CAUSAL EVIDENCE OF STACK REPRESENTATIONS IN MODELING COUNTER LANGUAGES USING TRANSFORMERS

Nishit Singh
Birla Institute of Technology and Science, Pilani
f20221317@pilani.bits-pilani.ac.in

ABSTRACT

Formal languages have proven to be effective conduits to understand the inner mechanisms of transformers. Past work has shown that transformers trained on next-token prediction over counter languages learn representations consistent with an underlying stack structure. Beyond representational analysis, this paper investigates the causal role of these representations. Linear probes are trained to predict the stack depth at each token from the model's hidden states, and a principal representation direction is extracted from the probe. Ablation of this direction from the model causes sequential accuracy to collapse to near 0%, providing strong empirical evidence that the stack representation is not just learned, but is causally necessary for model performance.

1 INTRODUCTION

Mechanistic interpretability aims to understand the blackbox-like behavior of language models (LMs) by, among other things, identifying features (Elhage et al., 2022). A feature is a human-interpretable property of the model's activation on specific inputs. We use probing classifiers to link these activations to properties. Formally, let $f : x \to \hat{y}$ be a language model trained on auto-regressive next-token prediction of some language. A probing classifier $p : f_l(x) \to \hat{z}$ returns a property z from the embeddings of the LM at some layer l. The probing classifier is trained and evaluated on a dataset $D_P = \{f_l(x), z^{(i)}\}$ which constitutes the embedding and the property of the sequence at that layer (Belinkov, 2022; Hewitt & Liang, 2019). While probing classifiers are capable of extracting the latent representations of the language model, *they do not make any statements about causality*. The computational primitives developed during training may or may not be utilized in the computation undertaken by these models.

Formal languages prove to be effective testbeds for performing experiments in interpretability, owing to their mathematically rigorous formulation (Bhattamishra et al., 2020; Hahn, 2020). All algorithmic tasks can be reduced to a language within a specific class of formal languages (Pérez et al., 2019), making this line of inquiry worthwhile. Tiwari et al. (2025) empirically show the emergence of stack representations in language models trained on formal languages like Dyck-1 and Shuffle-k. They explicitly do not make any comments on the causality of these stack representations.

This paper builds on previous research by investigating the causal role of stack representations. A probe is trained to identify stack depth and the corresponding component from the model's hidden states is ablated during inference. This intervention leads to a near-complete collapse in sequence prediction accuracy, suggesting that the learned stack representation is causally necessary for the model's internal computation.

2 SETUP

2.1 COUNTER LANGUAGE MODELING

We carry out our experiments over the languages Dyck-1 and Shuffle-k (Bhattamishra et al., 2020) for $k = 2, 4, 6, 8$. Following the formalism by Tiwari et al. (2025), Dyck-1 can be defined as a

language over alphabet $\Sigma = \{'(', ')'\}$ with production rules:

$$S \to \begin{cases} \epsilon \\ (S) \\ SS \end{cases}$$

Shuffle is a binary operation over two strings which interleaves two strings in all possible ways. Formally, for strings $u, v \in \Sigma^*$, the shuffle $u \odot v$ is defined inductively as:

$$u \odot \epsilon = \epsilon \odot u = \{u\}$$

$$au \odot bv = \{aw : w \in u \odot bv\} \cup \{bw : w \in au \odot v\}$$

for $a, b \in \Sigma$. For languages $L_1, L_2 \subseteq \Sigma^*$, define:

$$L_1 \odot L_2 = \bigcup_{u \in L_1, v \in L_2} u \odot v.$$

Let $\{\Sigma_i\}_{i=1}^k$ be pairwise disjoint alphabets, and let $L_i \subseteq \Sigma_i^*$ be Dyck-1 languages over each alphabet. Then, the Shuffle-k language is defined as:

$$\text{Shuffle}_k = L_1 \odot L_2 \odot \cdots \odot L_k.$$

Intuitively, Shuffle_k consists of all strings formed by interleaving k independent Dyck-1 strings while preserving the order of symbols within each string.

2.2 Transformer Architecture

An encoder-only transformer is trained with a causal attention mask for next-token prediction over Shuffle-k strings. The vocabulary consists of k bracket pairs, one padding token and one `BOS` token. Given a prefix, the model predicts the set of valid continuations as a k-hot vector. The model uses embedding dimension $d_{embed} = 32$, hidden dimension $d_{model} = 64$, a single layer with 4 attention heads, feedforward dimension $d_{ffn} = 64$, and learned positional encodings. The decoder is a linear layer with sigmoid activation, and training uses MSE loss with RMSProp (lr 5×10^{-3}, batch size 32, over 10,000 sequences (lengths 2–50, 80/20 training/validation split).

