Transformers on Markov Data: Constant Depth Suffices

Nived Rajaraman [1] **Marco Bondaschi** [2] **Kannan Ramchandran** [1] **Michael Gastpar** [2] **Ashok Vardhan Makkuva** [2]

Abstract

Attention-based transformers have been remarkably successful at modeling generative processes across various domains and modalities. In this paper, we study the behavior of transformers on data drawn from k^{th}-order Markov processes, where the conditional distribution of the next symbol in a sequence depends on the previous k symbols observed. We observe a surprising phenomenon empirically which contradicts previous findings: when trained for sufficiently long, a transformer with a fixed depth and 1 head per layer is able to achieve low test loss on sequences drawn from k^{th}-order Markov sources, even as k grows. Furthermore, this low test loss is achieved by the transformer's ability to represent and learn the in-context conditional empirical distribution. On the theoretical side, our main result is that a transformer with a single head and three layers can represent the in-context conditional empirical distribution for k^{th}-order Markov sources, concurring with our empirical observations. Along the way, we prove that *attention-only* transformers with $O(\log_2(k))$ layers can represent the in-context conditional empirical distribution by composing induction heads to track the previous k symbols in the sequence. These results provide more insight into our current understanding of the mechanisms by which transformers learn to capture context, by understanding their behavior on Markov sources. Code is available at: `https://github.com/Bond1995/Constant-depth-Transformers`

1. Introduction

Attention-based transformers have revolutionized the field of natural language processing (NLP) (Vaswani et al., 2017; Brown et al., 2020) and beyond (Dosovitskiy et al., 2021; He et al., 2021), achieving significant performance gains across tasks like machine translation, text generation, and sentiment analysis. A key factor in their success is their ability to model sequences far more efficiently, and the ability to learn in-context (Bietti et al., 2023; Edelman et al., 2024).

To understand this capability, a canonical approach is to sample the input from a *k^{th}-order Markov process*, where the next symbol's conditional distribution depends only on the previous k symbols. Recent studies (Makkuva et al., 2024; Edelman et al., 2024; Nichani et al., 2024) have investigated the ability of transformers to learn Markov processes and establish that learning happens in phases. The transformer eventually learns to represent the conditional k-gram model, which is the in-context MLE of the Markov process.

The results in (Edelman et al., 2024; Nichani et al., 2024) seem to suggest that for low depth transformers to learn Markov processes of order k, it is essential that the number of heads scale linearly in k. At first glance, this is a bit concerning - real world data generating processes often contain long-range dependencies. How is it that transformers succeed at capturing these kinds of long-range dependencies, while at the same time requiring so many heads to be able to capture the necessary context for k^{th}-order Markov sources?

To understand the nature of this phenomenon, we train low-depth transformers on k^{th}-order Markov sources. These experiments result in two surprising empirical phenomena that seem to contradict previous findings: when trained for sufficiently long, (i) a 2-layer, 1-head transformer can learn k^{th}-order Markov processes for k as large as 4, (ii) a 3-layer, 1-head transformer is able to achieve low test loss on sequences drawn from k^{th}-order Markov sources, even as k grows to be as large as 8 (Fig. 4). In both cases, the values of k for which the models appear to learn k^{th}-order Markov sources are much higher than those predicted in prior experiments (Edelman et al., 2024; Nichani et al., 2024). This discrepancy shows that our understanding of the mechanisms used by transformers to learn k^{th}-order Markov processes is not complete and raises a broader question:

> *What is the interplay between depth, number of heads and non-linearity in learning k^{th}-order Markov processes?*

[1] UC Berkeley [2] EPFL. Correspondence to: Nived Rajaraman <nived.rajaraman@berkeley.edu>.

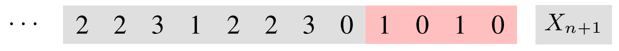

Figure 1: k^{th}-order Markov processes for $k = 4$. The next symbol X_{n+1} in the sequence is sampled from the distribution $P(\cdot|X_n, X_{n-1}, X_{n-2}, X_{n-3})$ which only depends on the last $k(= 4)$ symbols (marked in red).

In this paper, we approach this question from the point of view of representation power, and provide some partial explanations toward the phenomena illustrated previously.

Our main contributions are as follows:

1. We show, rather surprisingly, that the standard transformer architecture with 3 layers and 1 head per layer is capable of representing the conditional k-gram model (Definition 2.1), and thereby learn k^{th}-order Markov models in-context.

2. Along the way to building up to this result, we consider the simpler family of *attention-only transformers* and show that they can represent the conditional k-gram model with $\lceil \log_2(k + 1)\rceil$ layers.

3. Under a natural assumption on the nature of the attention patterns learnt by the transformer, we then argue that for $k \geqslant 3$ attention-only transformers *need* at least $\lceil 1 + \log_2(k - 2)\rceil$ layers to represent a "k^{th}-order induction head" (Definition 4.3). Empirically, transformers are observed to learn k^{th}-order induction heads whenever they achieve small test error (Edelman et al., 2024).

The last result is a consequence of a more general tradeoff between the number of layers, L, and heads per layer, H, an attention-only transformer requires to represent a k^{th}-order induction head, under a natural assumption on the learnt attention patterns. In conjunction, these results also reveal the role of non-linearities (aside from the softmax in the attention) in the transformer architecture. In particular, it appears that layer normalization plays a critical role in the ability of constant-depth transformers to learn the conditional k-gram model. Together with the experimental results mentioned previously, these results paint a more comprehensive picture about the representation landscape of transformers in the context of k^{th}-order Markov processes.

	Attention-only		Standard
Layers (L)	2	$\lceil \log_2(k+1)\rceil$	3
Heads (H)	k	1	1

Table 1: Each column in this table indicates that there is a transformer with L layers and H heads in the first layer which can represent the conditional k-gram model.[1]

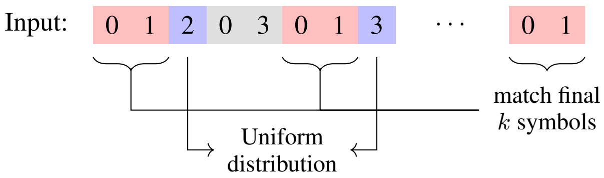

Figure 2: Conditional k-gram model. The conditional k-gram is the in-context estimate of the Markov process and is realized in two steps. The first step is to find the locations in the sequence (marked red) which match the final k symbols (functionally, a k^{th}-order induction head). The conditional k-gram model returns the uniform distribution over the next symbol at these locations (marked blue).

Notation. Scalars are denoted by italic lower case letters like x, y and Euclidean vectors and matrices in bold $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{M}$, etc. The notation $\mathbf{0}_{p\times q}$ (resp. $\mathbf{1}_{p\times q}$) refers to the all-zero (resp. all-one) matrix. When it is clear from the context, we omit the dimensions of a matrix. Define $[S] \triangleq \{1, 2, \ldots, S\}$ for $S \in \mathbb{N}$. $\mathbb{I}(\cdot)$ denotes the indicator function and $\mathrm{Unif}(S)$ denotes the uniform distribution over a set S.

1.1. Related work

There is a large body of active research focused on studying different aspects of transformer models (Weiss et al., 2021; Giannou et al., 2023; Oymak et al., 2023; Li et al., 2023). Our work closely relates to the aspects of understanding the representation power of transformers, and in-context learning. (Yun et al., 2020; Pérez et al., 2021; Wei et al., 2022a) study the representation capabilities of transformers and show properties such as universal approximation and Turing-completeness. Viewing transformers as sequence to sequence models, (Liu et al., 2023; Bhattamishra et al., 2020) study their ability to model formal languages and automata. Along more related lines to our work, (Sanford et al., 2024; 2023) present logarithmic depth transformer constructions for representing a k-hop generalization of the notion of an induction head (Olsson et al., 2022). On the other hand the theoretical and mechanistic understanding of in-context learning (Wei et al., 2022b) has received much attention lately (Bai et al., 2023; Lin and Lee, 2024; Akyürek et al., 2023; Hoogland et al., 2024), focusing on different operating regimes and phases of learning. There are a few recent papers which study the behavior of transformers when

[1] The requisite embedding dimension and bit-precision to achieve a target additive approximation is discussed in more detail in Sections 4 and 5

trained on data generated from Markov processes, and generalizations thereof (Bietti et al., 2023; Rajaraman et al., 2024). In particular, (Makkuva et al., 2024; Nichani et al., 2024) study the optimization landscape of gradient descent in learning generalizations of Markov processes, and (Edelman et al., 2024) present a study of how transformers learn to represent in-context k-gram models, focusing on different phases of learning.

2. Preliminaries

We provide the necessary background for Markov processes, the conditional k-gram model, and the transformer architecture.

2.1. Markov processes

Markov processes are one of the widely used models in sequence modeling (Norris, 1997). The characterizing property of these processes is that at any time step, the future evolution is only influenced by the most recent states. More formally, a sequence $(X_n)_{n\geqslant 1}$ is a k^{th}-order Markov process on a finite state space $[S]$ with the transition kernel P, if surely,

$$P\big(X_{n+1} \mid X_1, \cdots, X_n\big) = P\big(X_{n+1} \mid X_{n-k+1}, \cdots, X_n\big)$$

This property allows us to capture the conditional distribution at any position using only its previous k symbols. This motivates the notion of a conditional k-gram, its empirical counterpart, defined for any sequence $(x_1, \ldots, x_n)$.

Definition 2.1 (Conditional k-gram model)**.** Given a sequence $(x_1, \cdots, x_n)$ of length n in $[S]^n$, the conditional k-gram model $\widehat{\Pr}_k(\cdot \mid x_1, \cdots, x_n)$ corresponds to the in-context estimate of the distribution over symbols conditioned on the last k symbols, i.e. for $x \in [S]$,

$$\begin{aligned}&\widehat{\Pr}_k(x \mid x_1, \cdots, x_n)\\ &\quad\triangleq \frac{\sum_{i=k+1}^{n} \mathbb{I}(x_i = x, x_{i-1} = x_n, \ldots, x_{i-k} = x_{n-k+1})}{\sum_{i=k+1}^{n} \mathbb{I}(x_{i-1} = x_n, \ldots, x_{i-k} = x_{n-k+1})},\end{aligned} \tag{1}$$

which is defined only so long as the denominator is nonzero. This structure is illustrated in Figure 2. It is well known that the conditional k-gram in Eq. (1) with Laplace smoothing corresponds to the Bayes optimal estimate of the next symbol probability, when the data is drawn from fixed Markov process sampled from a prior distribution (Norris, 1997).

In our experiments, we will consider k^{th}-order Markov kernels sampled from a Dirichlet prior with parameter $\mathbf{1}$. Namely, the transition $P(\cdot|X_1 = i_1, \cdots, X_k = i_k)$ is sampled independently and uniformly on the S-dimensional simplex Δ_1^S, for each tuple $(i_1, \cdots, i_k)$.

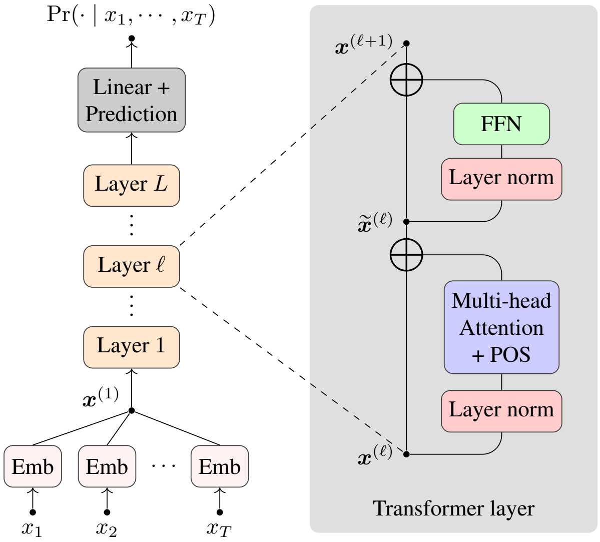
$\Pr(\cdot \mid x_1, \cdots, x_T)$
Linear + Prediction
Layer L
Layer ℓ
Layer 1
$\boldsymbol{x}^{(1)}$
Emb
Emb
Emb
x_1
x_2
x_T
$\boldsymbol{x}^{(\ell+1)}$
FFN
Layer norm
$\widetilde{\boldsymbol{x}}^{(\ell)}$
Multi-head Attention + POS
Layer norm
$\boldsymbol{x}^{(\ell)}$
Transformer layer

Figure 3: Transformer architecture. POS refers to the relative position encodings.

2.2. Transformer architecture

In this paper, we will consider variants of the standard transformer architecture in Figure 3 introduced in (Vaswani et al., 2017), with the goal to understand the role of depth and the non-linearities in the architecture. The simplest variant removes all the layer normalization and the (non-linear) feedforward layer, and is referred to as an *attention-only* transformer. The L-layer 1-head attention-only transformer with relative position encodings, operating on a sequence of length T is mathematically defined in Architecture 1.

```
for $n = 1, 2, \cdots, T$ do
    $\boldsymbol{x}_n^{(1)} = \mathtt{Emb}(x_n) \in \mathbb{R}^d.$        (Input embeddings)
for $\ell = 1, 2, \cdots, L,$ do
    for $n = 1, 2, \cdots, T,$ do
        $\widetilde{\boldsymbol{x}}_n^{(\ell)} = \sum_{i\in[n]} \mathrm{att}_{n,i}^{(\ell)} \cdot \boldsymbol{W}_V^{(\ell)} \left(\boldsymbol{x}_i^{(\ell)} + \boldsymbol{p}_{n-i}^{(\ell),V}\right) \in \mathbb{R}^d,$        (Attention)
        $\boldsymbol{x}_n^{(\ell+1)} = \boldsymbol{x}_n^{(\ell)} + \widetilde{\boldsymbol{x}}_n^{(\ell)},$        (Residual)

$\mathrm{logit}_T = \boldsymbol{A}\boldsymbol{x}_T^{(L+1)} + \boldsymbol{b} \quad \in \mathbb{R}^S,$        (Linear)
$\Pr_{\boldsymbol{\theta}}(\cdot \mid x_1, \cdots, x_T) = f(\mathrm{logit}_T)$        (Prediction)
```

Architecture 1: Attention-only transformer.

The attention scores in layer ℓ, $\{\mathrm{att}_{n,i}^{(\ell)} : i \leqslant n\}$, are computed as $\mathtt{Softmax}\big(\{\langle \boldsymbol{W}_K^{(\ell)}(\boldsymbol{x}_j^{(\ell)} + \boldsymbol{p}_{n-j}^{(\ell),K}), \boldsymbol{W}_Q^{(\ell)}\boldsymbol{x}_j^{(\ell)}\rangle :$

$j \in [n]\}\big)$. The superscript (ℓ) indicates the layer index, and the matrices $\boldsymbol{W}_K^{(\ell)}, \boldsymbol{W}_Q^{(\ell)}, \boldsymbol{W}_V^{(\ell)} \in \mathbb{R}^{d \times d}$ capture the key, query and value matrices in layer ℓ. Note that the attention-only transformer may include a feedforward layer with linear activations, i.e. a linear transformation. For representation purposes, this linear transformation can be combined with the projection matrix in the attention layer, allowing the feedforward layer to be omitted from the model. In the attention layer, we consider relative position encodings (the terms labeled in blue), which translates the key and value vectors depending on the relative position of the embedded symbol.

