# Position Information Emerges in Causal Transformers Without Positional Encodings via Similarity of Nearby Embeddings

**Chunsheng Zuo**
Dept. of Computer Science
Johns Hopkins University
czuo3@jh.edu

**Pavel Guerzhoy**
Dept. of Mathematics
University of Hawai'i at Mānoa
pavel@math.hawaii.edu

**Michael Guerzhoy**
Division of Engineering Science
University of Toronto
guerzhoy@cs.toronto.edu

## Abstract

Transformers with causal attention can solve tasks that require positional information without using positional encodings. In this work, we propose and investigate a new hypothesis about how positional information can be stored without using explicit positional encoding. We observe that nearby embeddings are more similar to each other than faraway embeddings, allowing the transformer to potentially reconstruct the positions of tokens. We show that this pattern can occur in both the trained and the randomly initialized Transformer models with causal attention and no positional encodings over a common range of hyperparameters.

## 1 Introduction

Recent results by Haviv et al. (2022), Kazemnejad et al. (2024), and Chi et al. (2023) suggest that positional encodings are not necessary when training decoder-only Transformer language models. These results motivate our investigation of how Transformers might represent positional information without positional encodings.

As shown in (Tsai et al., 2019; Zuo and Guerzhoy, 2024), the non-causal attention mechanism is equivariant to the permutation of the input tokens — the prediction for input token $n+1$ is invariant to permutations of tokens $1, 2, ..., n-1$. Therefore, without positional encodings, the causal attention mechanism is required for the Transformer to consider the order of the input tokens. Chi et al. (2023) hypothesize that causal attention allows positional information to be stored using the variance (taken across the indices of the embedding vector — essentially the norm) of the embeddings, which generally decreases for tokens at later positions. They argue that the variance will tend to decrease because, when using causal attention, embedding $n$ is computed using embeddings $1, 2, ..., n-1$ in the previous layer, whereas embedding $n+k$ will be computed using $k$ more input embeddings, leading to variance shrinkage for embedding $n+k$.

We identify a different possible way of representing positional information that also arises from the fact that embeddings at earlier positions are computed using fewer embeddings from the previous layer compared to those at later positions. Specifically, we observe that embeddings at nearby indices will tend to be more similar to each other (in the sense of cosine similarity). This property could, in principle, enable the reconstruction of a token's position.

The rest of the paper is organized as follows. We briefly review the literature on causal attention's connection to storing position information in Section 2.1. We then describe the pattern of nearby embeddings' being more similar to each other that we refer to as the *adjacency pattern*, which we later link to the storing of position information in the network. We then present theoretical observations that explain how and why the adjacency pattern arises across many contexts in Section 3. We confirm through experiments on synthetic data that the pattern we report appears in a variety of configurations, both in trained and untrained architectures that use causal attention, in Section 4. We demonstrate a range of synthetic tasks where the position is important in Section 4.2. In Sections 5.1, 5.2, and 5.3, we demonstrate that the pattern of large cosine similarity between nearby embeddings shows up in a variety of settings, for both trained *and* untrained models. In Section 5.4, we point out that Chi et al. (2023)'s theory of the position information's being stored in the embedding variance is insufficient to explain what we observe in our experiments. In Section 5.5, we explore the extent to which position information is stored in different Transformer layers, and to what extent that information can be thought of as being stored in the variance and in the *adjacency pattern*. We discuss our results in Section 6 and discuss some limitations in Section 7.



*Proceedings of the 31st International Conference on Computational Linguistics*, pages 9418–9430
January 19–24, 2025. 

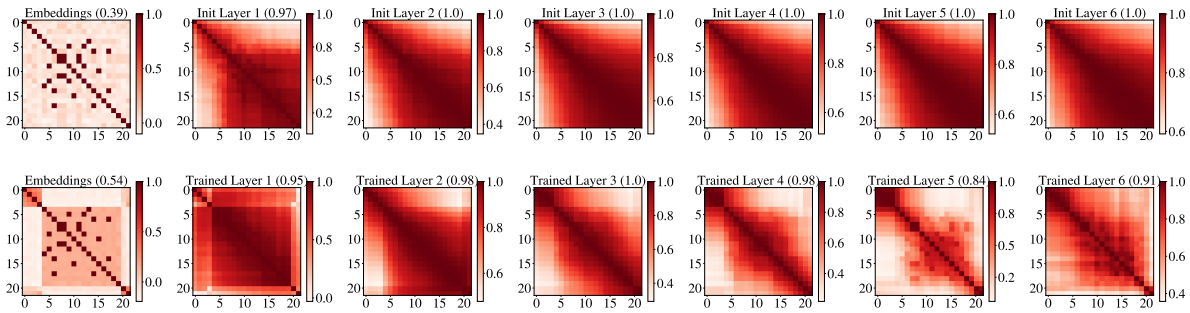



Figure 1: Self-cosine-similarity matrices of randomly initialized (first row) and trained (second row) 6-layer Transformers with causal attention and no positional encodings on the task of Reversal (22). The matrices are produced using a testing sample of 22 tokens, "rev(8502251258017069)=", as input, showing results from the embeddings to the output of layer 6 left to right for the initialized and trained models. The number in the bracket represents the adjacency probability score.

## 2 Background

### 2.1 Transformers with causal attention store position information without positional encodings

Mechanisms analogous to modern attention in Transformers have long been used in recurrent neural networks (Bahdanau et al., 2014; Schmidhuber, 1992). An attention mechanism is central to the Transformer architecture (Vaswani et al., 2017).

In a Transformer with "non-causal" attention, an output at the $k$-th position is agnostic to permutations in the positions of the inputs from other positions, a property known as permutation invariance (the logits above positions $1...k$ are permutation-*equivariant* in the input). Without positional encodings, permutation equivariance prevents the output of each layer from taking into account the position of input tokens. In contrast, Tsai et al. (2019) show that Transformers with causal attention are not permutation-equivariant to the input sequence. This implies the possibility of the success of Haviv et al. (2022) in training causal Transformers without positional encodings — "non-causal" attention could not accomplish that.

### 2.2 The self-cosine-similarity matrix and the adjacency pattern

The self-cosine-similarity matrix is a method to visualize the similarity (in the sense of a small angle) between all pairs of vectors within a sequence of embeddings. To create a self-cosine-similarity matrix $\boldsymbol{C}$ for a sequence of $n$ token embeddings $\boldsymbol{X} \in \mathbf{R}^{\mathbf{n}\times\mathbf{d}}$ of dimension $d$, we define each entry $\boldsymbol{D}_{ij}$ as the cosine similarity between the $i^{th}$ and $j^{th}$ token embeddings, namely, $\boldsymbol{D}_{ij} = \text{similarity}(\boldsymbol{X}_i, \boldsymbol{X}_j) = \cos\theta_{(\boldsymbol{X}_i, \boldsymbol{X}_j)}$ . Since the cosine similarity operation is commutative, $\boldsymbol{D}_{ij} = \boldsymbol{D}_{ji}$, resulting in the self-cosine-similarity matrix's being diagonally symmetrical.