Positional accuracy A_{pos} is defined as the fraction of valid token positions where the model's predicted set is equal to the ground truth set.

$$\text{A}_{\text{pos}} = \frac{1}{|\mathcal{M}|} \sum_{(b,t) \in \mathcal{M}} \mathbf{1}\left[\hat{y}_{b,t} = y_{b,t}\right]$$

Similarly, sequential accuracy of a string is defined as the fraction of strings which are valid in their entirety. One wrong token deems the whole string invalid, and thus this is naturally a more sensitive metric.

$$\text{A}_{\text{seq}} = \frac{1}{B} \sum_{b=1}^{B} \mathbf{1}\left[\forall\, t \in \mathcal{M}_b : \hat{y}_{b,t} = y_{b,t}\right]$$

$$\text{where we define } \hat{y}_{b,t} = \mathbf{1}\left[\sigma(\text{decoder}(h_{b,t})) > 0.5\right]$$

This language model achieves perfect validation accuracy in Shuffle-k for $k = 2, 3, .., 8$ by the 25th epoch, confirming that all representations analyzed in this work belong to a fully converged model.

2.3 Probe Design

Shuffle-k can be represented as k different stacks, each pertaining to a singular bracket type. An open bracket of type $'i'$ ($open_i$) increases the depth of the ith stack ($depth_i$)by 1, and a closing bracket $close_i$ decreases the stack depth by 1. The validity condition of the string is to have $depth_i = 0\ \forall i$.

Now, for a string $x = (x_0, x_1, ..., x_{n-1})$ the transformer produces, for a layer $'l'$ at token position $'t'$ a hidden state $h_t^{(l)}$ corresponding to $c_i(t)$, the depth of the ith stack at token position $'t'$. We now have:

$$D_P(l) = \{h_t^{(l)}, c_i(t)\}$$

Table 1: Some Selectivities of Linear Probes for Shuffle-K

Value of k	Selectivity
$k = 1$, Stack 1	48.8%
$k = 2$, Stack 1	68.7%
$k = 4$, Stack 1	83.6%
$k = 6$, Stack 2	85.8%

which is used as the training dataset for the probe. We construct a *linear classifier probe* (Belinkov, 2022) :

$$f(h) = Wh + b, W \in \mathbb{R}^{C \times d_{model}}, b \in \mathbb{R}^{C}$$

where $C =$ number of depth classes. Trained with cross entropy loss:

$$\mathcal{L} = -\sum_{(h,c)\in\mathcal{D}_{\text{train}}} \log \frac{e^{f(h)_c}}{\sum_{j=0}^{C-1} e^{f(h)_j}}$$

using Adam with learning rate 10^{-3}, batch size 32, for 10 epochs, with Xavier uniform weight initialization. A *control probe*, as described by Hewitt & Liang (2019), is trained on the dataset $D_{ctrl} = \{h_t^{(l)}, \pi(c_i(t))\}$ [1] and validated on the original validation set. Selectivity is then defined as the difference in the accuracy of the experimental probe and the control probe. High selectivity is an indicator of the stack depth genuinely being encoded in the hidden space vectors. As reported by Tiwari et al. (2025), selectivities for linear probes for Shuffle-k are presented in Table 1.

2.4 Intervention

The intervention follows the activation patching paradigm of Nanda et al. (2023) and Meng et al. (2023), where targeted manipulation of specific representation directions provides evidence of causal necessity. A linear classifier probe lets us extract the trained probe weight matrix $W \in \mathbb{R}^{C \times d_{model}}$. Performing singular value decomposition

$$W = U\Sigma V^{\dagger}$$

$$\text{where } U \in \mathbb{R}^{C\times C}, \Sigma \in \mathbb{R}^{C\times d_{model}}, \text{ and } V \in \mathbb{R}^{d_{model}\times d_{model}}$$

Defining $\mathbf{w} = V_{[0]} \in R^{d_{model}}, ||\mathbf{w}|| = 1$ as the top right singular vector of W, we move forward with ablating the direction of $\mathbf{w}$. For each hidden state $h_t^{(l)}$ at token t in layer l, living in $\mathbb{R}^{d_{model}}$, we perform the following operation:

$$h' = h - \alpha(\mathbf{w} \cdot h)\mathbf{w}, \alpha \in [0, 1]$$

We also ablate random directions $\mathbf{w} \sim Uniform(\mathcal{S}^{d_{model}-1})$. The ablated states are only passed through the decoder head, the transformer's internal computations are not re-run.