The extension to H heads is straightforward, where in each transformer layer there are H attention layers in parallel, resulting in $\boldsymbol{y}_n^{(\ell,1)}, \cdots, \boldsymbol{y}_n^{(\ell,H)} \in \mathbb{R}^d$ for each n. These vectors are concatenated and passed through a linear transformation $\boldsymbol{W}_O^{(\ell)} : \mathbb{R}^{dH} \to \mathbb{R}^d$ which is the output of the attention layer. Finally, the output of the model after L layers is passed through a linear layer, which projects the d-dimensional embeddings back into $\mathbb{R}^S$ and the resulting vector is passed through a non-linearity f, usually a softmax, to result in the model's prediction of the next symbol probabilities. The theoretical results in this paper will choose $f = \mathrm{ReLU}(\cdot)$.

3. Understanding the empirical behavior of transformers

The motivation for the present work comes from a series of experimental results, which challenge our current understanding of transformers in the context of learning Markov processes. Several works in the literature (Makkuva et al., 2024; Nichani et al., 2024; Edelman et al., 2024) have studied the ability of transformer models to learn k^{th}-order Markov processes. The experimental results present in the literature suggest that in order for a 2 layer transformer model to be able to learn a randomly sampled Markov process of order k, it is crucial for the number of heads in the first attention layer to scale linearly with the order, k. In particular, the authors of (Edelman et al., 2024) claim that in their experiments, "Single attention headed models could not achieve better performance than bigram (models)" in learning random k^{th}-order Markov processes in-context. Similarly, the authors of (Nichani et al., 2024) study a generalization of learning k^{th}-order Markov processes to learning causal processes on degree k graphs. The theory and experiments pertain to 2-layer transformers with k heads.

In Figure 4, we train 2 and 3-layer transformers with a single head on data drawn from random Markov processes of various orders drawn from a Dirichlet prior. With 2 layers and a single head, we see that the model is able to learn even order-4 Markov processes, and go beyond the simple order-1 processes which were projected to be the limit of its ability to learn. Likewise, with 3 layers, transformers are able to go much further and learn order-8 Markov processes, which was the largest value of k we evaluated on.

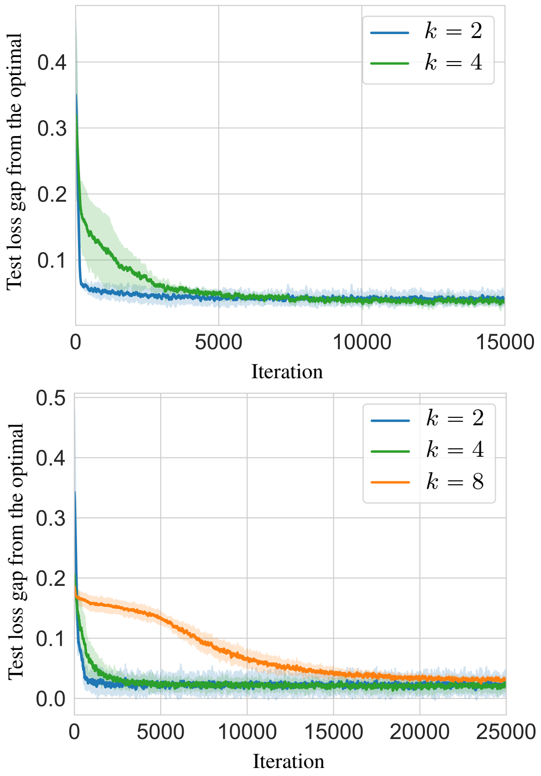

Figure 4: Gap with the optimal test loss for (a) a 2-layer, 1-head transformer model (above), and (b) a 3-layer, 1-head transformer (below), averaged over 3 runs for each k. The models learn the conditional k-gram model for randomly sampled k-th order Markov processes, even for large k.

These results contrast with our current understanding of how induction heads are realized in the parameter space (Edelman et al., 2024; Nichani et al., 2024) - existing constructions which realize these attention patterns require k heads when the number of layers is 2, and it's unclear how to implement them with fewer heads. At a high level, each of the k heads play a critical role - where, loosely speaking, the i^{th}-head looks back i positions in the sequence.

Building up to our main results, in the sequel, we study the simpler case of attention-only transformers where the feedforward layers and layer normalization are removed.

4. Warming up: Attention-only transformers

The study of attention-only transformers trained on Markov processes has garnered some attention in the prior literature. Notably, the authors of (Edelman et al., 2024) study 2-layer 1-head attention-only transformers trained on data drawn from 1$^{\text{st}}$-order Markov processes whose parameters are drawn from a Dirichlet prior. The model is observed to learn a very specific behavior, known as an "induction head" (Olsson et al., 2022), which in this setting is able to represent the conditional 1-gram (Eq. (1)).

The induction head mechanism is composed of two layers where the first layer learns the attention pattern $\text{att}^{(1)}_{n,i} = \mathbb{I}(i = n-1)$, thereby allowing the model to capture information about the symbol at position $n-1$ in the embedding vector at time n. In the second layer, the attention layer picks out those indices n where $x_{n-1} = x_T$, the final symbol in the sequence. At these positions, since $x_{n-1} = x_T$, one would expect that the next symbol x_n is a good predictor of x_{T+1}, and the model uses this information to predict the next symbol x_{T+1} according to its conditional empirical estimate, $\widehat{\Pr}_1(x_{T+1}|x_1, \cdots, x_T)$, i.e. the conditional 1-gram model.

Theorem 4.1. *The conditional* 1*-gram model can be represented by a* 2*-layer and* 1*-head attention-only transformer with embedding dimension* $d = 3S + 2$.

Although a version of this result is also proved in (Edelman et al., 2024), we include a proof in Appendix A for completeness.

Remark 4.2. In Theorem 4.1 and other results to follow, we de-emphasize the role of the bit-precision to which the transformer is implemented. That said, note that when the constructions in Theorems 4.1, 4.4 and 4.5 are implemented to $O(\log(T))$ bits of precision, the representation results are realized up to an additive $O(1/T)$ error.

The ideas in Theorem 4.1 readily extend to representing the conditional k-gram model, by instead using k heads in the first layer. The j^{th} head learns the attention pattern $\text{att}^{(1)}_{n,i} = \mathbb{I}(i = n-j)$ and concatenating the outputs of the heads, the model learns to aggregate information about $x_n, \cdots, x_{n-k}$ in the embedding vector at time n. The second layer realizes what is best described as a "k^{th}-order " induction head, where the model learns to pick out those positions n where for every $j \in [k]$, $x_{n-j} = x_{T-j+1}$, i.e. the history of length k at those positions match the final k symbols in the input sequence (see Figure 5). This mechanism is also referred to as a long-prefix induction head in the literature (Goldowsky-Dill et al., 2023).

Definition 4.3 (Higher-order induction head)**.** A 1-head attention layer is said to realize a k^{th}-order induction head if on any sequence $(x_1, \cdots, x_T) \in [S]^T$, for any fixed $n \leqslant T$, as a function of the input sequence, $\text{att}_{n,T}$ is maximized if and only if $x_{n-j} = x_{T-j+1}$ for every $j \in [k]$.

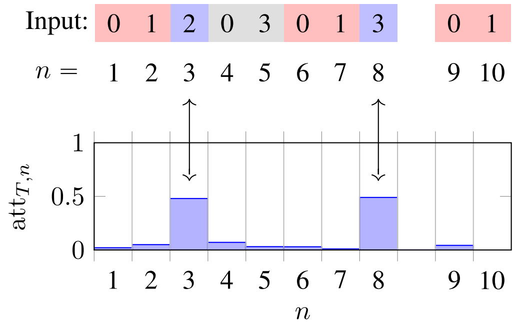

Figure 5: k^{th}-order induction head for $k = 2$. The attention pattern $\text{att}_{T,n}$ is maximized for those values of n at which $x_{T-j+1} = x_{n-j}$ for all $j \in [k]$. These are the positions where the k-length prefix at those positions matches with the last k symbols in the sequence.

The k^{th}-order induction head generalizes the concept of an induction head (Olsson et al., 2022), and is able to keep track of the positions $i \leqslant n$ where there is a perfect occurrence of the final k symbols in the sequence. Such attention patterns are immediately useful in representing the conditional k-gram - increasing the temperature within the softmax of this attention layer results in an attention pattern which converges to the uniform distribution over those positions where the final k symbols $x_{T-k+1}, \cdots, x_T$ are seen previously in the sequence. Loosely, this is what allows the model to "condition" on the last k symbols in the sequence. With k heads, the model can aggregate information from the previous k positions and implement a k^{th}-order induction head, which leads to the following result. A full proof is discussed in Appendix A.1.

Theorem 4.4. *The conditional* k*-gram model can be represented by an attention-only transformer with* 2 *layers,* k *heads and embedding dimension* $d = (k+2)S + k + 1$.

While this result is positive, it suggests that a 2-layer transformer requires approximately k times as many parameters to be able to represent the conditional k-gram model. The first result we prove is that increasing the depth of the model is exponentially more beneficial, in that a transformer with $O(\log(k))$ depth can estimate in-context k-grams.

Theorem 4.5. *The conditional* k*-gram model can be represented by an attention-only transformer with relative position encodings, with* $L = \lceil \log_2(k+1) \rceil$ *layers and* 1 *head per layer. The embedding dimension is* $\leqslant 2k(S+1) + S$.

With 2 layers and k heads, the transformer aggregates information about each of the previous k positions one step at a time through the k heads. However, with $\Omega(\log(k))$ layers, the same task can be done far more efficiently. In the first attention layer, the model aggregates information about the current and previous position. Namely, using the relative position embeddings, $\boldsymbol{x}^{(2)}_n$ is chosen as a linear combination of $\boldsymbol{x}^{(1)}_n = \texttt{Emb}(x_n)$ and $\boldsymbol{x}^{(1)}_{n-1} = \texttt{Emb}(x_{n-1})$. This allows the embedding at position n to aggregate information about

x_n and x_{n-1}. In the same vein, in the second attention layer, the model aggregates information from $\boldsymbol{x}_n^{(2)}$ and $\boldsymbol{x}_{n-2}^{(2)}$ in $\boldsymbol{x}_n^{(3)}$; the former has information about x_n and x_{n-1}, and the latter has information about x_{n-2} and x_{n-3}. This expands the "window" of x_i's on which $\boldsymbol{x}_n$ depends on to size 4. In the ℓ^{th} layer, the model aggregates information from $\boldsymbol{x}_n^{(\ell)}$ and $\boldsymbol{x}_{n-2^\ell}^{(\ell)}$ which allows $\boldsymbol{x}_n^{(\ell+1)}$ to effectively depend on the x_i's in a window of size $2^{\ell+1}$ starting at position n, namely $x_n, \cdots, x_{n-2^{\ell+1}+1}$. In the final layer, the embedding at position i, $\boldsymbol{x}_i^{(L)}$ for $L = \lceil \log_2(k+1) \rceil$ depends on $x_n, x_{n-1}, \cdots, x_{n-k}$. In the last layer, the model can realize the dot-product $\left\langle \boldsymbol{W}_K^{(L)} \boldsymbol{x}_n^{(L)}, \boldsymbol{W}_Q^{(L)} \boldsymbol{x}_T^{(L)} \right\rangle = \sum_{j=1}^k \mathbb{I}(x_{n-j} = x_{T-j+1})$ by choosing the key and query vectors appropriately. By increasing the temperature in the attention softmax, the attention pattern realized is the uniform distribution on values of n such that $x_{n-j} = x_{T-j+1}$ for every $j \in [k]$, i.e., a k^{th}-order induction head. The full proof of this result is provided in Appendix B.

While this is a promising step toward understanding the behavior transformers exhibit in Figure 4, showing that depth plays an important role in their ability to represent conditional k-gram models, the picture is still not complete. The experimental results in Section 3 do not preclude the possibility that a transformer might not even require logarithmic depth to be able to learn k^{th}-order Markov processes approximately. In the next section, we will study constant-depth transformers and establish a rather surprising positive result about the representation power of this class of models in capturing conditional k-grams.

5. Understanding the role of non-linearity: Constant-depth constructions

In the previous section, we saw how the transformer uses the power of depth to learn conditional k-grams far more efficiently. In particular, every additional attention layer effectively doubles the window of positions $i = n-1, n-2, \cdots$ which the model has access to information about at the current time n. By composing $L = \Omega(\log(k))$ attention layers, the model is able to collect enough information within the output embedding $\boldsymbol{x}_n^{(L+1)}$ to be able to realize a k^{th}-order induction head in the next layer. In this section, we prove that adding non-linearity to the architecture, in the form of layer normalization, can significantly change the mechanism in which the transformer realizes this k^{th}-order head. In particular, there are constant depth architectures which allow a k^{th}-order induction head to be realized, surpassing the logarithmic depth attention-only constructions.

Modification to the standard transformer architecture. To simplify the proof of our main result, we will consider a subtle modification to the standard transformer architecture,

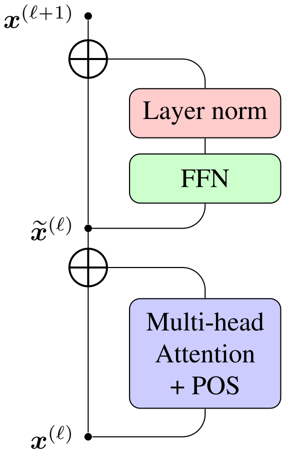
$\boldsymbol{x}^{(\ell+1)}$
Layer norm
FFN
$\tilde{\boldsymbol{x}}^{(\ell)}$
Multi-head Attention + POS
$\boldsymbol{x}^{(\ell)}$

Figure 6: Rearranged transformer layer with layer normalization and FFN.

which is presented in Architecture 2 and Figure 6. We will remove the first layer norm prior to the multi-head attention and move the second layer norm to after the feed-forward network. It is important to note that Theorem 5.1 holds even for the architecture presented in Figure 3, which is the architecture we evaluate empirically. The modification we present in Figure 6 allows the construction to be simpler and makes it much easier to convey the key intuition. The main difference compared to the attention-only design presented in Architecture 1 is the addition of layer normalization and a feedforward layer in the for-loop over $n \in [T]$ for each transformer layer ℓ. The differences between Architectures 2 and 1 are emphasized in blue.

$$\tilde{\boldsymbol{x}}_n^{(\ell)} = \boldsymbol{x}_n^{(\ell)} + \sum_{i \in [n]} \text{att}_{n,i}^{(\ell)} \cdot \boldsymbol{W}_V^{(\ell)} \left(\boldsymbol{x}_i^{(\ell)} + \boldsymbol{p}_{n-i}^{(\ell),V} \right), \quad \text{(Attention + Residual}_1\text{)}$$

$$\boldsymbol{y}_n^{(\ell)} = \boldsymbol{W}_2^{(\ell)} \, \text{ReLU} \left(\boldsymbol{W}_1^{(\ell)} \tilde{\boldsymbol{x}}_n^{(\ell)} \right) \in \mathbb{R}^d, \quad \text{(FFN)}$$

$$\boldsymbol{l}_n^{(\ell)} = \frac{\boldsymbol{y}_n^{(\ell)} - \mu \boldsymbol{1}_{d \times 1}}{\sigma} \in \mathbb{R}^d, \quad \text{(LN)}$$

$$\boldsymbol{x}_n^{(\ell+1)} = \boldsymbol{l}_n^{(\ell)} + \tilde{\boldsymbol{x}}_n^{(\ell)} \in \mathbb{R}^d, \quad \text{(Residual}_2\text{)}$$

Architecture 2: Modified transformer architecture. The computations above are carried out for each $n \in [T]$ in each layer $\ell \in [L]$. In the layer normalization step (LN), the feature mean μ is defined as, $\mathbb{E}_{i \sim \text{Unif}([d])} \left[\langle e_i^d \boldsymbol{y}_n^{(\ell)} \rangle \right]$ and the feature variance $\sigma^2 = \mathbb{E}_{i \sim \text{Unif}([d])} \left[\langle e_i^d, \boldsymbol{y}_n^{(\ell)} \rangle^2 \right] - \mu^2$.