We use the term *adjacency pattern* to describe a special type of self-cosine-similarity matrix that we observe. An example of this pattern can be found in Figure 1, where the matrix is darker (higher values) closer to the diagonal and brighter (lower values) further away, indicating that each embedding vector is more similar to vectors closer to it and less similar to vectors further away from it. The key idea in this paper is that embeddings exhibit an *adjacency pattern*, meaning position information may, in principle, be partially recoverable from them, as embeddings corresponding to spatially nearby positions tend to be more similar.

The self-cosine-similarity matrix is used in (Wang and Chen, 2020) to visualize various positional encodings, some of which, such as the sinusoidal embeddings, demonstrate the *adjacency pattern*. In our work, the self-cosine-similarity matrix is applied to the causal attention's output embeddings directly in order to examine their adjacency pattern.

## 3 How the adjacency pattern arises

Chi et al. (2023) demonstrate that, in the first hidden layer of the causal Transformer, the variance of the individual coordinate values within embeddings goes down with token index $k$. They infer that information about the position is related the the variance of the embedding. Chi et al. (2023) explain the decrease in variance by observing that embedding $k$ is computed using a larger and larger

context as $k$ grows.

In this section, we use the observation to show that we should expect for the adjacency pattern to arise in the first layer after the learned token embeddings.

### 3.1 Empirical evidence

Embeddings at positions $k-1$, $k$, and $k+1$ are computed using the linear combinations of the value vector sets $\{e_1, e_2, \ldots, e_{k-1}\}$, $\{e_1, e_2, \ldots, e_{k-1}, e_k\}$, and $\{e_1, e_2, \ldots, e_{k-1}, e_k, e_{k+1}\}$, respectively, where $e$ are the embeddings.

We first simulate the value vectors used in attention by a set of random normal 128-dimensional vectors $\{v_1, ..., v_k\}$ and the causal attention weights at the 4th, 5th, and 6th row by the following i.i.d. random coefficient sets $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$, $\{\beta_1, \beta_2, \beta_3, \beta_4\}$, $\{\gamma_1, \gamma_2, \gamma_3, \gamma_4\, \gamma_5\, \gamma_6\}$. We then mimic the attention output embeddings at token positions 4, 5, and 6 by the following linear combination of vectors: $a = \left(\sum_{i=1}^{4} \alpha_i v_j, b = \sum_{i=1}^{5} \beta_i v_j, c = \sum_{i=1}^{6} \gamma_i v_j\right)$. Denote the cosine similarity as "$sim$". We want to determine the condition for $sim(a, b)$ to be consistently higher than $sim(a, c)$, as well as for $sim(c, b)$ to be higher than $sim(c, a)$. We simulate with a range of standard deviations $\sigma_{init}$ from the set $\{0.001, 0.01, 0.1, 1, 10, 100\}$, and for each we repeat for 10000 trials and record $sim(a, b) - sim(a, c)$ and $sim(c, b) - sim(c, a)$ for each trial. The resulting histogram is plotted in Figure 2, where the first and second rows are for $sim(a, b) - sim(a, c)$ and $sim(c, b) - sim(c, a)$, respectively. The distribution is narrow and above zero for only small values of $\sigma_{init}$, corresponding to the condition that allows $sim(a, b)$ to be consistently higher than $sim(a, c)$ (same for $sim(c, b)$ and $sim(c, a)$). See also the experimental results in Table 5.

### 3.2 The averaging effect provably arises in the first layer

Here, we show that we can expect that, in the second layer (i.e., the first layer after the embeddings), the angle between embedding $k+t$ and embedding $k+t+1$ is smaller than the angle between embedding $k+t$ and embedding $k+t+2$, implying an adjacency pattern.

Assume embeddings $\{e_1, e_2, ..., e_k, ..., e_n\}$ are high-dimensional and normalized, and therefore approximately orthogonal. We are computing the next layer, with coefficients $\alpha$, $\alpha'$, $\beta$, and $\beta'$.

We would like to show that the angle between $\sum_{i=1}^{k+t} \alpha_i e_i$ and $\sum_{i=1}^{k+t+1} \beta_i e_i$ would tend to be smaller than the angle between $\sum_{i=1}^{k+t} \alpha_i e_i$ and $\sum_{i=1}^{k+t+2} \beta'_i e_i$. The weights $\alpha$, $\beta$, and $\beta'$, which correspond to the attention weight in a causal architecture, would all sum to 1: $\sum_{i=1}^{k} \alpha_i = \sum_{i=1}^{k+t} \beta_i = \sum_{i=1}^{k+t+1} \beta'_i = 1$.

Instead of the angles, we compute the dot products and show that we can expect the difference between the dot products to be positive, namely

$$\left(\sum_{i=1}^{k+1} \alpha_i v_i \cdot \sum_{i=1}^{k+t} \beta_i v_i\right) - \left(\sum_{i=1}^{k+1} \alpha_i v_i \cdot \sum_{i=1}^{k+t+1} \beta'_i v_i\right) > 0.$$

Indeed, the difference between the left and right sides is

$$\begin{aligned} &\sum_{i=1}^{k+1} \alpha_i v_i \cdot \sum_{j=1}^{k+t+1} (\beta_j - \beta'_j) v_j \\ \approx &\sum_{i=1}^{k+1} \alpha_i (\beta_i - \beta'_i) v_i \cdot v_i \\ \approx &||v|| \sum_{i=1}^{k+1} \alpha_i (\beta_i - \beta'_i) > 0, \end{aligned}$$

where the approximate equalities follow from the approximate orthogonality of large n-dimensional vectors normalized to norm 1.

### 3.3 Semantic-level explanation

If embedding $n$ is a summary of all information from positions $1..n-1$, we would expect that embeddings $n = k$ and $n = k+1$ be similar in *some* space.

## 4 The adjacency pattern appears in both non-trained and trained architectures in a variety of configurations

In this section, we explore the settings in which the adjacency pattern in causal Transformers with no positional encodings ("Causal-NoPE") appears. We define the way we measure the adjacency pattern, describe the tasks we are using, and provide the experimental details.