3 Results

The experiment was run[2] on Shuffle-k for $k = 6$. Figures 1 and 2 show the ablation of $\mathbf{w}$ from each stack and sweeping $\alpha = \{0, 0.25, 0.50, 0.75, 1\}$ causes the positional accuracy to dip, and sequential accuracy to collapse. The heightened sensitivity of sequential accuracy is expected, since modest per-position accuracy drops compound to severe drops in sequential accuracy. Furthermore, ablating random directions has no effect on the accuracies of the model. Building upon the discovery of emergent stack representations, these results provide strong empirical evidence that the stack representations are causally involved in the decoder's next token predictions. The sharp collapse points to the absence of alternative encodings partially compensating for the ablation of the stack direction.

[1] where $\pi(x)$ denotes a random perturbation of the training labels.

[2] Results for $k = 2, 4, 6, 8$ are provided in the Appendix A.

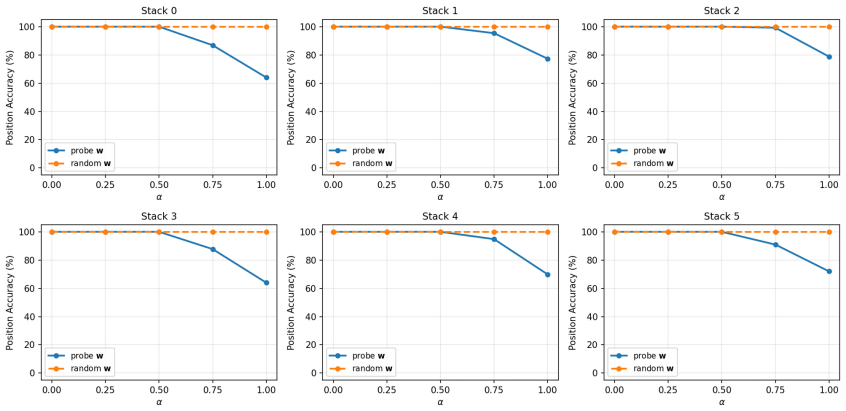

Figure 1: Position accuracy as a function of ablation strength for α for Shuffle-6. Accuracy drops from targeted ablation, random ablation keeps accuracy intact.

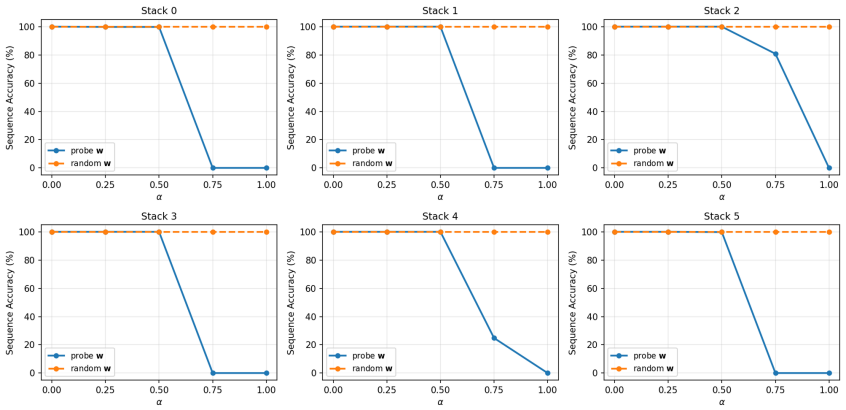

Figure 2: Sequence accuracy as a function of ablation strength α for Shuffle-6. Accuracy collapses from targeted ablation, random ablation keeps accuracy intact.

4 Future Work

A natural extension is to intervene *mid-network*. Importantly, the ablation is applied only at inference time to the final hidden states, without re-running the transformer. This isolates the causal contribution of the representation to the decoder's predictions. How the transformer tackles the manipulation of emergent computational primitives like stacks and counters while modeling formal languages needs to be explored by passing ablated or edited states through the feed-forward network and observing the change in outputs. Furthermore, the role of different architectures in the emergence of representations, their redundancy, causality, and their relationships remains an interesting direction.

References

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, March 2022. doi: 10.1162/coli_a_00422. URL https://aclanthology.org/2022.cl-1.7/.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the Ability and Limitations of Transformers to Recognize Formal Languages. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Process-*

ing (EMNLP), pp. 7096–7116, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.576. URL https://aclanthology.org/2020.emnlp-main.576/.