Theorem 5.1. *Conditional k-grams can be represented by a transformer with 3 layers, 1 head per layer, relative position encodings and layer normalization. The embedding dimension is $O(S)$.*

Remark 5.2. Although the proof stated does not bound the approximation error arising from a finite bound on the bit

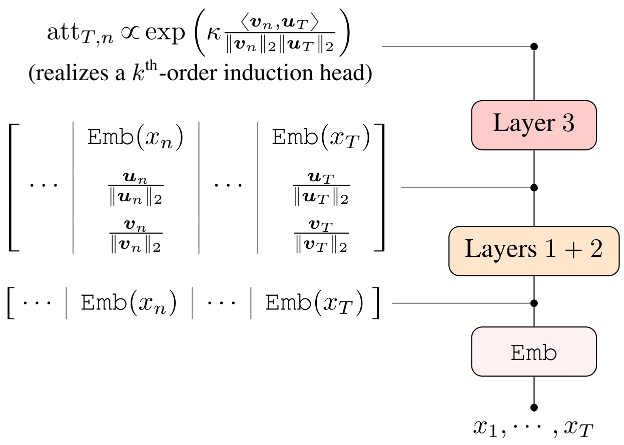
$\mathrm{att}_{T,n} \propto \exp\left(\kappa \frac{\langle \boldsymbol{v}_n, \boldsymbol{u}_T \rangle}{\|\boldsymbol{v}_n\|_2 \|\boldsymbol{u}_T\|_2}\right)$
(realizes a k^{th}-order induction head)
Layer 3
$\left[\cdots \,\middle|\, \begin{matrix} \mathtt{Emb}(x_n) \\ \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} \\ \frac{\boldsymbol{v}_n}{\|\boldsymbol{v}_n\|_2} \end{matrix} \,\middle|\, \cdots \,\middle|\, \begin{matrix} \mathtt{Emb}(x_T) \\ \frac{\boldsymbol{u}_T}{\|\boldsymbol{u}_T\|_2} \\ \frac{\boldsymbol{v}_T}{\|\boldsymbol{v}_T\|_2} \end{matrix} \right]$
Layers $1+2$
$[\cdots \mid \mathtt{Emb}(x_n) \mid \cdots \mid \mathtt{Emb}(x_T)]$
Emb
$x_1, \cdots, x_T$

Figure 7: *Disassembling the constant-depth construction.* The first two layers are critical in the model's ability to capture information from the previous k positions. Layer normalization plays a critical role in the 3rd layer which realizes a k^{th}-order induction head.

precision of the transformer, in theory, it should suffice to have $\Omega(\log(T) + k)$ bits per parameter for the statement of Theorem 5.1 to go through with an $O(1/T)$ additive approximation error. The main point is that none of the weights of the model exceed $\exp(k)$ and with $\log(T)$ additional bits per parameter, the approximation error scales as $O(1/T)$.

5.1. Proof sketch

In the attention-only transformer with 2 layers and k heads, the model is able to keep track of where the final k symbols in the sequence appeared previously (i.e., a k^{th}-order induction head) by, loosely, using each head to keep track of the occurrences of one of the final k symbols. On the other hand, with the benefit of more depth, with $L = \Omega(\log(k))$ layers, the model is able to collect enough information within the output embedding $\boldsymbol{x}_n^{(L+1)}$ to be able to realize the same behavior. However, neither of these constructions scale down to the case when the depth and number of heads of the transformer are both constants independent of k. We provide a brief intuition for the construction below.

Recall that a k^{th}-order induction head keeps track of the indices i such that $\forall j \in [k],\ x_{i-j} = x_{n-j+1}$. Defining $\boldsymbol{z}_i \triangleq \sum_{j=1}^{k} 2^j e_{x_{i-j+1}}$, notice that the condition $\{\forall j \in [k],\ x_{i-j} = x_{n-j+1}\}$ can equivalently be captured by writing $\{\boldsymbol{z}_{i-1} = \boldsymbol{z}_n\}$. This true because of the fact that the binary representation of any integer is unique. Furthermore, these vectors, up to scaling, can be realized by softmax attention (namely, $\mathrm{att}_{n,n-i} \propto 2^i$ for $1 \leqslant i \leqslant k$).

With this step, finding occurrences of the last k symbols in the input sequence boils down to realizing an attention pattern in the second layer, $\mathrm{att}_{n,i}^{(2)}$, which is maximized whenever $\boldsymbol{z}_{i-1} = \boldsymbol{z}_n$. While dot-product attention naively encourages those values of i for which $\boldsymbol{z}_{i-1}$ and $\boldsymbol{z}_n$ are "similar" to each other, a qualitative statement is lacking. In general, it will turn out to that a different measure of similarity is necessary within the softmax to be able to encourage those values of i for which these vectors match. This is where the role of layer-normalization comes in.

Instead of the usual dot-product, suppose the attention mechanism in the second layer was,

$$\mathrm{att}_{n,i}^{(2)} \propto \exp\left(-\kappa \left\| \frac{\boldsymbol{z}_{i-1}}{\|\boldsymbol{z}_{i-1}\|_2} - \frac{\boldsymbol{z}_n}{\|\boldsymbol{z}_n\|_2} \right\|_2^2 \right), \qquad (2)$$

where κ is the temperature parameter. Then, as the temperature κ grows, the attention pattern essentially focuses on those values of i for which $\boldsymbol{z}_i/\|\boldsymbol{z}_{i-1}\|_2 = \boldsymbol{z}_n/\|\boldsymbol{z}_n\|_2$. With this attention pattern, we are thus very close to the statement we wanted to check, $(\boldsymbol{z}_{i-1} \overset{?}{=} \boldsymbol{z}_n)$. As it turns out, for the special structure in the $\boldsymbol{z}_i$'s considered (dyadic sums of one-hot vectors), we may write down,

$$\boldsymbol{z}_{i-1} = \boldsymbol{z}_n \iff \boldsymbol{z}_{i-1}/\|\boldsymbol{z}_{i-1}\|_2 = \boldsymbol{z}_n/\|\boldsymbol{z}_n\|_2.$$

A quantifiable equivalence is provided in Lemma C.1.

Realizing L_2-norm attention (eq. (2)). Observe the equivalence,

$$\left\langle \frac{\boldsymbol{z}_{i-1}}{\|\boldsymbol{z}_{i-1}\|_2}, \frac{\boldsymbol{z}_n}{\|\boldsymbol{z}_n\|_2} \right\rangle = 1 - \frac{1}{2} \left\| \frac{\boldsymbol{z}_{i-1}}{\|\boldsymbol{z}_{i-1}\|_2} - \frac{\boldsymbol{z}_n}{\|\boldsymbol{z}_n\|_2} \right\|_2^2 \qquad (3)$$

Taking a softmax on both sides, notice that the RHS (up to an additive constant) is the L_2-norm based attention, while the LHS is the usual dot-product attention between $\boldsymbol{z}_{i-1}/\|\boldsymbol{z}_{i-1}\|_2$ and $\boldsymbol{z}_n/\|\boldsymbol{z}_n\|_2$. Thus on unit-normalized vectors, the L_2-norm attention and scaled-dot product attention are nothing but the same.

While the first layer of the transformer computes the $\boldsymbol{z}_i$'s by a weighted summation, layer normalization fills in the last missing piece of the puzzle which is to normalize them to unit norm. This is a consequence of defining the embedding vectors appropriately, in such a way that the feature variance evaluates to $\|\boldsymbol{z}_i\|_2^2$.

From this step, realizing the actual conditional k-gram model follows readily. In particular, as the temperature κ in the attention grows, the attention pattern zooms in on indices $i \in \mathcal{I}_n \triangleq \{k+1 \leqslant i \leqslant n : \forall j \in [k], x_{i-j} = x_{n-j+1}\}$ in the last layer. The value vectors at this step are the one-hot encoding of x_i; putting everything together, the logits realized by the transformer are,

$$\mathrm{logit}_T(x_{T+1}) = \frac{1}{|\mathcal{I}_n|} \sum_{i \in \mathcal{I}_n} \mathbb{I}(x_i = x_T), \qquad (4)$$

which is the conditional k-gram model (eq. (1)).

While the transformer construction described above only requires two layers, the actual construction we propose differs slightly and has an additional layer. The first two layers of the transformer respectively compute $\boldsymbol{z}_i$ and $\boldsymbol{z}_{i-1}$ which are added to the embedding vector at time i. This is important because we need to test whether $\boldsymbol{z}_{i-1} \overset{?}{=} \boldsymbol{z}_n$ and not whether $\boldsymbol{z}_i \overset{?}{=} \boldsymbol{z}_n$ or $\boldsymbol{z}_{i-1} \overset{?}{=} \boldsymbol{z}_n$.

Summary. The construction can be summarized as follows: the first layer computes $\boldsymbol{z}_n = \sum_{j=1}^k 2^{j-1} \cdot e_{x_{n-j}}$ by choosing appropriate value vectors and relative position embeddings to realize the attention pattern $\mathrm{attn}_{n,n-i} \propto 2^i \mathbb{I}(1 \leqslant i \leqslant k)$. The layer normalization that follows subsequently can be made to carry out L_2 normalization, by a simple trick which we discuss in Appendix C.1, resulting in $\boldsymbol{z}_n / \|\boldsymbol{z}_n\|_2$ to be appended to the embedding at time n. Using a very similar construction, layer 2 computes $\boldsymbol{z}_{n-1}/\|\boldsymbol{z}_{n-1}\|_2$, which is added to the embedding at time n. Finally, in the last layer, the dot-product $\left\langle \frac{\boldsymbol{z}_{i-1}}{\|\boldsymbol{z}_{i-1}\|_2}, \frac{\boldsymbol{z}_n}{\|\boldsymbol{z}_n\|_2} \right\rangle$ defines the attention score, and as the temperature κ grows, the pattern converges to $\mathrm{Unif}(\mathcal{I}_n)$. Choosing the value vectors in this layer appropriately results in eq. (4).

6. Lower bounds on transformer size

Having established positive results on the representation power of the standard transformer model in the previous section, we ask what are the limits of how shallow the model can be made to be able to capture conditional k-grams. The first result we establish in this vein is a lower bound against 1 layer transformers showing that their expressive power is too limited unless the embedding dimension or number of heads scale near-linearly in T.

Theorem 6.1. *Consider any* 1*-layer transformer with layer normalization and feedforward layers, where all the coordinates of the embedding vectors and unnormalized attention scores are computed with* p *bits of precision. If the transformer is able to compute the conditional* 3*-gram on inputs drawn from* $\{0, 1, 2\}^T$ *to within an additive error of* $1/3T$*, then* $2pH + dp + 2 \geqslant T/3$.

Choosing the bit precision to be $p = O(\log(T))$, this implies that for transformers with 1 layer, the sum of the number of heads and the embedding dimension must be at least $\Omega(T/\log(T))$, in order to represent conditional 3-grams to within an additive error of $1/3T$.

6.1. Conditional lower bounds on attention-only transformers

While the previous section shows that 1-layer transformers have fairly limited representation power, it is not immediately clear how whether any of these issues are present with transformers with more layers. Indeed, as we discussed in Section 4, an attention-only transformer with $O(\log_2(k))$ layers and 1 head per layer can represent conditional k-grams on its input sequences. With the addition of non-linearities, Theorem 5.1 shows that the model can represent conditional k-grams using just a constant number of layers. In this section, we try to understand the gap between these two results and prove conditional lower bounds on the size of attention-only transformers which do not have non-linearities arising from layer normalization. This sheds some more light on the components of the architecture.

We prove conditional lower bounds under some natural assumptions on the nature of the attention patterns learnt by the transformer. To motivate these assumptions, consider the experiment in Figure 8, where we train an attention-only transformer with 2 layers and 1 head, on data drawn from a random order-1 Markov process. At test-time, we sample sequences from the same process and plot the attention patterns learnt in the first layer of the model. Notice that the attention pattern learnt by the model at layer 1 is largely independent of the input sequences themselves and only depends on the position. These, and additional experiments in Appendix G support the assumption we make on the transformer in the size lower bound.

For higher values of k, we observe that there is some dependency of the learnt attention pattern on the inputs, corresponding to higher variance For higher values of k, we defer the experimental results to the appendix, since it requires plotting the attention across multiple heads.

Assumption 6.2. In an L-layer attention-only transformer with H heads per layer, assume that layers $\ell = 1, 2, \cdots, L-1$ and heads $h \in [H]$ realize an attention pattern where $\mathrm{att}_{n,i}^{(\ell,h)}$ only depends on the positions n and i and on ℓ and h, but not on the input sequence $x_1, \cdots, x_T$.

Rather than proving the size lower bound depending on the transformers ability to represent the conditional k-gram itself, we consider a simplification and assume that the goal of the model is to represent a k^{th}-order induction head (Definition 4.3) in the last layer. Although learning a k^{th}-order induction head is not strictly necessary for the transformer to be able to represent conditional k-grams, note that every construction we have considered so far (cf. Theorems 4.1, 4.4, 4.5 and 5.1) go through this mechanism to realize the conditional k-gram model. Likewise, for other related problems, such as the causal learning task in (Nichani et al., 2024), the causal structure is captured by an extension of the k^{th}-order induction head to general causal graphs.