### 4.1 Adjacency probability score

We propose the *adjacency probability score* as a metric to quantify the "intensity" of the adjacency

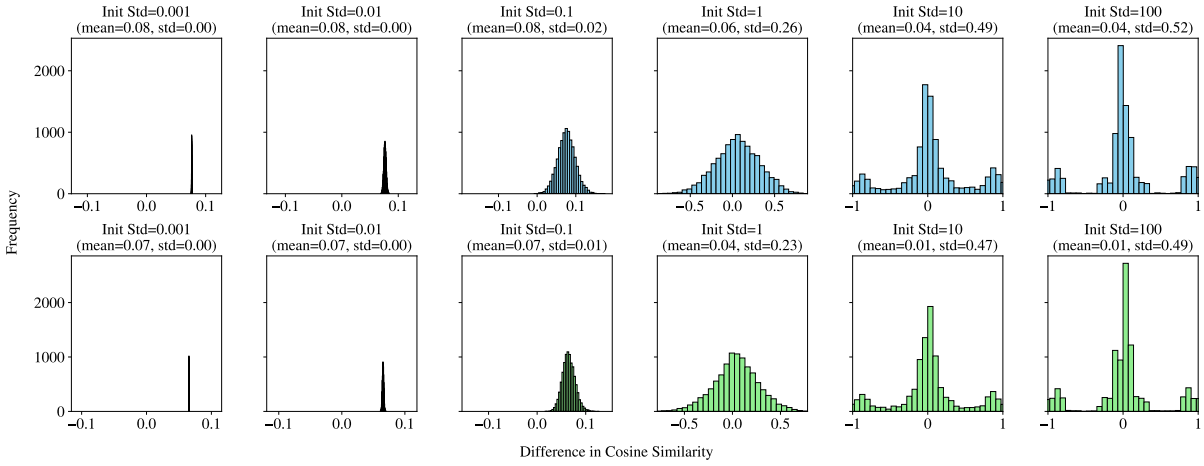



Figure 2: Histograms on the differences between the cosine similarity of nearby tokens and further ones. Images in the first and the second row are for $sim(a, b) - sim(a, c)$, and $sim(c, b) - sim(c, a)$, respectively.

patterns. The score is constructed to correlate with the amount of positional information that can be inferred from the self-similarity matrix.

We compute the proportion of time that the embeddings of tokens with closer positions have higher cosine similarity than those farther away, which can be derived directly from the self-cosine-similarity matrix. Consider the $k^{th}$ row of a squared matrix up to the column of the diagonal entry, denoted by $\boldsymbol{C}_{k1}, \boldsymbol{C}_{k2}, \ldots, \boldsymbol{C}_{kk}$. The row-wise adjacency probability score for this row is defined as:

$$
\begin{aligned}
P_{\text{Adjacency}} &= \mathbb{P}\left(\boldsymbol{C}_{ki} < \boldsymbol{C}_{kj} \text{ if } i < j\right) \\
&= \frac{1}{\binom{k}{2}} \sum_{j=0}^{i} \mathbb{I}\left(\boldsymbol{C}_{ki} < \boldsymbol{C}_{kj}\right)
\end{aligned}
$$

where $\mathbb{I}(\boldsymbol{C}_{ki} < \boldsymbol{C}_{kj})$ is 1 when $\boldsymbol{C}_{ki} < \boldsymbol{C}_{kj}$ and 0 otherwise. The adjacency probability score for the entire self-cosine-similarity matrix is calculated as the average row-wise adjacency probability score of all matrices. Notice that only the lower triangular portion of the matrix is involved in the calculation (see Appendix A).

### 4.2 Tasks

We trained Causal-NoPE Transformers for a variety of tasks that require positional information. The tasks were selected for being trainable from scratch and always requiring positional information.

**Addition:** The Addition task involves generating the completion of strings like `"123+456="`. Following Lee et al. (2024), whose code base we also use, we train NanoGPT to generate the answer in reverse order. The input length (maximum and 90% of the time) is 9 for 3-digit addition (we include strings like `"12+45"` as well).

**Reversal:** The Reversal task requires the model to generate the reversed sequence. For example, for the prompt `"rev(1234)="`, the model is supposed to output `"4321"`. The input length (maximum and 90% of the time) is 22 for reversing 16-or-less-digit numbers.

**Indexing:** The Indexing task requires the model to locate the position of the first occurrence of a number in the sequence. For an example, for the prompt `"wherex(134504392,4)="`, the model is supposed to output `"2"`, which is the index for the first occurrence of `"4"`. The input length (maximum and 90% of the time) is 20 for indexing at most 9 digits.

**Ordering:** Given a sequence of numbers and its reordered version, the Ordering task requires the model to output the new order of the original indices based on the reordered sequence. As an example, for the prompt `"order(67812,28716)="`, the model is supposed to generate the answer `"42130"`. The input length (maximum and 90% of the time) is 18.

### 4.3 Experimental Setup

We first want to examine whether the adjacency pattern persists for models trained for different tasks. We train the baseline 6-layer NanoGPT with 10.6 million parameters on each of the tasks. By default, all models are initialized by the normal distribution $\mathcal{N}(0, 0.02)$. The training for each configuration is repeated for 5 different random seeds. Each task

has 20000 training and 20000 testing samples. Otherwise, the configuration follows the work of Lee et al. (2024), who trained the NanoGPT model to converge on the 3-digit Addition task. All experiments are conducted using an NVIDIA RTX4090 graphics card, with each trial being approximately 15 minutes.

Additionally, we want to compare the effect of different hyperparameters, particularly the number of layers and hidden dimensions. We choose the task of reversal and train models with 6, 12, and 24 layers and 192, 384, and 768 hidden dimensions, respectively, with the same train-test split. Unless further specified, the trained models have achieved more than 90% accuracy in the testing set.

## 5 Results

### 5.1 Transformer from random initialization knows positions right after the first causal attention

We computed the self-cosine-similarity matrix and the adjacency score across settings. Figure 1 is representative of what we observe. In Figure 1, while there is no adjacency pattern in the matrices of the zeroth layer (i.e., the token embeddings), the adjacency pattern starts to appear in the output of the first attention layer and continues in the rest of the layers. The adjacency probability scores in the zeroth layer (i.e., the token embeddings) — $0.39$ and $0.54$ for the randomly initialized and trained models respectively — are much lower than in the other layers (where the minimum is 0.84). In those upper layers, the embeddings have been through at least 1 layer of causal attention. Hence, one layer of causal attention could be sufficient to generate the adjacency pattern.

### 5.2 Adjacency pattern across different models and datasets

The adjacency probability scores of models trained for various tasks and with different hyperparameters (the number of hidden dimensions and the number of layers) are listed in Tables 1, 2, and 3. Each column of the table indicates the location where the embeddings are taken to produce the self-cosine-similarity matrices. Figure 3 presents the adjacency probability scores for the embeddings at each layer, averaged across different tasks. For Table 2 and 3, the Reversal (22) task is chosen to demonstrate the effect of hyperparameters on the adjacency probability score. We observed that the effect of hyperparameters is the same across different tasks.

For most configurations, the adjacency probabilities spike up from around 50% in the token embeddings to more than 80% at the first layer, as well as the rest of the layers. This is consistent regardless of the task type, the training state (initialized/trained), the number of layers, or the dimensions. As a general trend, the adjacency score is the highest for output embeddings in the second layer and declines gradually from there to the end.