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

Michael Hahn. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171, 2020. doi: 10.1162/tacl_a_00306. URL https://aclanthology.org/2020.tacl-1.11/.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2733–2743, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1275. URL https://aclanthology.org/D19-1275/.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt, 2023. URL https://arxiv.org/abs/2202.05262.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability, 2023. URL https://arxiv.org/abs/2301.05217.

Jorge Pérez, Javier Marinkovic, and Pablo Barceló. On the turing completeness of modern neural network architectures. *CoRR*, abs/1901.03429, 2019. URL http://arxiv.org/abs/1901.03429.

Utkarsh Tiwari, Aviral Gupta, and Michael Hahn. Emergent stack representations in modeling counter languages using transformers. In *ICLR 2025 Workshop on World Models: Understanding, Modelling and Scaling*, 2025. URL https://openreview.net/forum?id=o5xTyDilNs.

A APPENDIX

Experimental results for Shuffle-k for different values of k.

A.1 SHUFFLE-2

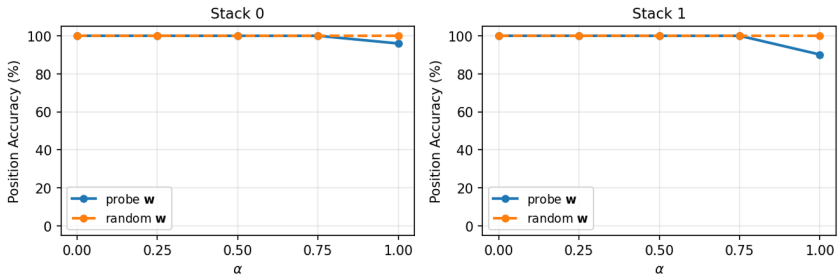

Figure 3: Position accuracy as a function of ablation strength α.

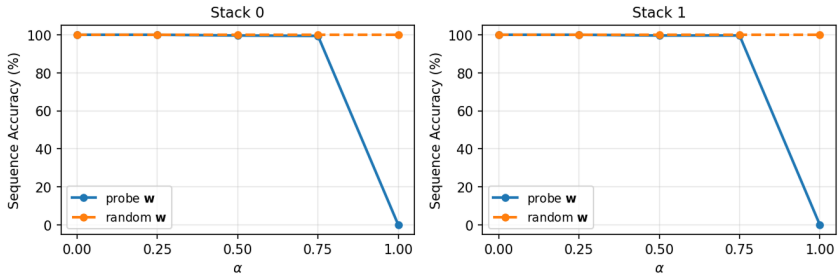

Figure 4: Sequence accuracy as a function of ablation strength α.

A.2 SHUFFLE-4

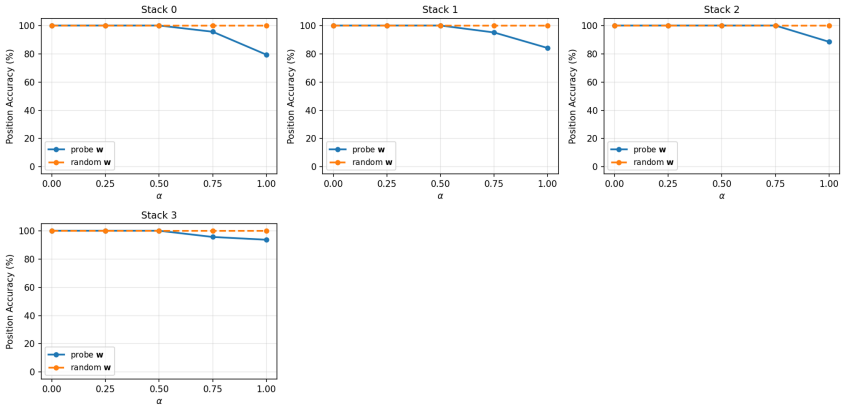

Figure 5: Position accuracy as a function of ablation strength α.