Theorem 6.3. *Consider an* L*-layer attention-only transformer with* 1 *head per layer and relative position encodings, which satisfies Assumption 6.2. If* $L \leqslant 1 + \log_2(k-2)$*, the attention pattern in layer* L *of the transformer cannot represent a* k^{th}*-order induction head.*

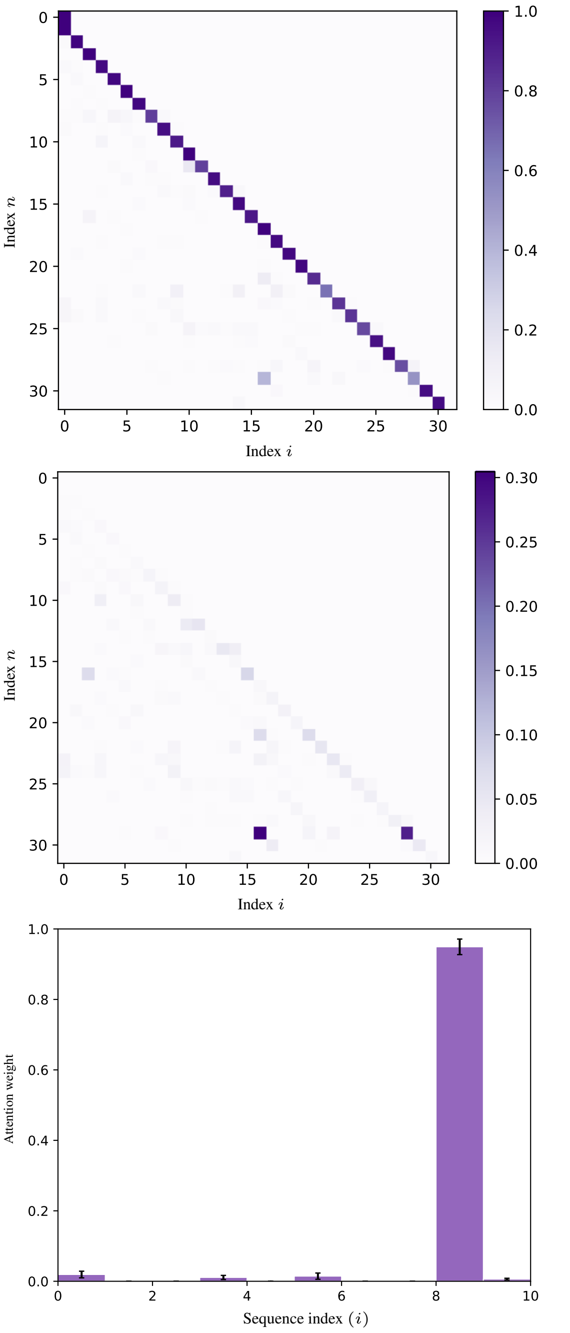

Figure 8: Attention matrix of the first attention layer, for a 2-layer 1-head transformer model trained on an order-1 Markov process, averaged across 100 input sequences of length 128. (a) and (b) plot the mean and standard deviation of the first 32 rows and columns of the attention matrix, while (c) zooms in on the column $n = 10$ and plots the mean attention for this column. (a) and (c) show that for almost all indices n, the attention layer focuses only on the previous symbol x_{n-1}. (b) also shows that the attention pattern is largely independent of the actual sequence $\boldsymbol{x}$ considered, thereby providing evidence toward Assumption 6.2.

While this lower bound is not unconditional, meaning that it does not directly imply that the transformer cannot represent conditional k-grams, it is important to understand the interpretation of this result: attention-only transformers which somehow break through this barrier need to use a significantly different mechanism to realize the conditional k-gram model. It is an interesting question to see if the size lower bound on representing k^{th}-order induction heads also necessarily implies a lower bound on representing the conditional k-gram model under the same assumptions. We also prove a size lower bound on 2-layer transformer models under Assumption 6.2.

Theorem 6.4. *Consider an 2-layer attention-only transformer with H heads in the first layer and relative position encodings, and assume that Assumption 6.2 is satisfied. If $H \leqslant k - 3$, the attention pattern in the 2^{nd} layer cannot represent a k^{th}-order induction head.*

This result implies that under Assumption 6.2, attention-only transformers indeed require the number of heads to scale linearly in k to be able to represent a k^{th}-order induction head. In particular, under Assumption 6.2 this implies that a 2-layer attention-only transformer with 1 head cannot realize a k^{th}-order induction head for any $k \geqslant 4$. Likewise, under the same assumption, a 3-layer attention-only transformer with 1 head cannot realize a k^{th}-order induction head for any $k \geqslant 6$. These results give more weight to the experiment in Figure 4 where we observe that a 2-layer transformer learns a k^{th}-order Markov process for $k = 4$ and a 3-layer transformer learns a k^{th}-order Markov process for $k = 8$, and show that non-linearities in the architecture allow the transformer to break past the size barriers in Theorems 6.3 and 6.4.

Both of these theorems follow as a consequence of a more general size lower bound on attention-only transformers with H_ℓ heads in layer $\ell \in [L]$ which are able to represent a k^{th}-order induction head under Assumption 6.2. This general result also shows the utility in having deeper transformers, relative to wider (i.e., more number of heads) ones, in the context of learning k^{th}-order Markov processes.

Theorem 6.5. *Consider an L-layer transformer with h_ℓ heads in layer L. Assuming the transformer satisfies Assumption 6.2, if $\prod_{\ell=1}^{L-1}(H_\ell + 1) \leqslant k - 2$, the attention pattern in layer L cannot represent a k^{th}-order induction head.*

7. Conclusion

We observe empirically that 2 and 3 layer transformers are able to learn k^{th}-order Markov chains for much higher values of k than previously anticipated. We show there are $O(\log(k))$-layer constructions of attention-only transformers which are able to learn the conditional k-gram model, which is the in-context MLE of the Markov model. With non-linearities in the model, we show that a 3-layer 1-head transformer is capable of representing the same. We show that 1-layer transformers cannot represent conditional k-grams for any $k \geqslant 3$ unless the number of heads or embedding dimension scale almost linearly in T. We also prove a conditional lower bound showing attention-only transformers need $\Omega(\log(k))$ layers to represent k^{th}-order induction heads, under an assumption on the realized attention patterns. Under the same assumptions, we show that a 2-layer transformer needs $\Omega(k)$ heads in the first layer to realize a k^{th}-order induction head in the second layer. Our work focuses on the representational aspects of transformers; understanding the learning dynamics of gradient descent on these problems is an important next question.

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models, 2023.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection, 2023.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the ability and limitations of transformers to recognize formal languages, 2020.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. In Thirty-seventh Conference on Neural Information Processing Systems, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

Benjamin L. Edelman, Ezra Edelman, Surbhi Goel, Eran Malach, and Nikolaos Tsilivis. The evolution of statistical induction heads: In-context learning markov chains, 2024.

Angeliki Giannou, Shashank Rajput, Jy-Yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In Proceedings of the 40th International Conference on Machine Learning, pages 11398–11442, 23–29 Jul 2023.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching. arXiv preprint arXiv:2304.05969, 2023.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021.

Jesse Hoogland, George Wang, Matthew Farrugia-Roberts, Liam Carroll, Susan Wei, and Daniel Murfet. The developmental landscape of in-context learning. arXiv preprint arXiv:2402.02364, 2024.

Hongkang Li, Meng Wang, Sijia Liu, and Pin-Yu Chen. A theoretical understanding of shallow vision transformers: Learning, generalization, and sample complexity.

In The Eleventh International Conference on Learning Representations, 2023.

Ziqian Lin and Kangwook Lee. Dual operating modes of in-context learning, 2024.

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata, 2023.

Ashok Vardhan Makkuva, Marco Bondaschi, Adway Girish, Alliot Nagle, Martin Jaggi, Hyeji Kim, and Michael Gastpar. Attention with markov: A framework for principled analysis of transformers via markov chains. arXiv preprint arXiv:2402.04161, 2024.

Eshaan Nichani, Alex Damian, and Jason D. Lee. How transformers learn causal structure with gradient descent, 2024.

J. R. Norris. Markov Chains. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022.

Samet Oymak, Ankit Singh Rawat, Mahdi Soltanolkotabi, and Christos Thrampoulidis. On the role of attention in prompt-tuning. In Proceedings of the 40th International Conference on Machine Learning, 2023.

Matteo Pagliardini. GPT-2 modular codebase implementation. https://github.com/epfml/llm-baselines, 2023.

Jorge Pérez, Pablo Barceló, and Javier Marinkovic. Attention is Turing-complete. Journal of Machine Learning Research, 22(75):1–35, 2021.

Nived Rajaraman, Jiantao Jiao, and Kannan Ramchandran. Toward a theory of tokenization in llms, 2024.

Clayton Sanford, Daniel J Hsu, and Matus Telgarsky. Representational strengths and limitations of transformers. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, Advances in Neural Information Processing Systems, volume 36, pages 36677–36707. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/73bf692447f174984f30499ec9b20e04-Paper-Conference.pdf.

Clayton Sanford, Daniel Hsu, and Matus Telgarsky. Transformers, parallel computation, and logarithmic depth, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, pages 5998–6008, 2017.

Colin Wei, Yining Chen, and Tengyu Ma. Statistically meaningful approximation: a case study on approximating Turing machines with transformers. In Advances in Neural Information Processing Systems, volume 35, pages 12071–12083, 2022a.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems, 35:24824–24837, 2022b.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. In International Conference on Machine Learning, pages 11080–11090, 2021.

Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In Proceedings of the eleventh annual ACM symposium on Theory of computing, pages 209–213, 1979.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In International Conference on Learning Representations, 2020.

Appendix

Notation. The notation $e_i^{d'} \in \mathbb{R}^{d'}$ refers to the one-hot encoding of i in d' dimensions. In other words it is the i^{th} standard basis vector in d' dimensions. The notation $\texttt{Blkdiag}(\{A_1, A_1, \cdots, A_m\})$ refers to the block diagonal matrix with i^{th} block as A_i.

A. Proof of Theorem 4.1

We will first prove Theorem 4.1. In the first layer, choose the embeddings as,

$$\boldsymbol{x}_n^{(1)} = \texttt{Emb}(x_n) = \kappa \begin{bmatrix} \mathbf{1}_{1\times 2} & e^S_{x_n} & \mathbf{0}_{1\times 2S} \end{bmatrix}^T \in \mathbb{R}^d. \tag{5}$$

for a constant $\kappa > 0$ to be chosen later and $d = 2S + 2$. The relative position encodings will essentially be supported on the first two coordinates, the middle S coordinates are a one-hot encoding of the symbol x_n and the last $2S$ coordinates are 0. The relative position encodings in the first layer are chosen to be $\boldsymbol{p}_{n-i}^{(1),K} = \kappa\left(-1 + \mathbb{I}(n-i=1\})\right) e_1^d \in \mathbb{R}^d$ and $\boldsymbol{p}_{n-i}^{(1),V} = \mathbf{0} \in \mathbb{R}^d$. Choose $\boldsymbol{W}_K^{(1)}$ and $\boldsymbol{W}_Q^{(1)}$ to be $e_1^d(e_1^d)^T \in \mathbb{R}^{d\times d}$. With this choice,

$$\left\langle \boldsymbol{W}_K^{(1)}\left(\boldsymbol{x}_i^{(1)} + \boldsymbol{p}_{n-i}^{(1),K}\right), \boldsymbol{W}_Q^{(1)}\boldsymbol{x}_n^{(1)} \right\rangle = \kappa\mathbb{I}(n-i=1) \tag{6}$$

As $\kappa \to \infty$, the attention pattern (which takes the softmax over of these inner products over $i \in [n]$) computes,

$$\text{att}_{n,i}^{(1)} = \mathbb{I}(i = n-1) \tag{7}$$

for any $n > 1$. Choose the value matrix as,

$$\boldsymbol{W}_V^{(1)} = \begin{bmatrix} \mathbf{0}_{(2+S)\times 2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{S\times S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{d\times d} \tag{8}$$

And with this choice and the residual connection, we get,

$$\boldsymbol{x}_n^{(2)} = \kappa \begin{bmatrix} \mathbf{1}_{1\times 2} & e^S_{x_n} & e^S_{x_{n-1}} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^d \tag{9}$$

which serves as the input to the 2nd transformer layer.

Layer 2. In layer 2, the relative position encodings $\boldsymbol{p}_{n-i}^{K,(2)}$ and $\boldsymbol{p}_{n-i}^{V,(2)}$ are all set as 0. The key matrix picks out the $e^S_{x_n}$ block out of $\boldsymbol{x}_n^{(2)}$ and the query vector picks out the $e^S_{x_{i-1}}$ block out of $\boldsymbol{x}_{i-1}^{(2)}$. In particular, these matrices are chosen so that,

$$\begin{aligned} \boldsymbol{W}_K^{(2)}\boldsymbol{x}_i^{(2)} &= \kappa \begin{bmatrix} \mathbf{1}_{1\times 2} & e^S_{x_{i-1}} & \mathbf{0} \end{bmatrix}^T \in \mathbb{R}^d, \\ \boldsymbol{W}_Q^{(2)}\boldsymbol{x}_n^{(2)} &= \kappa \begin{bmatrix} \mathbf{1}_{1\times 2} & e^S_{x_n} & \mathbf{0} \end{bmatrix}^T \in \mathbb{R}^d \end{aligned} \tag{10}$$

Taking the inner product of these vectors, and taking $\kappa \to \infty$, observe that the attention pattern concentrates on the uniform distribution over all coordinates i such that $x_{i-1} = x_n$. More formally, the attention pattern for any $n > 1$ is,

$$\text{att}_{n,i}^{(2)} = \frac{\mathbb{I}(x_{i-1} = x_n)}{\sum_{i=2}^n \mathbb{I}(x_{i-1} = x_n)}, \tag{11}$$

assuming $\sum_{i=2}^n \mathbb{I}(x_{i-1} = x_n) > 0$. Having realized this attention pattern, may choose the value and subsequent linear layer appropriately. The value matrix simply picks out the $e^S_{x_i}$ block from $\boldsymbol{x}_i^{(2)}$ and places it into the last S coordinates of $\boldsymbol{x}_i^{(3)}$, and the linear layer simply extracts this block and outputs it (after scaling down by a factor of κ), realizing the logits,

$$\text{logit}_n = \frac{1}{\sum_{i=2}^n \mathbb{I}(x_{i-1} = x_n)} \sum_{i=2}^n \mathbb{I}(x_{i-1} = x_n) \cdot e^S_{x_i}. \tag{12}$$

if $\sum_{i=2}^{n} \mathbb{I}(x_{i-1} = x_n) > 0$. In particular, under the same condition,

$$\mathrm{logit}_T(x_{T+1}) = \frac{\sum_{n=2}^{T} \mathbb{I}(x_n = x_{T+1}, x_{n-1} = x_T)}{\sum_{i=2}^{n} \mathbb{I}(x_{n-1} = x_T)} \tag{13}$$

assuming $\sum_{i=2}^{n} \mathbb{I}(x_{n-1} = x_T)$, which is the conditional 1-gram model.