### 5.3 The adjacency pattern across different initializations

We further test different initialization schemes, showing that the adjacency pattern is robust for the commonly used initialization schemes. Table 4 and Table 5 show the results for the adjacency probability scores obtained in models initialized by Normal distribution with different means ($\mu_{init} \in \{0, 4, 8\}$) and different standard deviations ($\mu_{init} \in \{0.002, 0.02, 0.2\}$). The highlighted adjacency probability scores indicate a lack of discernible adjacency patterns qualitatively. The adjacency pattern is missing when the mean and the standard deviation are large enough ($\mu_{init} = 4$ and $\sigma_{init} = 0.2$), which are not typical values for initialization. It can be inferred that the mean has a smaller influence than the variance, since the first layer for the model with $\mu_{init} = 4$ can still produce the adjacency pattern. Yet, it is likely that the large $\mu_{init}$ only causes the variance after the first layer to be large, which is why the adjacency pattern for the rest of the layers is removed.

### 5.4 The variance alone may not be sufficient for accurate position information

Chi et al. (2023) propose that the variance of the output embeddings tends to decrease from earlier to later positions, thereby serving as a signal of position information. Hence, we applied the adjacency probability score to the variance of the embeddings to examine how well they are ordered, similar to what has been done to the self-cosine-similarity matrices. Given a sequence of embedding norms of length n, we repeat it n times to form a matrix and apply the same calculation in 4.1 to obtain the adjacency probability score. We also perform this evaluation for each task and put the results in Table 6.

In the trained Causal-NoPE Transformers, there is a much more severe drop in the adjacency score

| Tasks | Embeddings | Layer 1 | Layer 2 | Layer 3 | Layer 4 | Layer 5 | Layer 6 |
|---|---|---|---|---|---|---|---|
| Addition (9) Init | 0.47 | 0.99 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| Addition (9) Trained | 0.48 | 0.95 | 0.98 | 0.99 | 0.98 | 0.88 | 0.85 |
| Reversal (22) Init | 0.49 | 0.97 | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 |
| Reversal (22) Trained | 0.58 | 0.91 | 0.98 | 0.99 | 0.88 | 0.82 | 0.83 |
| Indexing (20) Init | 0.49 | 0.98 | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 |
| Indexing (20) Trained | 0.55 | 0.80 | 0.96 | 0.96 | 0.88 | 0.79 | 0.83 |
| Ordering (18) Init | 0.49 | 0.98 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| Ordering (18) Trained | 0.56 | 0.89 | 0.98 | 0.96 | 0.77 | 0.80 | 0.76 |

Table 1: Averaged Layer-wise adjacency probability score for the 4 tasks, with initialization and trained results, each averaged over 256 samples. The number in the parentheses beside each task indicates the length (maximum and most frequent) of the equations in the task.

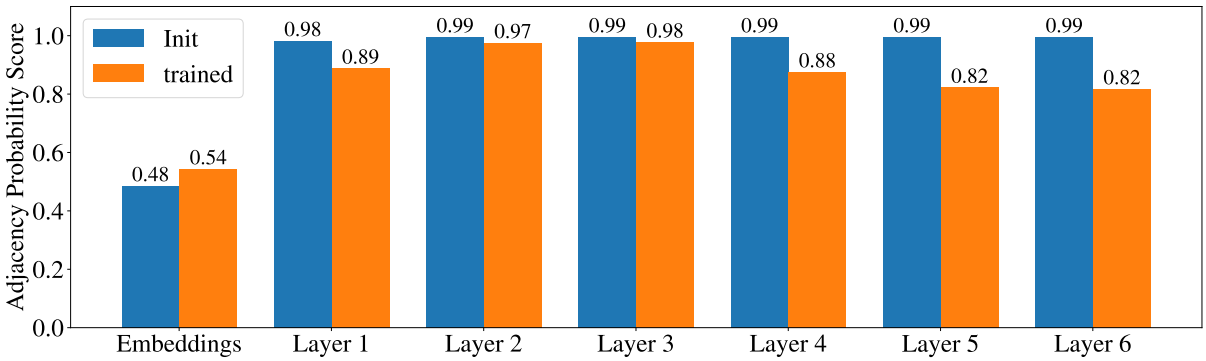



Figure 3: The layer-wise adjacency probability score for randomly initialized and trained models averaged over the 4 tasks, correspond to the values presented in Table 1.

of the norms than in the self-cosine-similarity matrices. Figure 7 presents a visualization of the average results for Table 6 across the 4 tasks. Compared to Figure 3, it is clear that the average adjacency probability scores of the norms for the trained model are lower than for the self-cosine-similarity matrix. As an example, Figure 10 and Figure 11 show the self-cosine-similarity matrices and norms for the initialized and trained model on the same task (Reversal (22)) with the same input. Though the norms tend to be monotonically decreasing at the initialized layers, they are not necessarily ordered in the trained layers, with the last layer even showing a reversed order. Even in comparison with the self-cosine-similarity matrices at the trained layers, except for the first layer, the norms are generally worse at indicating clear position information than the adjacency matrices.

### 5.5 Probing for position information

We further compare variance to cosine similarity by the effectiveness of using them as a feature to probe the position information. For each layer of the Causal-NoPE, the probe is trained to predict the position of an attention output vector using one of the following features as input: the output vector embeddings itself, its variance, and the cosine similarity between an output vector embeddings and the vector at the last position. The probe is a 4-layer Multi-Layer Perception (MLP) with 3 ReLU activation functions in between. To prevent the probe from memorizing the samples (Hewitt and Liang, 2019), the training and testing datasets for probing consist of random digits from 5 to 9 and 0 to 4, respectively, which are all contained in the 4 synthetic tasks 4.2. We fix the input length to 32 and the training and testing sample size to 1600 each. The Root Mean Squared Error normalized by the input length (NRMSE) and the Pearson-R values are presented in Figure 4. We verify the

| Layers | Embeddings | Layer 1 | Layer 2 | Layer 3 | Layer n-2 | Layer n-1 | Layer n |
|---|---|---|---|---|---|---|---|
| 6 | 0.58 | 0.91 | 0.98 | 0.99 | 0.88 | 0.82 | 0.83 |
| 12 | 0.49 | 0.90 | 0.93 | 0.96 | 0.86 | 0.81 | 0.84 |
| 24 | 0.51 | 0.84 | 0.94 | 0.84 | 0.90 | 0.78 | 0.75 |

Table 2: Layer-wise adjacency probability score for models with different numbers of layers trained on the Reversal (22) task, each averaged over 256 samples.

| Dimensions | Embeddings | Layer 1 | Layer 2 | Layer 3 | Layer 4 | Layer 5 | Layer 6 |
|---|---|---|---|---|---|---|---|
| 192 | 0.49 | 0.93 | 0.96 | 0.96 | 0.92 | 0.81 | 0.73 |
| 384 | 0.58 | 0.91 | 0.98 | 0.99 | 0.88 | 0.82 | 0.83 |
| 768 | 0.50 | 0.96 | 0.96 | 0.95 | 0.94 | 0.90 | 0.93 |

Table 3: Layer-wise adjacency probability score for models with different numbers of hidden dimensions trained on the Reversal (22) task, averaged over 256 samples. The only configuration that did not achieve more than 90% accuracy is the model with 192 dimensions, which has an accuracy of 56%. Yet, we observed that in most cases where the model makes an error, the majority of digits are correct, with only a few being incorrect.

validity of the results, showing that when using the output embeddings as features, the probe's testing performance in the setting of the untrained Causal-NoPE Transformers embeddings is the worst (in Appendix B Figure 8).