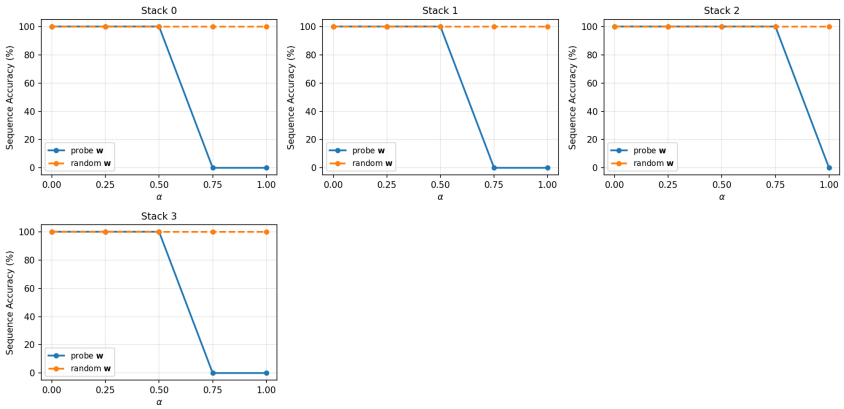

Figure 6: Sequence accuracy as a function of ablation strength α.

A.3 SHUFFLE-8

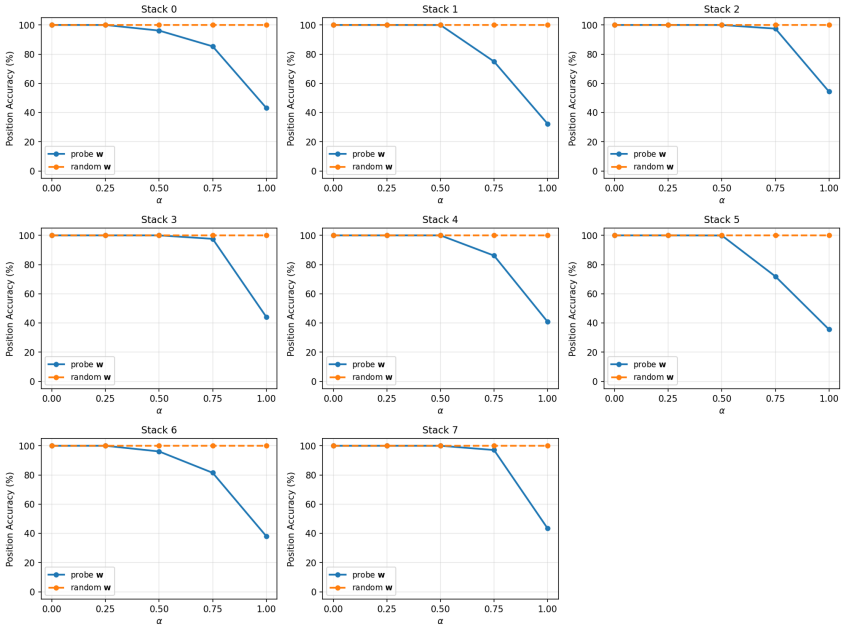

Figure 7: Position accuracy as a function of ablation strength α.

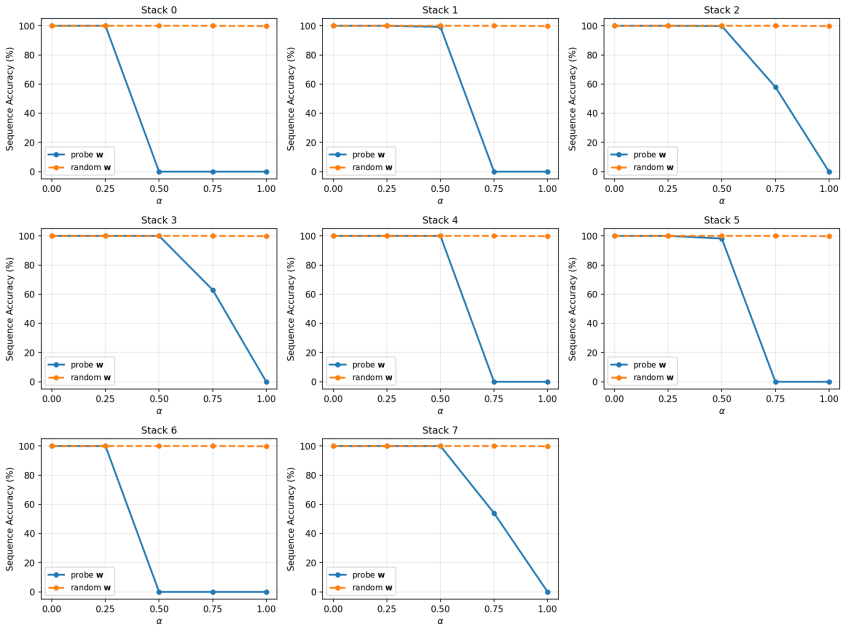

Figure 8: Sequence accuracy as a function of ablation strength α.