A.1. Extension to k-heads: Proof of Theorem 4.4

In the first layer, the embeddings are chosen to be,

$$\boldsymbol{x}_n^{(1)} = \texttt{Emb}(x_n) = \kappa \left[\ \mathbf{0}_{1\times k} \mid 1 \mid e^S_{x_n} \mid \mathbf{0}_{1\times (k+1)S}\ \right]^T \in \mathbb{R}^d \tag{14}$$

With $d = (k+1)(S+1) + S$. The relative position encodings are chosen as $\boldsymbol{p}_i^{K,(1)} = \begin{bmatrix} e_i^k & \mathbf{0} \end{bmatrix}^T$ for $1 \leqslant i \leqslant k$ and $\boldsymbol{p}_i^{K,(1)} = \mathbf{0}$ otherwise. Similarly, $\boldsymbol{p}_i^{V,(1)} = \mathbf{0}$ for every i. The h^{th} head has key and query matrices,

$$\begin{aligned} \boldsymbol{W}_Q^{(1,h)} &= \begin{bmatrix} \mathbf{0}_{1\times k} & 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\ \boldsymbol{W}_K^{(1,h)} &= \begin{bmatrix} \mathbf{0}_{1\times (h-1)} & 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \end{aligned} \tag{15}$$

With these choices, and letting $\kappa \to \infty$, the h^{th} layer computes the attention pattern,

$$\mathrm{att}_{n,i}^{(1,h)} = \mathbb{I}(i = n-h). \tag{16}$$

Choose the corresponding value matrix as,

$$\boldsymbol{W}_V^{(1,h)} = \begin{bmatrix} \mathbf{0}_{(2+hS)\times 2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{S\times S} & \mathbf{0} \end{bmatrix} \tag{17}$$

choosing the projection matrix appropriately, the output of the transformer after the first residual connection is,

$$\boldsymbol{x}_n^{(2)} = \kappa \left[\ \mathbf{0}_{1\times k} \mid 1 \mid e^S_{x_n} \mid \cdots \mid e^S_{x_{n-k}}\ \right]^T. \tag{18}$$

Layer 2. In this layer, the relative position encodings $\boldsymbol{p}_{n-i}^{K,(2)}$ and $\boldsymbol{p}_{n-i}^{V,(2)}$ are all set as 0. The key and query matrices are chosen as,

$$\begin{aligned} \boldsymbol{W}_Q^{(2)} &= \begin{bmatrix} \mathbf{0}_{Sk\times k} & I_{(Sk+1)\times(Sk+1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\ \boldsymbol{W}_K^{(2)} &= \begin{bmatrix} \mathbf{0}_{Sk\times (k+S)} & I_{(Sk+1)\times(Sk+1)} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}. \end{aligned} \tag{19}$$

With this choices, we have that,

$$\left\langle \boldsymbol{W}_K^{(2)} \boldsymbol{x}_i^{(2)}, \boldsymbol{W}_Q^{(2)} \boldsymbol{x}_n^{(2)} \right\rangle = \kappa \sum_{j=1}^{k} \mathbb{I}(x_{i-j} = x_{n-j+1}). \tag{20}$$

Taking $\kappa \to \infty$, observe that the attention pattern concentrates on the uniform distribution over all coordinates i such that $x_{i-j} = x_{n-j+1}$ for all $j \in [k]$. More formally, if $\sum_{i=2}^{n} \mathbb{I}(x_{i-1} = x_n) > 0$, the attention pattern for any $n > 1$ is,

$$\mathrm{att}_{n,i}^{(2)} = \frac{\mathbb{I}(\forall j \in [k],\ x_{i-j} = x_{n-j+1})}{\sum_{i=k+1}^{n} \mathbb{I}(\forall j \in [k],\ x_{i-j} = x_{n-j+1})}. \tag{21}$$

The value matrix picks out $e^S_{x_i}$ from the embedding $\boldsymbol{x}_i^{(2)}$ (Equation (18)) and places it in the last S coordinates. The subsequent linear layer picks out the last S coordinates, resulting in the logits,

$$\mathrm{logit}_n = \sum_{i=k+1}^{n} \frac{\mathbb{I}(\forall j \in [k],\ x_{i-j} = x_{n-j+1})}{\sum_{i=k+1}^{n} \mathbb{I}(\forall j \in [k],\ x_{i-j} = x_{n-j+1})} e^S_{x_i}, \tag{22}$$

assuming that $\sum_{i=k+1}^{n} \mathbb{I}(\forall j \in [k],\ x_{i-j} = x_{n-j+1}) > 0$. In particular,

$$\text{logit}_T(x_{T+1}) = \frac{\sum_{n=k+1}^{T} \mathbb{I}(\forall 0 \leqslant j \leqslant k,\ x_{n-j} = x_{T-j+1})}{\sum_{n=k+1}^{T} \mathbb{I}(\forall 1 \leqslant j \leqslant k,\ x_{n-j} = x_{T-j+1})}, \tag{23}$$

assuming $\sum_{n=k+1}^{T} \mathbb{I}(\forall 1 \leqslant j \leqslant k,\ x_{n-j} = x_{T-j+1}) > 0$, i.e., the conditional k-gram model.

B. Proof of Theorem 4.5

Define $k^\star = 2^{\lceil \log_2(k+1) \rceil}$ by rounding $k+1$ up to the nearest power of 2 and $\ell^\star = \log_2(k^\star)$. In the setting of relative position encodings, given the sequence $x_1, \cdots, x_n$, while generating the output of the attention + feedforward layer for the symbol x_n, the embeddings $\boldsymbol{x}_i = \texttt{Emb}(x_n) + \boldsymbol{p}_{n-i}$ are used for $i \in [n]$. In other words, the position encoding vector is taken relative to the end of the sequence, rather than the start of the sequence. Consider the embedding of x as,

$$\boldsymbol{x}_n^{(1)} = \texttt{Emb}(x_n) = \left[\ \mathbf{0}_{1 \times \ell^\star} \ \middle|\ 1 \ \middle|\ e^S_{x_n} \ \middle|\ \mathbf{0}_{1 \times (k^\star - 1)S} \ \middle|\ \mathbf{0}_{1 \times S}\ \right]^T \in \mathbb{R}^{(k^\star+1)S + \ell^\star + 1} \tag{24}$$

where $e_i^{d'} \in \mathbb{R}^S$ is the standard basis vector in d' dimensions. And the relative position encoding for the keys as,

$$\boldsymbol{p}_i^{(1),K} = \begin{cases} \begin{bmatrix} \mathbf{1}_{1 \times \ell^\star} & \mathbf{0} \end{bmatrix}^T, & \text{if } i = 0, \\ \begin{bmatrix} e^{\ell^\star}_{1+\log_2(i)} & \mathbf{0} \end{bmatrix}^T & \text{if } i \in \{1, 2, 4, \cdots, k^\star/2\} \\ \mathbf{0}_{d \times 1} & \text{otherwise.} \end{cases} \tag{25}$$

And for the value vectors, $\boldsymbol{p}_i^V = \mathbf{0}$ for all i.

For the first layer and first head, we will describe the value, key and query matrices. Choose,

$$\begin{aligned} \boldsymbol{W}_K^{(1)} &= \sqrt{\kappa} \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \text{ and,} \\ \boldsymbol{W}_Q^{(1)} &= \sqrt{\kappa} \begin{bmatrix} \mathbf{0}_{1 \times l^\star} & 1 & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} \end{bmatrix}. \end{aligned} \tag{26}$$

Then, observe that for $i \geqslant 1$,

$$\left\langle \boldsymbol{W}_K^{(1)} \big(\boldsymbol{x}_{n-i} + \boldsymbol{p}_i^{(1),K}\big), \boldsymbol{W}_Q^{(1)} \boldsymbol{x}_n \right\rangle = \kappa \mathbb{I}(i = 1)$$

and for $i = 0$,

$$\left\langle \boldsymbol{W}_K^{(1)} \big(\boldsymbol{x}_n + \boldsymbol{p}_0^{(1),K}\big), \boldsymbol{W}_Q^{(1)} \boldsymbol{x}_n \right\rangle = \kappa$$

In particular, letting $\kappa \to \infty$, the attention pattern is,

$$\text{att}^{(1)}_{n,n-i} = \frac{1}{2}\mathbb{I}(i = 0) + \frac{1}{2}\mathbb{I}(i = 1). \tag{27}$$

Choose the value matrix as,

$$\boldsymbol{W}_V^{(1)} = \begin{bmatrix} \mathbf{0}_{(\ell^\star + S) \times \ell^\star} & \mathbf{0} \\ \mathbf{0} & 2I \end{bmatrix}$$

together with the residual connection, we get,

$$\boldsymbol{x}_n^{(2)} = \left[\ \mathbf{0}_{1 \times \ell^\star} \ \middle|\ 1 \ \middle|\ e^S_{x_n} \ \middle|\ e^S_{x_n} + e^S_{x_{n-1}} \ \middle|\ \mathbf{0}_{1 \times (k^\star - 2)S} \ \middle|\ \mathbf{0}_{1 \times S}\ \right]^T \tag{28}$$

Layer $\ell+1$. By induction, assume that the output of the ℓ^{th} transformer layer is of the form,

$$\boldsymbol{x}_n^{(\ell+1)} = \left[\begin{array}{c|c|c|c|c} \mathbf{0}_{1\times \ell^\star} & 1 & \boldsymbol{v}_n & \mathbf{0}_{1\times (k^\star - 2^\ell)S} & \mathbf{0}_{1\times S} \end{array} \right]^T \tag{29}$$

for some vector $\boldsymbol{v}_n \in \mathbb{R}^{2^\ell S}$. We will show that with appropriately chosen key, query and value vectors in the $(\ell+1)^{\text{th}}$ layer, the output of this layer is,

$$\boldsymbol{x}_n^{(\ell+2)} = \left[\begin{array}{c|c|c|c|c|c} \mathbf{0}_{1\times \ell^\star} & 1 & \boldsymbol{v}_n & \boldsymbol{v}_n + \boldsymbol{v}_{n-2^\ell} & \mathbf{0}_{1\times (k^\star - 2^{\ell+1})S} & \mathbf{0}_{1\times S} \end{array} \right]^T \tag{30}$$

We will consider the same relative position encodings and query matrix in this layer as in the first layer (Equations (25) and (26)). Consider a key matrix of the form,

$$\boldsymbol{W}_K^{(\ell+1)} = \begin{bmatrix} \mathbf{0}_{1\times \ell} & \sqrt{\kappa} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

With this choice, observe that for $i \geqslant 1$,

$$\left\langle \boldsymbol{W}_K^{(\ell+1)} \big(\boldsymbol{x}_{n-i}^{(\ell+1)} + \boldsymbol{p}_i^{(\ell+1),K} \big), \boldsymbol{W}_Q^{(\ell+1)} \boldsymbol{x}_n^{(\ell+1)} \right\rangle = \kappa \cdot \mathbb{I}(i = 2^\ell)$$

and for $i = 0$,

$$\left\langle \boldsymbol{W}_K^{(\ell+1)} \big(\boldsymbol{x}_{n}^{(\ell+1)} + \boldsymbol{p}_0^{(\ell+1),K} \big), \boldsymbol{W}_Q^{(\ell+1)} \boldsymbol{x}_n^{(\ell+1)} \right\rangle = \kappa$$

In particular, letting $\kappa \to \infty$, the attention pattern is,

$$\text{att}_{n,n-i}^{(\ell+1)} = \frac{1}{2}\mathbb{I}(i = 0) + \frac{1}{2}\mathbb{I}(i = 2^\ell). \tag{31}$$

Choosing the value matrix as,

$$\boldsymbol{W}_V^{(\ell+1)} = \begin{bmatrix} \mathbf{0}_{(\ell^\star + 2^\ell S)\times \ell^\star} & \mathbf{0} \\ \mathbf{0} & 2I \end{bmatrix},$$

we get,

$$\boldsymbol{x}_n^{(\ell+2)} = \left[\begin{array}{c|c|c|c|c|c} \mathbf{0}_{1\times \ell^\star} & 1 & \boldsymbol{v}_n & \boldsymbol{v}_n + \boldsymbol{v}_{n-2^\ell} & \mathbf{0}_{1\times (k^\star - 2^{\ell+1})S} & \mathbf{0}_{1\times S} \end{array} \right]^T \tag{32}$$

Final last transformer layer ($\ell = \ell^\star$). The output of the second last transformer layer, indexed $\ell^\star - 1$ is,

$$\begin{aligned} \boldsymbol{z}_n^{(\ell^\star)} \triangleq \boldsymbol{x}_n^{(\ell^\star)} &= \left[\begin{array}{c|c|c|c|c} \mathbf{0}_{1\times \ell^\star} & 1 & \boldsymbol{v}_n^{(\ell^\star-1)} & \boldsymbol{v}_n^{(\ell^\star-1)} + \boldsymbol{v}_{n-2^{\ell^\star-1}}^{(\ell^\star-1)} & \mathbf{0}_{1\times S} \end{array} \right]^T \\ &= \left[\begin{array}{c|c|c|c|c} \mathbf{0}_{1\times \ell^\star} & 1 & \boldsymbol{v}_n^{(\ell^\star-1)} & \boldsymbol{v}_n^{(\ell^\star-1)} + \boldsymbol{v}_{n-\frac{k^\star}{2}}^{(\ell^\star-1)} & \mathbf{0}_{1\times S} \end{array} \right]^T, \end{aligned}$$

which follows by plugging in the definition of $k^\star$. Note that there exists a linear transformation $\boldsymbol{L}^{(\ell^\star)}$ such that,

$$\boldsymbol{z}_n^{(\ell^\star-1)} \triangleq \boldsymbol{L}^{(\ell^\star)} \boldsymbol{x}_n^{(\ell^\star)} = \left[\begin{array}{c|c|c|c|c} \mathbf{0}_{1\times \ell^\star} & 1 & \boldsymbol{v}_n^{(\ell^\star-1)} & \boldsymbol{v}_{n-\frac{k^\star}{2}}^{(\ell^\star-1)} & \mathbf{0}_{1\times S} \end{array} \right]^T$$

This can be further decomposed as,

$$\begin{aligned} &\boldsymbol{z}_n^{(\ell^\star-1)} \\ &= \left[\begin{array}{c|c|c|c|c|c|c} \mathbf{0}_{1\times \ell^\star} & 1 & \boldsymbol{v}_n^{(\ell^\star-2)} & \boldsymbol{v}_n^{(\ell^\star-2)} + \boldsymbol{v}_{n-2^{\ell^\star-2}}^{(\ell^\star-2)} & \boldsymbol{v}_{n-\frac{k^\star}{2}}^{(\ell^\star-2)} & \boldsymbol{v}_{n-\frac{k^\star}{2}}^{(\ell^\star-2)} + \boldsymbol{v}_{n-\frac{k^\star}{2}-2^{\ell^\star-2}}^{(\ell^\star-2)} & \mathbf{0}_{1\times S} \end{array} \right]^T \end{aligned}$$

And yet again there exists a linear transformation $\boldsymbol{L}^{(\ell^\star-1)}$ which transforms this as,

$$\boldsymbol{z}_n^{(\ell^\star-2)} \triangleq \boldsymbol{L}^{(\ell^\star-1)} \boldsymbol{z}_n^{(\ell^\star-1)}$$

$$
\begin{aligned}
&= \left[\; \mathbf{0}_{1\times\ell^\star} \;\middle|\; 1 \;\middle|\; \boldsymbol{v}_n^{(\ell^\star-2)} \;\middle|\; \boldsymbol{v}_{n-2^{\ell^\star-2}}^{(\ell^\star-2)} \;\middle|\; \boldsymbol{v}_{n-\frac{k^\star}{2}-2^{\ell^\star-2}}^{(\ell^\star-2)} \;\middle|\; \boldsymbol{v}_{n-\frac{k^\star}{2}-2^{\ell^\star-2}}^{(\ell^\star-2)} \;\middle|\; \mathbf{0}_{1\times S} \;\right]^T \\
&= \left[\; \mathbf{0}_{1\times\ell^\star} \;\middle|\; 1 \;\middle|\; \boldsymbol{v}_n^{(\ell^\star-2)} \;\middle|\; \boldsymbol{v}_{n-\frac{k^\star}{4}}^{(\ell^\star-2)} \;\middle|\; \boldsymbol{v}_{n-\frac{k^\star}{2}}^{(\ell^\star-2)} \;\middle|\; \boldsymbol{v}_{n-\frac{3k^\star}{4}}^{(\ell^\star-2)} \;\middle|\; \mathbf{0}_{1\times S} \;\right]^T
\end{aligned} \tag{33}
$$