In almost all layers of the initialized and trained models, using the cosine similarity values as the input feature produces the best probing outcomes with the lowest NRMSE and highest Pearson-R value. In addition to the adjacency probability score results, this probing result further demonstrates the robustness of inter-token cosine similarity as a positional indicator. In comparison, the variance seems less informative. In trained Causal-NoPE Transformers, from layers 2 to 6, the Correlation Coefficients of probes produced from the variance is worse than from the embeddings. This implies that if the Causal-NoPE Transformers learn to synthesize some absolute position information, it should rely on some characteristics of the embeddings more than just the variance.

## 6 Discussion

### 6.1 Is the adjacency pattern unique to causal attention?

Yes. We also applied a self-cosine-similarity matrix to Transformers with "vanilla" attention and confirmed that there is no adjacency pattern. An example is shown in Figure 5, where the self-cosine-similarity matrices look random and the adjacency scores are low. There is a learned absolute positional encoding added to the token embeddings of this model only to let the model converge.

## 7 Limitations

The claims that the paper makes are partly based on empirical analyses of particular Transformer architectures, and using particular datasets. The observations would not necessarily generalize to other architectures. While an attempt was made to construct synthetic datasets that are interesting and display a variety of features, we do not mathematically prove that the observations we make would generalize to any dataset, and in fact it is likely that there could exist datasets to which our observations would not generalize.

## 8 Conclusions and future work

In Transformers with causal attention and no positional encodings, the adjacency pattern can occur for models with a wide range of hyperparameters, including the number of layers, hidden dimensions, and initialization schemes. It exists in the output embeddings of the Transformer's first causal attention layer and persists throughout the rest of the layers. For randomly initialized weights, the adjacency pattern can be observed for various initializations, especially for the ones commonly occurring in practice. For trained models, it is typical that the adjacency pattern in the first few layers is more prominent than in later ones, which we consider reasonable because knowing enough position information in the earlier layers may allow the models to focus on other more contextual information required by the tasks in later layers.

Neither the adjacency pattern nor the in-embedding variance of (Chi et al., 2023) can likely fully account for the fact that we are able to ob-

| $\mu_{init}$ | Embeddings | Layer 1 | Layer 2 | Layer 3 | Layer 4 | Layer 5 | Layer 6 |
|---|---|---|---|---|---|---|---|
| 0 | 0.52 | 0.97 | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 |
| 4 | 0.49 | 0.97 | 0.96 | 0.95 | 0.99 | 0.96 | 0.98 |
| 8 | 0.47 | 0.97 | **0.56** | **0.53** | **0.57** | **0.63** | **0.65** |

Table 4: Layer-wise adjacency probability score for models initialized by Gaussian distribution with different means $\mu_{init}$, averaged over 256 samples from the Reversal (22) tasks.

| $\sigma_{init}$ | Embeddings | Layer 1 | Layer 2 | Layer 3 | Layer 4 | Layer 5 | Layer 6 |
|---|---|---|---|---|---|---|---|
| 0.002 | 0.46 | 0.97 | 0.98 | 0.98 | 0.99 | 0.98 | 0.99 |
| 0.02 | 0.51 | 0.97 | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 |
| 0.2 | 0.49 | **0.55** | **0.58** | **0.66** | **0.70** | **0.73** | **0.68** |

Table 5: Layer-wise adjacency probability score for models initialized by Gaussian distribution with different standard deviation $\sigma_{init}$, averaged over 256 samples from the Reversal (22) tasks.

tain 100% performance on position-sensitive tasks since the probing results indicate that both are not 100% informative. Nevertheless, we believe that the adjacency pattern provides another piece of the puzzle.

## Acknowledgments

We thank Prof. Ran Gilad-Bachrach for useful discussion. We thank Prof. Jonathan Rose for the conversation that initiated this investigation.

## References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Ta-Chung Chi, Ting-Han Fan, Li-Wei Chen, Alexander Rudnicky, and Peter Ramadge. 2023. Latent positional information is in the self-attention variance of transformer language models without positional embeddings. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1183–1193, Toronto, Canada. Association for Computational Linguistics.

Adi Haviv, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy. 2022. Transformer language models without positional encodings still learn positional information. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1382–1390, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2024. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36.

Nayoung Lee, Kartik Sreenivasan, Jason D Lee, Kangwook Lee, and Dimitris Papailiopoulos. 2024. Teaching arithmetic to small transformers. *International Conference on Learning Representations*.

Jürgen Schmidhuber. 1992. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4344–4353, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yu-An Wang and Yun-Nung Chen. 2020. What do position embeddings learn? an empirical study of pre-trained language model positional encoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6840–6849, Online. Association for Computational Linguistics.

Chunsheng Zuo and Michael Guerzhoy. 2024. Breaking symmetry when training transformers. *NAACL Student Research Workshop*.

| Tasks | Embeddings | Layer 1 | Layer 2 | Layer 3 | Layer 4 | Layer 5 | Layer 6 |
|---|---|---|---|---|---|---|---|
| Addition (9) Init | 0.45 | 0.96 | 0.98 | 0.99 | 0.99 | 0.99 | 0.98 |
| Addition (9) Trained | 0.38 | 0.93 | 0.84 | 0.92 | 0.89 | 0.84 | 0.51 |
| Reversal (22) Init | 0.47 | 0.90 | 0.95 | 0.95 | 0.97 | 0.95 | 0.97 |
| Reversal (22) Trained | 0.66 | 0.98 | 0.61 | 0.93 | 0.88 | 0.53 | 0.63 |
| Indexing (20) Init | 0.49 | 0.94 | 0.98 | 0.99 | 0.99 | 0.99 | 0.98 |
| Indexing (20) Trained | 0.74 | 0.93 | 0.95 | 0.98 | 0.91 | 0.90 | 0.85 |
| Ordering (18) Init | 0.55 | 0.94 | 0.99 | 0.99 | 0.99 | 0.98 | 0.98 |
| Ordering (18) Trained | 0.22 | 0.95 | 0.76 | 0.94 | 0.71 | 0.72 | 0.43 |

Table 6: Layer-wise adjacency probability score of the variance of embeddings for the 4 tasks, with initialization and trained results. The number in parentheses beside each task indicates the input length involved in the task.

