answer to what structures are necessary to perform optimal prediction [31].

Interestingly, computational mechanics shows that prediction is substantially more complicated than generation [19, 30]. Informally, the reason for this is that even if an agent knows the underlying hidden generative structure that creates the data they are trying to predict, the agent must still perform computational work in order to infer which state the generative process is in, given finite observations of data. Thus, there are conceptually two distinct inference processes related to prediction: learning the hidden structure of the data generating process, and subsequently inferring which state the data generating process is in given finite observations of data.

Computational mechanics formally captures the computational structure of the latter inference process with the *mixed-state presentation* (MSP). Whereas the HMM of a data generating process describes a generative structure (Figure 3A), the MSP is the structure of prediction (Figure 3B). The MSP answers the question of how an optimal observer updates their beliefs over the states of the data-generating process given finite observations of tokens [27, 15]. If the observer is in a belief state given by a probability distribution $\boldsymbol{\eta}$ (a row vector) over the hidden states of the data generating process, then the update rule for the new belief state $\boldsymbol{\eta}'$ given that the observer sees a new token x is:

$$\boldsymbol{\eta}' = \frac{\boldsymbol{\eta} T^{(x)}}{\boldsymbol{\eta} T^{(x)} \mathbf{1}} \tag{1}$$

where $\mathbf{1}$ is a column vector of ones of appropriate dimension, with the denominator ensuring proper normalization of the updated belief state. In general, starting from the initial belief state $\boldsymbol{\eta}_\varnothing$, we can find the belief state after observing a sequence of tokens $x_0, x_1, \ldots, x_N$:

$$\boldsymbol{\eta} = \frac{\boldsymbol{\eta}_\varnothing T^{(x_0)} T^{(x_1)} \cdots T^{(x_N)}}{\boldsymbol{\eta}_\varnothing T^{(x_0)} T^{(x_1)} \cdots T^{(x_N)} \mathbf{1}} \ . \tag{2}$$

This process of belief updating is itself another HMM, where hidden states are associated with belief states, and paths leading to particular belief states are the observed token sequence. The probability of transitioning from belief state $\boldsymbol{\eta}$ to $\boldsymbol{\eta}' = \boldsymbol{\eta} T^{(x)} / \boldsymbol{\eta} T^{(x)} \mathbf{1}$ of Eq. (1) is simply given by the probability $\boldsymbol{\eta} T^{(x)} \mathbf{1}$ of observing an x from $\boldsymbol{\eta}$. For stationary processes, the optimal initial belief state is given by the stationary distribution $\boldsymbol{\eta}_\varnothing = \boldsymbol{\pi}$ over latent states (the left-eigenvector of the transition matrix $T = \sum_x T^{(x)}$ associated with the eigenvalue of 1).

Each belief state of the MSP is a probability distribution over the states of the data generating process. Belief states thus inherit a natural geometry; we can plot these belief states in a probability simplex, as indicated in Figure 3C. We refer to the geometric arrangement of the points on the simplex as the *belief state geometry*. Note that each belief state—as a distribution over generator states—induces a probability density over all possible futures. Distributions over the entire future clearly carry more than just next-token information. We can see this in our example in Figure 3D. The states $\boldsymbol{\pi}$, $\boldsymbol{\eta}_{\mathtt{10}}$, and $\boldsymbol{\eta}_{\mathtt{01}}$ all have the same next-token prediction despite being distinct belief states.