By recursing this argument and composing all the linear transformations, up to a global permutation, we get that,

$$
\begin{aligned}
\prod_{\ell=1}^{\ell^\star} \boldsymbol{L}^{(\ell)} \boldsymbol{x}_n^{(\ell^\star)} &= \left[\; \mathbf{0}_{1\times\ell^\star} \;\middle|\; 1 \;\middle|\; \boldsymbol{v}_n^{(1)} \;\middle|\; \boldsymbol{v}_{n-1}^{(1)} \;\middle|\; \cdots \;\middle|\; \boldsymbol{v}_{n-(k^\star-1)}^{(1)} \;\middle|\; \mathbf{0}_{1\times S} \;\right]^T \\
&= \left[\; \mathbf{0}_{1\times\ell^\star} \;\middle|\; 1 \;\middle|\; e_{x_n}^S \;\middle|\; \cdots \;\middle|\; e_{x_{n-(k^\star-1)}}^S \;\middle|\; \mathbf{0}_{1\times S} \;\right]^T
\end{aligned} \tag{34}
$$

In the final layer, we will right multiply the key, query and value matrices by $\boldsymbol{L}^\star = \prod_{\ell=1}^{\ell^\star} \boldsymbol{L}^{(\ell)}$. The effect can be interpreted as operating the original key, query and value matrices on the embedding vectors in Equation (34). In the final layer, we will set all the position encodings to be $\mathbf{0}$ and consider the key and query matrices,

$$
\begin{aligned}
\boldsymbol{W}_K^{(\ell^\star)} &= \sqrt{\kappa} \begin{bmatrix} \mathbf{0}_{Sk\times(\ell^\star+1+S)} & I_{Sk\times Sk} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\
\boldsymbol{W}_Q^{(\ell^\star)} &= \sqrt{\kappa} \begin{bmatrix} \mathbf{0}_{Sk\times(\ell^\star+1)} & I_{Sk\times Sk} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}
\end{aligned} \tag{35}
$$

Then,

$$
\left\langle \boldsymbol{W}_K^{(\ell^\star)} \boldsymbol{L}^\star \boldsymbol{x}_{n-i}^{(\ell^\star)}, \boldsymbol{W}_Q^{(\ell^\star)} \boldsymbol{L}^\star \boldsymbol{x}_n^{(\ell^\star)} \right\rangle = \kappa \sum_{j=0}^{k-1} \mathbb{I}(x_{n-j} = x_{i-1-j}) \tag{36}
$$

Where we must be careful to note that the input $\boldsymbol{x}_n^{(\ell^\star)}$ contains copies of $e_{x_n}, e_{x_{n-1}}, \cdots, e_{x_{n-k}}$ since $k^\star \geqslant k+1$ by definition.

Letting $\kappa \to \infty$, if there exists i such that $\sum_{j=0}^{k-1} \mathbb{I}(x_{n-j} = x_{i-j-1}) > 0$, for $n \geqslant k$, the attention pattern is,

$$
\mathrm{att}_{n,i}^{(\ell^\star)} = \frac{\mathbb{I}(x_{i-1} = x_n, x_{i-2} = x_{n-1}, \cdots, x_{i-k} = x_{n-k+1})}{\sum_{i=k}^n \mathbb{I}(x_{i-1} = x_n, x_{i-2} = x_{n-1}, \cdots, x_{i-k} = x_{n-k+1})} \tag{37}
$$

Finally, choose,

$$
\boldsymbol{W}_V^{(\ell^\star+2)} = \begin{bmatrix} \mathbf{0}_{(d-S)\times(\ell^\star+1)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0}_{S\times(\ell^\star+1)} & I_{S\times S} & \mathbf{0} \end{bmatrix}, \tag{38}
$$

we get,

$$
\boldsymbol{x}_n^{(\ell^\star+1)} + \sum_{i=k}^n \frac{\mathbb{I}(x_{i-1} = x_n, x_{i-2} = x_{n-1}, \cdots, x_{i-k} = x_{n-k+1})}{\sum_{i=k}^n \mathbb{I}(x_{i-1} = x_n, x_{i-2} = x_{n-1}, \cdots, x_{i-k} = x_{n-k+1})} \begin{bmatrix} \mathbf{0}_{(d-S)\times 1} \\ e_{x_i} \end{bmatrix} \tag{39}
$$

Choosing the subsequent linear layer as,

$$
\boldsymbol{A} = \begin{bmatrix} \mathbf{0}_{S\times(d-S)} & I_{S\times S} \end{bmatrix} \tag{40}
$$

$$
\boldsymbol{b} = \mathbf{0}_{S\times 1} \tag{41}
$$

Results in the output,

$$
\mathrm{logit}_T(x_{T+1}) = \sum_{n=k}^T \frac{\mathbb{I}(x_n = x_{T+1}, x_{n-1} = x_T, x_{n-2} = x_{T-1}, \cdots, x_{n-k} = x_{T-k+1})}{\sum_{n=k}^T \mathbb{I}(x_{n-1} = x_T, x_{n-2} = x_{T-1}, \cdots, x_{n-k} = x_{T-k+1})} \tag{42}
$$

which is precisely the in-context conditional k-gram.

C. Proof of Theorem 5.1

C.1. Modifying the definition of layer normalization

In every layer, we will perform a simple transformation which is to double the hidden dimension d and add a copy of $-\boldsymbol{x}_n^{(\ell)}$ into the last d coordinates. This is possible by modifying the weights of the transformer appropriately as discussed below. A consequence of this transformation is that the feature mean of the $\boldsymbol{x}_n$'s is $\mu_n = 0$, and therefore the standard deviation σ_n simply normalizes by the L_2-norm of the features. In order to avoid having to explicitly state this transformation at each layer, we will simply redefine the layer norm LN to output $\boldsymbol{v}/\|\boldsymbol{v}\|_2$ for the input vector $\boldsymbol{v}$, which is realized on the first d coordinates of the transformed embeddings.

This transformation can be realized automatically by redefining the initial embeddings $\texttt{Emb}(x_n)$, and modifying the weights of the attention and feedforward subnetworks as follows: The input embeddings are changed to $\big[\texttt{Emb}(x_n) \quad -\texttt{Emb}(x_n)\big]^T \in \mathbb{R}^{2d}$. The key and query matrices are chosen to be 0 on the last d coordinates in every layer; the value matrix for $i \geqslant 1$ is transformed to $\texttt{Blkdiag}\big(\{\boldsymbol{W}_V^{(\ell)}, \boldsymbol{W}_V^{(\ell)}\}\big)$, and likewise changing the feedforward layer to the block diagonal matrices $\texttt{Blkdiag}\big(\{\boldsymbol{W}_1^{(\ell)}, \boldsymbol{W}_1^{(\ell)}\}\big)$ and $\texttt{Blkdiag}\big(\{\boldsymbol{W}_2^{(\ell)}, \boldsymbol{W}_2^{(\ell)}\}\big)$. This transformation adds a copy of $-\boldsymbol{x}_n^{(\ell)}$ into the last d coordinates of the corresponding embeddings.

C.2. Notation and supplementary lemmas

For each $i \in [T]$, define,

$$\boldsymbol{v}_i = e_{x_{i-1}} + 3 \cdot e_{x_{i-2}} + \cdots + 3^{k-1} \cdot e_{x_{i-k}} \tag{43}$$

$$\boldsymbol{u}_i = e_{x_i} + 3 \cdot e_{x_{i-1}} + \cdots + 3^{k-1} \cdot e_{x_{i-k+1}} \tag{44}$$

Note that although $\boldsymbol{v}_i = \boldsymbol{u}_{i-1}$, we make the distinction between the two to avoid any confusion in what is stored in the embedding vector at time i and at time $i-1$. Furthermore, define,

$$\mathcal{I}_n = \{k+1 \leqslant i \leqslant n : \forall j \in [k], x_{i-j} = x_{n-j+1}\}. \tag{45}$$

Lemma C.1. *If $i \in \mathcal{I}_n$, $\boldsymbol{z}_i = \boldsymbol{z}_{n-1}$. However, if $i \geqslant k+1$ but $i \notin \mathcal{I}_n$, then, $\left\| \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|_2} - \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} \right\|_2 \geqslant 3^{-k}$.*

Let $j^\star \in \{0, 1, \cdots, k-1\}$ denote the largest index j such that $x_{n-j} \neq x_{i-j-1}$. Consider the coordinates $a = x_{n-j^\star} \in [S]$ and $b = x_{i-j^\star-1} \in [S]$. Then,

$$\langle \boldsymbol{v}_n, e_a \rangle - \langle \boldsymbol{u}_i, e_a \rangle \geqslant 3^j - \sum_{j=0}^{j^\star-1} 3^j = \frac{3^{j^\star}}{2}, \tag{46}$$

$$\langle \boldsymbol{u}_i, e_b \rangle - \langle \boldsymbol{v}_n, e_b \rangle \geqslant \frac{3^{j^\star}}{2} \tag{47}$$

If $\|\boldsymbol{v}_n\|_2 \geqslant \|\boldsymbol{u}_i\|_2$, then,

$$\left\langle \frac{\boldsymbol{u}_i}{\|\boldsymbol{u}_i\|_2}, e_b \right\rangle - \left\langle \frac{\boldsymbol{v}_n}{\|\boldsymbol{v}_n\|_2}, e_b \right\rangle \geqslant \frac{\langle \boldsymbol{u}_i, e_b \rangle - \langle \boldsymbol{v}_n, e_b \rangle}{\max\{\|\boldsymbol{u}_i\|_2, \|\boldsymbol{v}_n\|_2\}} \geqslant \frac{3^{j^\star}}{2 \cdot \frac{3^k}{2}} = 3^{j^\star - k} \tag{48}$$

This uses the fact that $\boldsymbol{u}_i$ and $\boldsymbol{v}_n$ are coordinate-wise non-negative. On the other hand, if $\|\boldsymbol{v}_n\|_2 \leqslant \|\boldsymbol{u}_i\|_2$, using a similar analysis,

$$\left\langle \frac{\boldsymbol{u}_i}{\|\boldsymbol{u}_i\|_2}, e_a \right\rangle - \left\langle \frac{\boldsymbol{v}_n}{\|\boldsymbol{v}_n\|_2}, e_a \right\rangle \geqslant 3^{j^\star - k}. \tag{49}$$

In either case, there is a coordinate (a or b) such that, $\boldsymbol{u}_i/\|\boldsymbol{u}_i\|_2$ and $\boldsymbol{v}_n/\|\boldsymbol{v}_n\|_2$ differ by at least $3^{j^\star - k}$. This implies the lower bound on the L_2 norm of the difference of the vectors.

C.3. Proof of Theorem 5.1

Choose the input embeddings as,

$$\boldsymbol{x}_n^{(1)} = \texttt{Emb}(x_n) = \begin{bmatrix}\mathbf{0}_{1\times 3} & e_x^S & \mathbf{0}_{1\times 5S}\end{bmatrix}^T \in \mathbb{R}^{6S+3} \tag{50}$$

In the first two layers we will use the same relative position embeddings, in particular,

$$\boldsymbol{p}_i^{(1),K} = \boldsymbol{p}_i^{(2),K} = \begin{cases} \sqrt{\log(3)} \cdot \begin{bmatrix}1 & \mathbf{0}\end{bmatrix}^T, & \text{if } i = 0, \\ (i+1)\sqrt{\log(3)} \cdot \begin{bmatrix}0 & 1 & \mathbf{0}\end{bmatrix}^T, & \text{if } i \in \{1, 2, \cdots, k-1\}, , \\ (k+1)\sqrt{\log(3)} \cdot \begin{bmatrix}0 & 0 & 1 & \mathbf{0}\end{bmatrix}^T, & \text{if } i = k. \end{cases} \tag{51}$$

and the value embeddings,

$$\boldsymbol{p}_i^{(1),V} = \boldsymbol{p}_i^{(2),V} = \begin{cases} 3^i \begin{bmatrix}1 & \mathbf{0}\end{bmatrix}^T & \text{for } i \leqslant k \\ \mathbf{0} & i > k. \end{cases} \tag{52}$$

In the final layer, we will drop all position-related information and choose $\boldsymbol{p}_i^{(3),K} = \boldsymbol{p}_i^{(3),V} = \mathbf{0}$ for all i.