## A More about the adjacency probability score

We only consider each row up to the diagonal because, for causal attention, each self-cosine-similarity matrix $S \in \mathbb{R}^{n \times n}$ of size n contains n sub-matrices, from $S_1 \in \mathbb{R}^{1 \times 1}$ to $S_n \in \mathbb{R}^{n \times n}$. For a sub-matrix of length $k \in [1, .., n]$, it is formed by embeddings resulting exactly from the first $k$ out of $n$ tokens of the original sequence. Therefore, each row-wise adjacency probability score at row $k$ measures the last row of sub-matrix $S_k$. Another way to think of this is that causal attention at the current token only considers anything before it. Hence, we measure just the adjacency probability score for anything up to the current token, which is up to the diagonal of each row.

Figure 6 demonstrates different adjacency probability scores with their respective sample matrix. A higher adjacency probability score can be interpreted as the model being more likely to know the exact ordering of other tokens before a certain token. Meanwhile, although a zero adjacency probability score will also allow the model to know the token order oppositely, it is unachievable in a self-cosine-similarity matrix unless all embeddings are the same. For random matrices, the adjacency probability score is about 0.5.

## B Visualizations for probing results

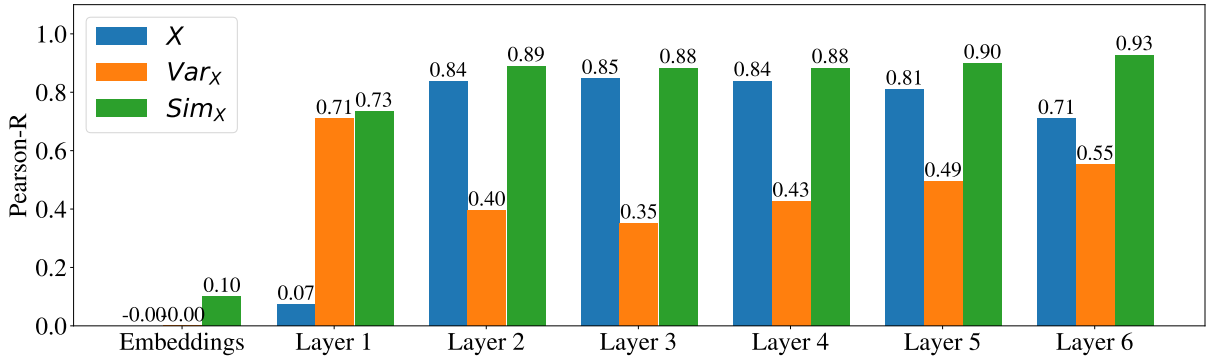
Figure 8 shows the probing results on randomly initialized Causal-NoPE Transformers. The poor performance from using the embeddings as input indicates that the models do not contain any fixed/absolute positional information from the beginning, whereas the descent performance from using the cosine similarity as input suggests the existence of relative positional information inherent to the causal attention.

Figure 9 demonstrates the prediction of probes trained using various input features of the trained Causal-NoPE Transformers.

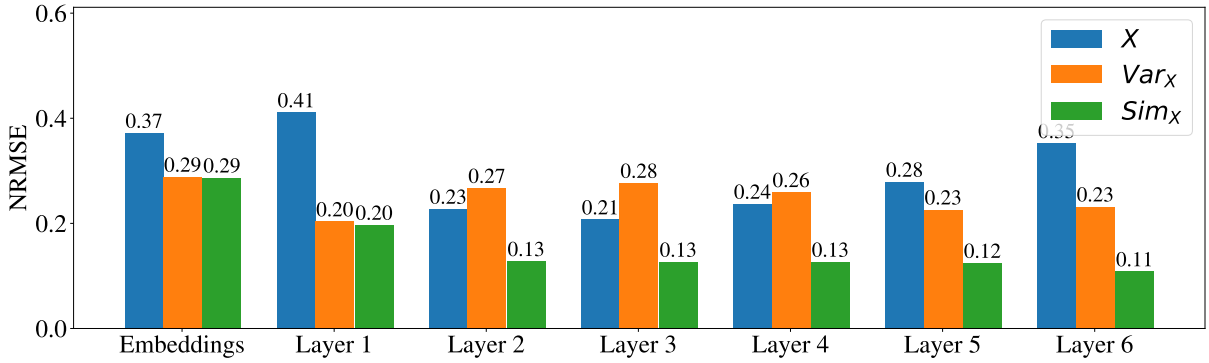
## C More Visualizations of Experimental Results

Figure 12 provides an example of a model with 12 layers for the indexing task.

To determine if there are clusters of samples that exhibit extremely low to extremely high values, we check the distributions of the adjacency scores for all configurations. Typically, we observe distributions like the ones in Figure 13 indexing task. In this example, while the distributions of adjacency scores concentrate around 1 for the untrained model, after training, only the adjacency scores for layer 2 and layer 3 distribute densely and closely to 1. In particular, the adjacency scores are the highest and most concentrated in layer 3 of the trained model, to an extent that matches the ones in the untrained model. We interpret these observations as an indication that the model learns to keep the adjacency pattern in earlier layers and gradually discard it in later ones.




(a)




(b)

Figure 4: Average layer-wise probing results for trained Causal-NoPE Transformers of (a) Pearson-R and (b) Normalized Root Mean Squared Error (NRMSE) using one of the following as the input: the output vector embeddings $X$, their variance $Var_X$, and the cosine similarity between the output vector embeddings and the vector at the last position $Sim_X$.

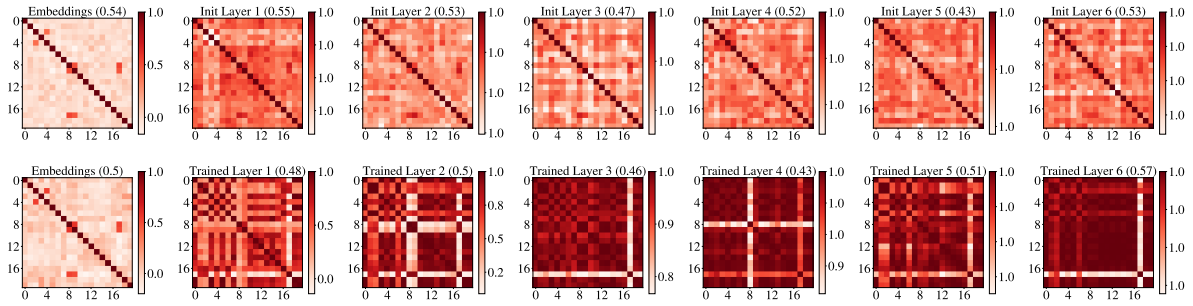



Figure 5: Self-cosine-similarity matrices of randomly initialized (first row) and trained (second row) 6-layer Transformers with normal attention and learned absolute positional encodings on the task of Indexing (20). The matrices are produced using a testing sample of 20 tokens, "wherex(299517340,9)=", as input.