Layer 1. Consider the key and query matrices,

$$\begin{aligned} \boldsymbol{W}_K^{(1)} &= \sqrt{\kappa} \cdot \begin{bmatrix} \mathbf{1}_{1\times 2} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \\ \boldsymbol{W}_Q^{(1)} &= \sqrt{\kappa} \cdot \begin{bmatrix} \mathbf{0}_{1\times 3} & \mathbf{1}_{1\times S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \end{aligned} \tag{53}$$

Then, observe that,

$$\left\langle \boldsymbol{W}_K^{(1)} \big(\texttt{Emb}(x_{n-i}) + \boldsymbol{p}_i^{(1),K}\big), \boldsymbol{W}_Q^{(1)} \texttt{Emb}(x_n) \right\rangle = \kappa (i+1) \log(3) \cdot \mathbb{I}(0 \leqslant i \leqslant \min\{n,k\} - 1)$$

Letting $\kappa \to \infty$, this results in the attention pattern,

$$\text{att}_{n,n-i}^{(1)} = \frac{3^i \mathbb{I}(0 \leqslant i \leqslant \min\{n,k\} - 1)}{\sum_{i'=0}^{\min\{n,k\}-1} 3^{i'}} \tag{54}$$

Choose the value matrix as,

$$\boldsymbol{W}_V^{(1)} = \begin{bmatrix} \mathbf{0}_{(S+3)\times 3} & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix}$$

The output of the attention layer (with the residual connection) is,

$$\widetilde{\boldsymbol{x}}_n^{(1)} = \left[\begin{array}{c|c|c|c} \mathbf{0}_{1\times 3} & e_{x_n}^S & \boldsymbol{u}_n & \mathbf{0}_{1\times 3S} \end{array}\right]^T, \text{ where, } \boldsymbol{u}_n = \sum_{i=0}^{\min\{n,k\}-1} \text{att}_{n,n-i}\, e_{x_{n-i}}^S. \tag{55}$$

In the feedforward layer to follow, we will choose,

$$\begin{aligned} \boldsymbol{W}_1^{(1)} &= I \\ \boldsymbol{W}_2^{(1)} &= \begin{bmatrix} \mathbf{0}_{(3+2S)\times(3+S)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{S\times S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \end{aligned} \tag{56}$$

Which simply extracts $\boldsymbol{u}_n$ from $\widetilde{\boldsymbol{x}}_n^{(1)}$. With the subsequent layer norm and residual connection, the output of the first layer is,

$$\boldsymbol{x}_n^{(2)} = \left[\begin{array}{c|c|c|c|c} \mathbf{0}_{1\times 3} & e_{x_n}^S & \boldsymbol{u}_n & \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} & \mathbf{0}_{1\times 3S} \end{array}\right]^T \tag{57}$$

Layer 2. In this layer, the relative position encodings and query matrix are the same as in layer 1 but the key matrix is chosen as,

$$\boldsymbol{W}_K^{(2)} = \sqrt{\kappa} \begin{bmatrix} 0 & \mathbf{1}_{1\times 2} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{58}$$

With this choice, observe that,

$$\left\langle \boldsymbol{W}_K^{(2)}(\boldsymbol{x}_{n-i}^{(2)} + \boldsymbol{p}_i^{(1),K}), \boldsymbol{W}_Q^{(2)} \boldsymbol{x}_n^{(2)} \right\rangle = \kappa(i+1)\log(3) \cdot \mathbb{I}(1 \leqslant i \leqslant k) \tag{59}$$

As before, since $\kappa \to \infty$, this results in the attention pattern,

$$\mathrm{att}_{n,n-i}^{(2)} = \frac{3^i \mathbb{I}(1 \leqslant i \leqslant \min\{k, n-1\})}{\sum_{i'=1}^{\min\{k,n-1\}} 3^{i'}} \tag{60}$$

which is similar, but subtly different from the attention pattern in the first layer (Equation (54)). The first layer focuses on indices $n-i$ such that $0 \leqslant i \leqslant k-1$, while this layer focuses on $1 \leqslant i \leqslant k$. Choosing the value and projection matrices as,

$$\boldsymbol{W}_V^{(2)} = \begin{bmatrix} I_{3\times 3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0}_{3S\times 3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{S\times S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{61}$$

The output of the attention layer (with the first residual connection) is,

$$\begin{aligned} \tilde{\boldsymbol{x}}_n^{(2)} &= \left[\begin{array}{c|c|c|c|c|c|c} Z_n & \mathbf{0}_{1\times 2} & e^S_{x_n} & \boldsymbol{u}_n & \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} & \boldsymbol{v}_n & \mathbf{0}_{1\times 2S} \end{array} \right]^T, \\ \text{where, } \boldsymbol{v}_n &= \sum_{i=1}^{\min\{k,n-1\}} \mathrm{att}_{n,n-i}\, e^S_{x_{n-i}}, \\ \text{and, } Z_n &= \sum_{i=1}^{\min\{k,n-1\}} \mathrm{att}_{n,n-i}\, 3^i, \end{aligned} \tag{62}$$

It is a short calculation to see that $Z_n = 3^{k+1}/5$ if $n \geqslant k+1$ and otherwise, $Z_n \leqslant 3^k/5$. This will be useful later, since the value of Z_n can be used to determine whether $n \geqslant k+1$ or $n \leqslant k$ which will allow the the next layer to avoid calculating the attention at $i \leqslant k$, where the evaluation $x_n = x_{i-1}, \cdots, x_{n-k+1} = x_{i-k}$ is not well defined. In the subsequent FFN layer, we will choose,

$$\begin{aligned} \boldsymbol{W}_1^{(2)} &= I \\ \boldsymbol{W}_2^{(2)} &= \begin{bmatrix} \mathbf{0}_{(3+4S)\times(3+3S)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{S\times S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0}_{S\times 2S} \end{bmatrix} \end{aligned} \tag{63}$$

Which extracts $\boldsymbol{v}_n$ from the embedding $\tilde{\boldsymbol{x}}_n^{(2)}$. With the layer norm and adding the final residual connection, the output of this layer is,

$$\boldsymbol{x}_n^{(3)} = \left[\begin{array}{c|c|c|c|c|c|c|c} Z_n & \mathbf{0}_{2\times 1} & e^S_{x_n} & \boldsymbol{u}_n & \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} & \boldsymbol{v}_n & \frac{\boldsymbol{v}_n}{\|\boldsymbol{v}_n\|_2} & \mathbf{0}_{S\times 1} \end{array} \right]^T \tag{64}$$

Layer 3. In this layer, all the relative position encodings are set as $\mathbf{0}$ and instead,

$$\begin{aligned} \boldsymbol{W}_Q^{(3)} &= \sqrt{2\kappa} \begin{bmatrix} 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}_{S\times(2+3S)} & I_{S\times S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\ \boldsymbol{W}_K^{(3)} &= \sqrt{2\kappa} \cdot \begin{bmatrix} 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}_{S\times(2+4S)} & I_{S\times S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \end{aligned} \tag{65}$$

With these choices,

$$\begin{aligned}\left\langle \boldsymbol{W}_K^{(3)}\boldsymbol{x}_i^{(3)}, \boldsymbol{W}_Q^{(3)}\boldsymbol{x}_n^{(3)} \right\rangle &= 2\kappa Z_i Z_n + \frac{2\kappa\langle \boldsymbol{v}_i, \boldsymbol{u}_n\rangle}{\|\boldsymbol{v}_i\|_2 \cdot \|\boldsymbol{u}_n\|_2} \\ &= 2\kappa Z_i Z_n + 2\kappa - \kappa \left\| \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|_2} - \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} \right\|^2 \end{aligned} \tag{66}$$

The resulting attention scores are,

$$\mathrm{att}_{n,i}^{(3)} \propto \exp\left(-\kappa \left\| \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|_2} - \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} \right\|^2 + 2\kappa Z_i Z_n \right) \tag{67}$$

Recall that $\mathcal{I}_n = \{k+1 \leqslant i \leqslant n : \forall j \in [k],\ x_{n-j+1} = x_{i-j}\}$. Then for any $i \in \mathcal{I}_n$, $\boldsymbol{v}_i = \boldsymbol{u}_n$, and by Lemma C.1, for any $i \geqslant k+1$ but not in $\mathcal{I}_n$,

$$\left\| \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|_2} - \frac{\boldsymbol{u}_n}{\|\boldsymbol{u}_n\|_2} \right\|_2 \geqslant \frac{1}{3^k}.$$

Note that this gap is small but non-zero. Furthermore, recall that $Z_i = 3^{k+1}/5$ if $i \geqslant k$ and otherwise $Z_i \leqslant 3^k/5$. Thus the attention prefers values of i such that $\boldsymbol{v}_i = \boldsymbol{u}_n$ and such that $i \geqslant k+1$. In particular, as $\kappa \to \infty$, the resulting attention pattern is,

$$\mathrm{att}_{n,\cdot}^{(3)} = \mathrm{Unif}(\mathcal{I}_n). \tag{68}$$

Choosing,

$$\boldsymbol{W}_V^{(3)} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0}_{S\times 3} & I_{S\times S} & \mathbf{0} \end{bmatrix}.$$

We get that,

$$\widetilde{\boldsymbol{x}}_n^{(3)} = \boldsymbol{x}_n^{(3)} + \sum_{i=1}^{n} \mathrm{att}_{n,i}^{(3)} \begin{bmatrix} \mathbf{0} \\ e_{x_i}^S \end{bmatrix} = \boldsymbol{x}_n^{(3)} + \frac{1}{|\mathcal{I}_n|} \sum_{i \in \mathcal{I}_n} \begin{bmatrix} \mathbf{0} \\ e_{x_i}^S \end{bmatrix}.$$

The feedforward layer is chosen to have $\boldsymbol{W}_1^{(3)} = \boldsymbol{W}_2^{(3)} = \mathbf{0}$, and the overall output of the final transformer layer is therefore just $\widetilde{\boldsymbol{x}}_n^{(3)}$. In the output linear layer, choose,

$$\begin{aligned} \boldsymbol{A} &= \begin{bmatrix} \mathbf{0}_{S\times(d-S)} & I_{S\times S} \end{bmatrix} \\ \boldsymbol{b} &= \mathbf{0} \end{aligned} \tag{69}$$

which results in,

$$\mathrm{logit}_n = \frac{1}{|\mathcal{I}_n|} \sum_{i \in \mathcal{I}_n} e_{x_i}^S = \sum_{i=k+1}^{n} \frac{\mathbb{I}(\forall 1 \leqslant j \leqslant k,\ x_{i-j} = x_{n-j+1})}{\sum_{i'=k+1}^{n} \mathbb{I}(\forall 1 \leqslant j \leqslant k,\ x_{i'-j} = x_{n-j+1})} \cdot e_{x_i}$$

In particular,

$$\mathrm{logit}_T(x_{T+1}) = \frac{\sum_{n=k+1}^{T} \mathbb{I}(\forall 0 \leqslant i \leqslant k,\ x_{n-i} = x_{T-i+1})}{\sum_{n=k+1}^{T} \mathbb{I}(\forall 1 \leqslant i \leqslant k,\ x_{n-i} = x_{T-i+1})} \tag{70}$$

which is the conditional k-gram.

D. Representation lower bounds for 1-layer transformers: Proof of Theorem 6.1

We prove this lower bound by a reduction to communication complexity, and specifically to the set disjointness problem.

Suppose Alice and Bob are given strings $\boldsymbol{a}, \boldsymbol{b} \in \{0,1\}^n$ which are indicator vectors of sets A and B. Their goal is to jointly compute $\mathrm{DIS}(\boldsymbol{a}, \boldsymbol{b}) = \mathbb{I}(\exists i : \boldsymbol{a}_i = \boldsymbol{b}_i = 1)$, which indicates whether A and B intersect or not. Alice and Bob may send a single bit message to the other party over a sequence of communication rounds. The following seminal result by (Yao, 1979) asserts a lower bound on amount of communication required between Alice and Bob to carry out this task.

Theorem D.1 ((Yao, 1979))**.** *Any deterministic protocol for computing* $\mathrm{DIS}(\boldsymbol{a}, \boldsymbol{b})$ *requires at least n rounds of communication.*

We show that a 1-layer transformer with sufficiently small embedding dimension / number of heads can be used to simulate a two-way communication protocol between Alice and Bob to solve $\mathrm{DIS}(\boldsymbol{a}, \boldsymbol{b})$ in a way which contradicts Yao's lower bound in Theorem D.1.

With $m = T/3 - 1$, suppose Alice and Bob have length m bit strings $\boldsymbol{a}, \boldsymbol{b} \in \{0,1\}^m$. The transformer's input will be a sequence of the form,

$$2, \boldsymbol{a}_1, \boldsymbol{b}_1, 2, \boldsymbol{a}_2, \boldsymbol{b}_2, \cdots, 2, \boldsymbol{a}_m, \boldsymbol{b}_m, 2, 1, \tag{71}$$

of length $3m + 2 = T - 1$. The input basically contains a repeating motif, composed of the symbol 2 followed by one of Alice's bits, and then one of Bob's bits. The last 2 symbols are 2 and 1. We will consider the empirical conditional 3-gram probability the transformer associates with the symbol $x_T = 2$. Noting that $x_{T-1} = 1$ and $x_{T-2} = 1$, the conditional 3-gram is computed to be,

$$\frac{\sum_{i=3}^{T-1} \mathbb{I}(x_i = 1, x_{i-1} = 1, x_{i-2} = 2)}{\sum_{i=3}^{T-1} \mathbb{I}(x_{i-1} = 1, x_{i-2} = 2)} \tag{72}$$

Note that if $x_{i-2} = 2$, then i must be of the form $3j$ for $j = 1, \cdots, n$, and we may rewrite the sum as,

$$\frac{\sum_{j=1}^{m} \mathbb{I}(x_{3j} = 1, x_{3j-1} = 1)}{\sum_{j=1}^{m} \mathbb{I}(x_{3j-1} = 1)} = \frac{|A \cap B|}{|B|} \tag{73}$$

Now, let us use the transformer to construct a deterministic communication protocol between Alice and Bob. Alice is given $(x_2, x_5, \cdots, x_{3m-1}) = (\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_m)$ and Bob is given $(x_3, x_6, \cdots, x_{3m}) = (\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_m)$.

In the first round, Alice computes the normalization in the softmax of the attention which comes from the set of inputs she holds. For simplifying notation define,

$$\mathrm{score}^{(h)}(i) = \exp\left(\left\langle \boldsymbol{W}_K^{(h)}(\mathtt{Emb}(x_i) + \boldsymbol{p}_i), \boldsymbol{W}_Q^{(h)}\mathtt{Emb}(x_{T-1})\right\rangle\right) \tag{74}$$

In particular, for each head $h \in [H]$, she computes,

$$Z_{\text{Alice}}^{(h)} = \log\left(\sum_{j=1}^{m} \mathrm{score}^{(h)}(3j-1)\right) \tag{75}$$

Assuming that the transformer uses p bits of precision, Alice communicates $Z_{\text{Alice}}^{(h)}$ for each h, which corresponds to pH bits of communication. With this information, Bob completes the rest of the normalization term (again up to p bits of precision) and computes,

$$Z^{(h)} = \log\left(Z_{\text{Alice}}^{(h)} + Z_{\text{Bob}}^{(h)} + Z_{\text{common}}^{(h)}\right), \tag{76}$$

$$\text{where } Z_{\text{Bob}}^{(h)} = \log\left(\sum_{j=1}^{m} \mathrm{score}^{(h)}(3j)\right) \tag{77}$$

$$\text{and } Z_{\text{common}}^{(h)} = \log\left(\sum_{j=1}^{m} \mathrm{score}^{(h)}(3j-2) + \mathrm{score}^{(h)}(T-2) + \mathrm{score}^{(h)}(T-1)\right) \tag{78}$$

which is the overall normalization term in the softmax. This is communicated back to Alice, using another pH bits of communication. Next using this information, Alice computes the output of the attention layer, taking the convex combination corresponding to the inputs she knows. In particular, for each $h \in [H]$ she computes,

$$\sum_{j=1}^{n} \frac{\text{score}^{(h)}(3j-1)}{\exp(Z^{(h)})} \texttt{Emb}(x_{3j-1}) \in \mathbb{R}^d. \tag{79}$$

across all the heads. Rather than transmitting everything, she concatenates the outputs of the heads, and multiplies them by the value and projection matrices to result in the output $\boldsymbol{y}_{\text{Alice}}$ which is d-dimensional. This is sent to Bob using dp bits of communication. Subsequently, Bob computes the terms in the attention corresponding to the inputs he knows as well as the public inputs (all the 2's at positions $3j-2$ as well as the last two symbols). In particular,

$$\sum_{j=1}^{m} \frac{\text{score}^{(h)}(3j)}{\exp(Z^{(h)})} \texttt{Emb}(x_{3j}) + \sum_{j=1}^{m+1} \frac{\text{score}^{(h)}(3j-2)}{\exp(Z^{(h)})} \texttt{Emb}(2) + \frac{\text{score}^{(h)}(T-1)}{\exp(Z^{(h)})} \texttt{Emb}(1) \tag{80}$$

These are yet again concatenated across all the heads and multiplied by the value and projection matrices to result in the output $\boldsymbol{y}_{\text{Bob}}$ which is added to $\boldsymbol{y}_{\text{Alice}}$ to result in $\boldsymbol{y}$. Bob passes $\boldsymbol{y}$ through the residual connection, layer norm, and feedforward layers, and subsequently through the linear layer and softmax of the model to result in the output of the model. By assumption, the output of the model approximately captures the conditional 3-gram, which by Equation (73) equals $|A \cap B|/|B|$. Note that if $|A \cap B|/|B|$ is non-zero, it must be at least $1/T$. This means, if the transformer is able to compute the conditional 3-gram to within an additive error of $1/3T$, then Bob can simply threshold the output of the transformer to decide whether $A \cap B = \varnothing$ or not, thereby solving $\text{DIS}(\boldsymbol{a}, \boldsymbol{b})$.