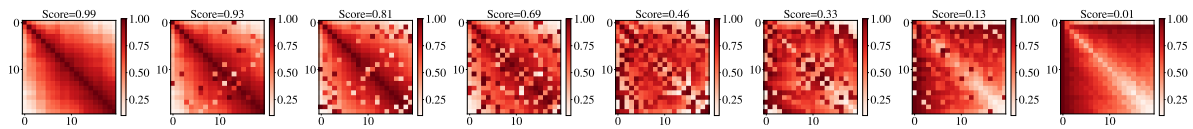



Figure 6: Synthetic matrices with different adjacency probability score values. (See Appendix A

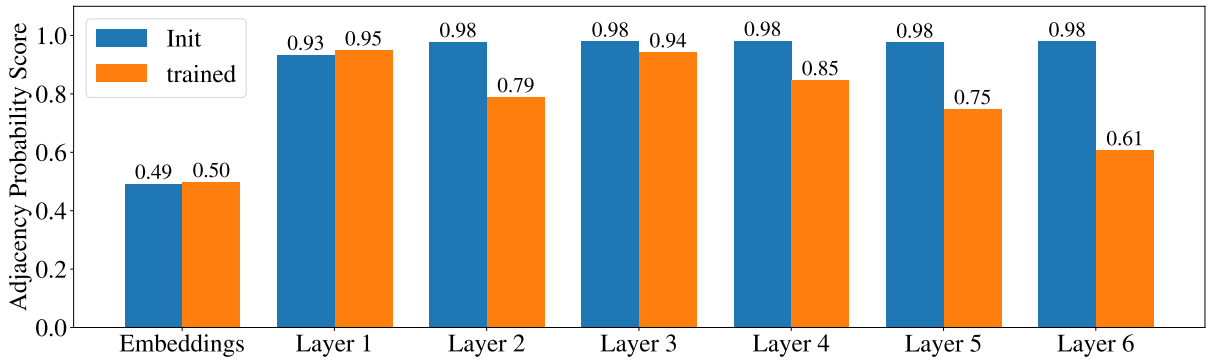
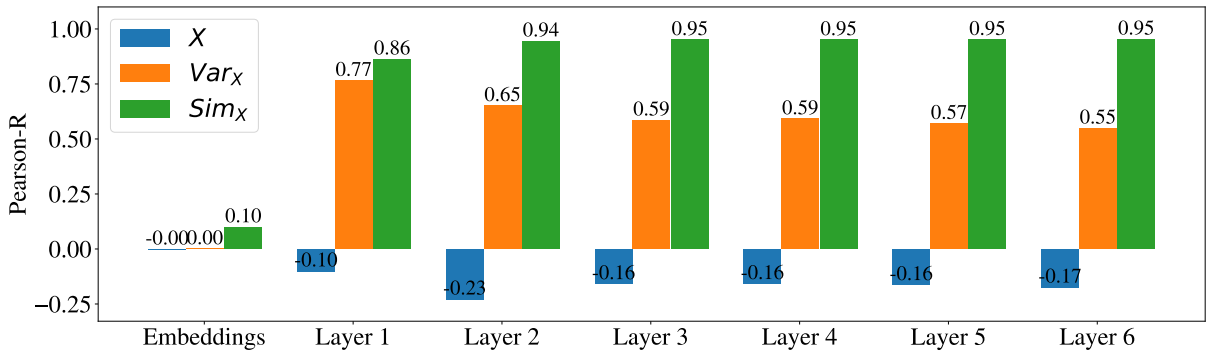
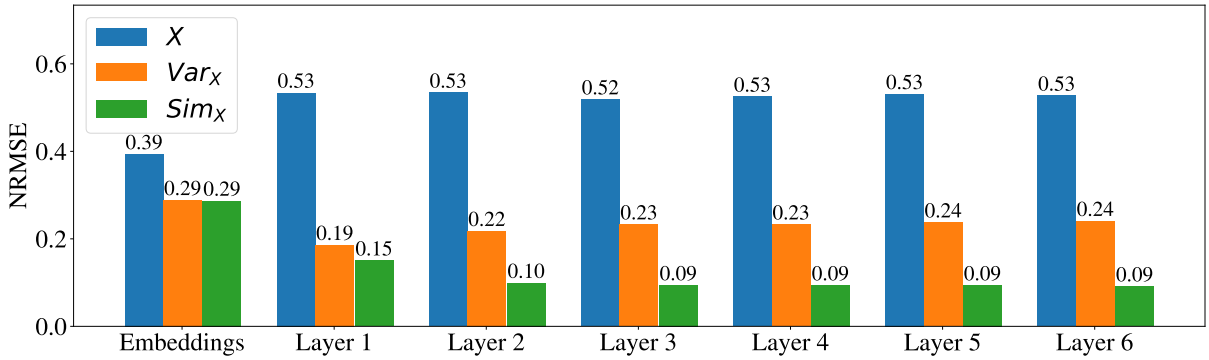



Figure 7: The layer-wise adjacency probability score of the norms for randomly initialized and trained models averaged over the 4 tasks, correspond to the values presented in Table 6.

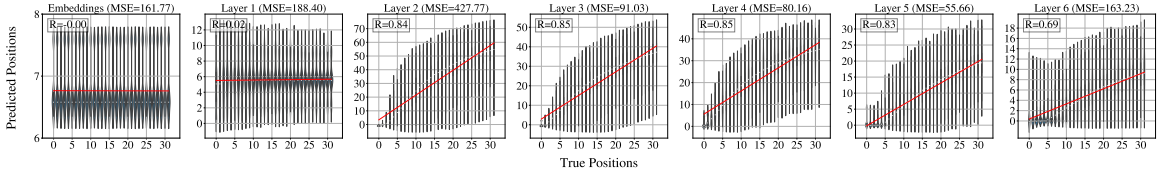



(a)

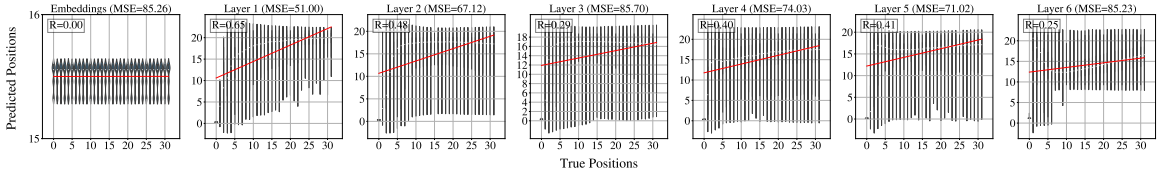



(b)

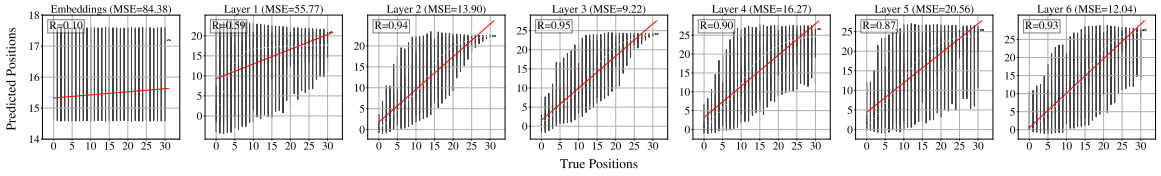
Figure 8: Average layer-wise probing results for initialized Causal-NoPE Transformers of (a) Pearson-R and (b) Normalized Root Mean Squared Error (NRMSE) using one of the following as the input: the output vector embeddings $X$, their variance $Var_X$, and the cosine similarity between the output vector embeddings and the vector at the last position $Sim_X$.




(a)




(b)




(c)

Figure 9: Violin plots for the test predictions of a trained probe for a Causal-NoPE Transformer trained on the ordering task. The 3 different features, (a) embeddings, (b) variance, and (c) cosine similarity, are used independently.

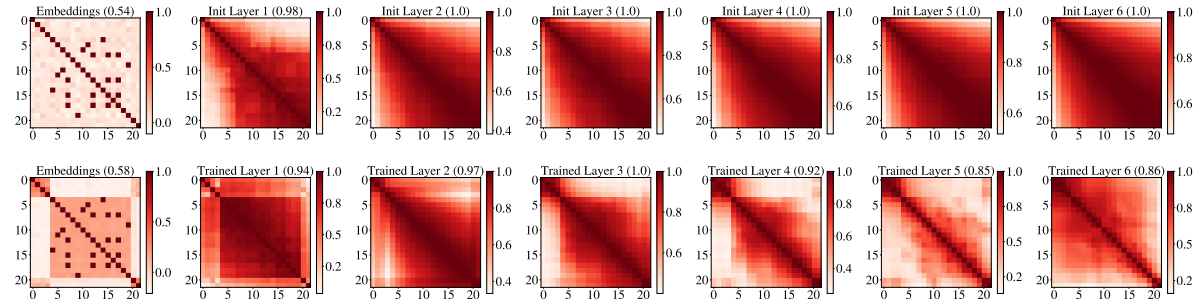



Figure 10: Layer-wise self-cosine-similarity matrices of randomly initialized (first row) and trained (second row) Causal-NoPE Transformers on the task of ordering, with "rev(1849364897192906)=" as the input.

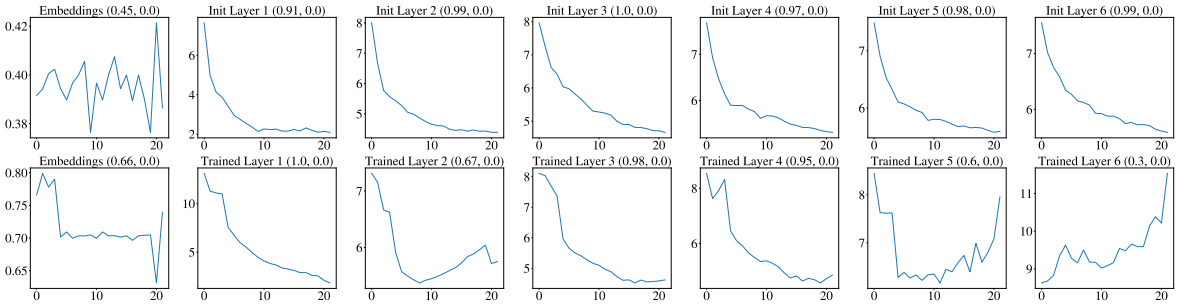



Figure 11: Layer-wise embedding norms for randomly initialized (first row) and trained (second row) Causal-NoPE Transformers on the task of Reversal (22), with "rev(1849364897192906)=" as the input.

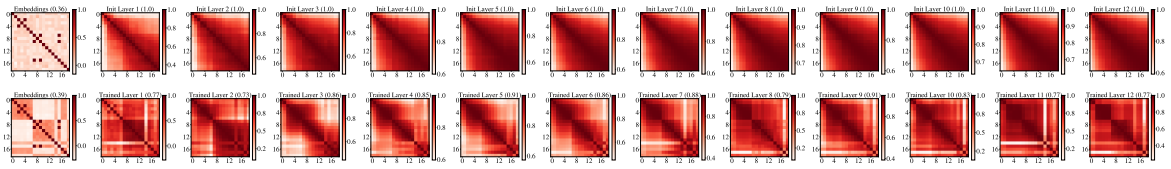


Figure 12: Self-cosine-similarity matrices of randomly initialized (first row) and trained (second row) 12-layer Transformers with causal attention and no positional encodings on the task of Indexing. The matrices are produced using a testing sample of 22 tokens, "wherex(8483561,8)=0", as input, showing results from the embeddings to the output of layer 12 left to right for the initialized and trained models. The number in the bracket represents the adjacency probability score.

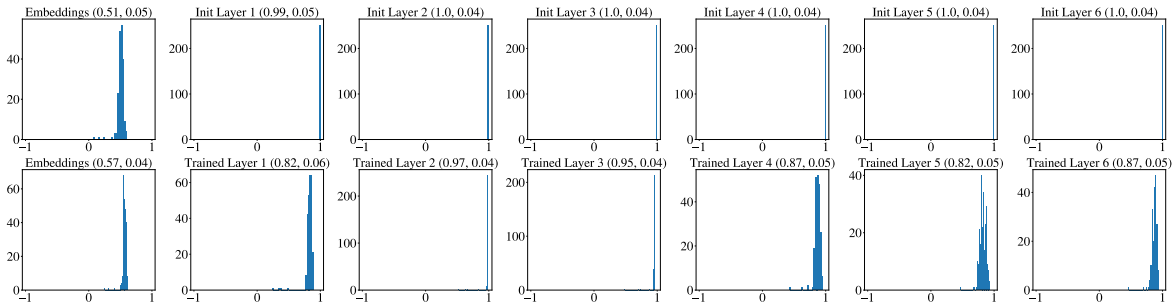



Figure 13: Distribution of adjacency probability score for a model before and after training ("Init"/"Trained") on the indexing task. The sample size of the histograms is 256. The two numbers inside the brackets of the subplot titles are the distribution's mean and standard deviation. Notice that the 7 pairs of means and standard deviations for the trained model (the second row) correspond to the values presented in Table 1 for the indexing task.