Since this communication protocol is deterministic, by Yao's lower bound in Theorem D.1, the number of bits communicated between Alice and Bob must be at least $m = T/3 - 1$. The total number of bits of communication in the protocol is $2pH + dp + 1$ (the last 1 comes from Bob having to communicate the answer to Alice), completing the proof.

E. Lower bounds on representing k^{th}-order induction heads: Proof of Theorems 6.3 to 6.5

In this section we prove a size-lower bound on attention-only transformers representing k^{th}-order induction heads which generalizes both Theorem 6.3 and Theorem 6.4. We will first prove the result for the case $L = 2$ and $H = 1$, showing that they cannot represent k^{th}-order induction heads for $k \geqslant 4$ under Assumption 6.2. We subsequently extend it to the general L-layer transformer in Appendix E.2 and to the general case with H_ℓ heads in layer $\ell \in [L]$ in Appendix E.3.

E.1. Lower bounds on 2-layer 1-head attention-only transformers

In this section we show that under Assumption 6.2, a 2-layer 1-head attention-only transformer cannot represent k^{th}-order induction heads for any $k \geqslant 4$. We will prove lower bounds on the transformer when the input is binary, i.e., $S = \{0, 1\}$. With relative position embeddings, observe that the first layer of the transformer model learns representations of the form,

$$\boldsymbol{x}_n^{(2)} = \texttt{Emb}(x_n) + \sum_{i \leqslant n} \text{att}_{n,i}^{(1)} \boldsymbol{W}_V^{(1)} \texttt{Emb}(x_i) + \sum_{i \leqslant n} \boldsymbol{W}_V^{(1)} \boldsymbol{p}_{n-i}^{V,(1)} \tag{81}$$

where note that the attention pattern only depends on n and i and not on x_i or x_n. These representations are input into the second layer, which realizes the attention pattern $\text{att}_{n,i}^{(2)}$, which is proportional to,

$$\exp\left(\left\langle \boldsymbol{W}_K^{(2)} \big(\boldsymbol{x}_i^{(2)} + \boldsymbol{p}_{n-i}^{K,(2)} \big), \boldsymbol{W}_Q^{(2)} \boldsymbol{x}_n^{(2)} \right\rangle \right). \tag{82}$$

We need this function to be maximized uniquely when $x_{i-1} = x_n, \cdots, x_{i-k} = x_{n-k+1}$. Denoting $\phi(0) = \boldsymbol{W}_V^{(1)} \texttt{Emb}(0)$ and $\phi(1) = \boldsymbol{W}_V^{(1)} \texttt{Emb}(1)$,

$$\begin{aligned} \boldsymbol{x}_n^{(2)} &= \texttt{Emb}(x_n) + \sum_{i \leqslant n} \text{att}_{n,i}^{(1)} \boldsymbol{W}_V^{(1)} \texttt{Emb}(x_i) + \sum_{i \leqslant n} \boldsymbol{W}_V^{(1)} \boldsymbol{p}_{n-i}^{(1),V} && (83) \\ &= x_n \texttt{Emb}(1) + (1-x_n) \texttt{Emb}(0) + \sum_{i \leqslant n} \text{att}_{n,i}^{(1)} \big(x_i \cdot \phi(1) + (1-x_i) \cdot \phi(0) \big) + \sum_{i \leqslant n} \boldsymbol{W}_V^{(1)} \boldsymbol{p}_{n-i}^{(1),V} && (84) \end{aligned}$$

$$= \left(\frac{\texttt{Emb}(1) + \texttt{Emb}(0)}{2} + x'_n \cdot \frac{\texttt{Emb}(1) - \texttt{Emb}(0)}{2} \right) + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(1)} \left(\frac{\phi(1) + \phi(0)}{2} + x'_i \cdot \frac{\phi(1) - \phi(0)}{2} \right) \\ + \sum_{i \leqslant n} \boldsymbol{W}_V^{(1)} \boldsymbol{p}_{n-i}^{(1),V} \tag{85}$$

where $x'_i \leftarrow 2x_i - 1$. We can write this down as,

$$\boldsymbol{x}_n^{(2)} = \boldsymbol{m}_n^{(1)} + \boldsymbol{M}_n^{(1)} \begin{bmatrix} x'_n & x'_{n-1} & \cdots & x'_1 \end{bmatrix}^T \tag{86}$$

where $\boldsymbol{M}_n^{(1)}$ is a matrix of rank at most 2 and of the form,

$$\boldsymbol{M}_n^{(1)} = \left(\frac{\phi(1) - \phi(0)}{2} \right) \begin{bmatrix} \mathrm{att}_{n,n}^{(1)} & \cdots & \mathrm{att}_{n,1}^{(1)} \end{bmatrix} + \left(\frac{\texttt{Emb}(1) - \texttt{Emb}(0)}{2} \right) \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \tag{87}$$

which is independent of $x'_1, \cdots, x'_n$. Likewise $\boldsymbol{m}_n^{(1)}$ collects all the vectors in the sum that don't depend on $x'_1, \cdots, x'_n$. Now, observe that in the next layer, we wish to show that an induction head cannot be realized by $\mathrm{att}_{n,i}^{(2)}$ for each $i \leqslant n$. We will show this for any value of $i \leqslant n - k$.

In the second layer, we may write down the key vectors as,

$$\boldsymbol{W}_K^{(2)} \left(\boldsymbol{x}_i^{(2)} + \boldsymbol{p}_{n-i}^{(2),K} \right) = \boldsymbol{W}_K^{(2)} \boldsymbol{m}_i^{(1)} + \boldsymbol{W}_K^{(2)} \boldsymbol{M}_i^{(1)} \begin{bmatrix} x'_i & x'_{i-1} & \cdots & x'_1 \end{bmatrix}^T + \boldsymbol{W}_K^{(2)} \boldsymbol{p}_{n-i}^{(2),K}. \tag{88}$$

Again, defining the vector $\overline{\boldsymbol{m}}_i^{(1)}$ and the matrix $\overline{\boldsymbol{M}}_i^{(1)}$ appropriately (having rank at most 2), this equals,

$$\overline{\boldsymbol{m}}_i^{(1)} (\{x'_i\} \cup \{x'_{-k-1}, \cdots, x'_1\}) + \overline{\boldsymbol{M}}_i^{(1)} \boldsymbol{y} \tag{89}$$

where $\boldsymbol{y} \triangleq \begin{bmatrix} x'_{i-1} & \cdots & x'_{i-k} \end{bmatrix}^T$ and the vector $\overline{\boldsymbol{m}}_i^{(1)}$ depends on x'_i as well as the inputs $x'_{i-k-1}, \cdots, x'_1$, which in this context, are treated as nuisance variables since they do not intersect with $\{x'_{i-1}, \cdots, x'_{i-k}\} \cup \{x_n, \cdots, x_{n-k+1}\}$. Henceforth we will avoid explicitly stating the dependency of $\overline{\boldsymbol{m}}_i^{(1)}$ on the x_j's. Similarly, the query vector can be written down as,

$$\boldsymbol{W}_Q^{(2)} \boldsymbol{x}_n^{(2)} = \widehat{\boldsymbol{m}}_n^{(1)} + \widecheck{\boldsymbol{M}}_n^{(1)} \boldsymbol{x} + \widehat{\boldsymbol{M}}_n^{(1)} \boldsymbol{y} \tag{90}$$

where $\widehat{\boldsymbol{m}}_n^{(1)}$, $\widecheck{\boldsymbol{M}}_n^{(1)}$ and $\widehat{\boldsymbol{M}}_n^{(1)}$ are defined appropriately, with $\widehat{\boldsymbol{M}}_n^{(1)}$ and $\widehat{\boldsymbol{M}}_n^{(1)}$ of rank at most 2, and $\boldsymbol{x}$ is defined as $\begin{bmatrix} x'_n & \cdots & x'_{n-k+1} \end{bmatrix}^T$. For an appropriate matrix $\boldsymbol{M}_{n,i}^{\times}$, vectors $\boldsymbol{m}_{n,i}^{\times}$ and $\widetilde{\boldsymbol{m}}_{n,i}^{\times}$ and scalar $m_{n,i}^{\times}$, the dot-product of the key and query vectors can be written as,

$$\left\langle \boldsymbol{W}_K^{(2)} \big(\boldsymbol{x}_i^{(2)} + \boldsymbol{p}_{n-i}^{(2),K} \big), \boldsymbol{W}_Q^{(2)} \boldsymbol{x}_n^{(2)} \right\rangle \\ = \boldsymbol{x}^T \boldsymbol{M}_{n,i}^{\times} \boldsymbol{y} + \boldsymbol{y}^T \big(\overline{\boldsymbol{M}}_{n,i}^{\times} \big) \boldsymbol{y} + (\boldsymbol{m}_{n,i}^{\times})^T \boldsymbol{x} + (\overline{\boldsymbol{m}}_{n,i}^{\times})^T \boldsymbol{y} + m_{n,i}^{\times} \triangleq f_{n,i}(\boldsymbol{x}, \boldsymbol{y}), \tag{91}$$

Which is a linear function in $\boldsymbol{x}$ and quadratic in $\boldsymbol{y}$, both of which lie on $\{\pm 1\}^k$. Note that the matrix $\boldsymbol{M}_{n,i}^{\times}$ has rank at most 2 since it is a product of $\overline{\boldsymbol{M}}_i^{(1)}$ and $\widecheck{\boldsymbol{M}}_n^{(1)}$, each with rank at most 2. Next we introduce a lemma showing that if $\boldsymbol{M}_{n,i}^{\times}$ is inherently low rank, the quadratic form in Equation (91) which captures the dot-product between the key and value vectors cannot satisfy the property that for every $\boldsymbol{y}$, the function is uniquely maximized at $\boldsymbol{x} = \boldsymbol{y}$. In particular, this means that for any $i \leqslant n - k$, there is some choice of $x_n, x_{n-1}, \cdots, x_{n-k+1}$ such that there are $x_{i-1}, \cdots, x_{i-k}$ such that for at least one $j \in [k]$, x_{i-j} and x_{n-j-1} are not equal, but the attention score is larger than the case when x_{i-j} were equal to x_{n-j-1} for each $j \in [k]$.

Lemma E.1. *If $\boldsymbol{M}_{n,i}^{\times}$ has rank $\leqslant k - 2$, it is impossible for $f_{n,i}(\boldsymbol{x}, \boldsymbol{y})$ to satisfy the property that for every $\boldsymbol{y} \in \{\pm 1\}^k$, the maximizer is uniquely $\boldsymbol{x} = \boldsymbol{y}$.*

The proof is almost complete: if $k \geqslant 4$, then the rank of $\boldsymbol{M}_{n,i}^{\times}$, which is at most 2, does not exceed $k - 2$. This means that when $k \geqslant 4$, any attention pattern realized in the second layer must satisfy the property that there exists a string such that the attention is no longer uniquely maximized when $x_n = x_{i-1}, \cdots, x_{n-k+1} = x_{i-k}$.

Proof. For the purpose of brevity, define $\mathcal{H}_k = \{\pm 1\}^k$. First consider the reparameterization,

$$\widetilde{\boldsymbol{x}} = \widetilde{\boldsymbol{M}}^{\times}_{n,i}\boldsymbol{x}, \text{ where } \widetilde{\boldsymbol{M}}^{\times}_{n,i} = \begin{bmatrix}(\boldsymbol{M}^{\times}_{n,i})^T \\ (\boldsymbol{m}^{\times}_{n,i})^T\end{bmatrix}. \tag{92}$$

Then, the dot-product of the key and query matrices can be written as,

$$\begin{bmatrix}\boldsymbol{y}^T & 1\end{bmatrix}\widetilde{\boldsymbol{x}} + \boldsymbol{y}^T\big(\overline{\boldsymbol{M}}^{\times}_{n,i}\big)\boldsymbol{y} + (\overline{\boldsymbol{m}}^{\times}_{n,i})^T\boldsymbol{y} + m^{\times}_{n,i} \tag{93}$$

Note that this function is linear in $\widetilde{\boldsymbol{x}}$ and therefore must be maximized on a vertex of the convex hull of the domain, $\widetilde{\boldsymbol{M}}^{\times}_{n,i}\mathcal{H}_k \triangleq \big\{\widetilde{\boldsymbol{M}}^{\times}_{n,i}\boldsymbol{h} : \boldsymbol{h} \in \mathcal{H}_k\big\}$. If $\boldsymbol{M}^{\times}_{n,i}$ has rank at most $k-2$, the rank of $\widetilde{\boldsymbol{M}}^{\times}_{n,i}$ is at most $k-1$ and cannot be full rank. We show that this must imply that there is a vertex $\boldsymbol{v} \in \mathcal{H}_k$ such that $\widetilde{\boldsymbol{M}}^{\times}_{n,i}\boldsymbol{v}$ is not a unique vertex of the convex hull of $\widetilde{\boldsymbol{M}}^{\times}_{n,i}\mathcal{H}_k$. This means that $\boldsymbol{v}$ cannot be a unique maximizer for $\widetilde{\boldsymbol{x}}$ when maximizing over all strings in Equation (93), and specifically $\boldsymbol{y} = \boldsymbol{v}$ is a witness to Lemma E.1.

Below we discuss how to find such a vector $\boldsymbol{v}$. Note that $\widetilde{\boldsymbol{M}}^{\times}_{n,i}$ is not full rank, which implies that there exists a vector $\boldsymbol{n}$ such that $\widetilde{\boldsymbol{M}}^{\times}_{n,i}\boldsymbol{n} = \boldsymbol{0}$. Without loss of generality, let $\boldsymbol{n}_1$ be the smallest non-zero coordinate of $\boldsymbol{n}$ in absolute value. Then the vector $\boldsymbol{n}_1^{-1}\boldsymbol{n}$ has no non-zero coordinates in the interval $(-1, 1)$. We will show that $\mathrm{sign}(\boldsymbol{n}_1^{-1}\boldsymbol{n})$ is a good choice for $\boldsymbol{v}$.

Consider two cases,

Case I. Every non-zero coordinate of $\boldsymbol{n}_1^{-1}\boldsymbol{n}$ is in $\{\pm 1\}$. Consider any $\boldsymbol{x} \in \mathcal{H}_k$ which matches with $\boldsymbol{n}$ on the non-zero coordinates. Consider $\boldsymbol{x}'$ which is the same as $\boldsymbol{x}$, except a negation is taken on the coordinates where $\boldsymbol{n}$ is non-zero. Note that $\widetilde{\boldsymbol{M}}^{\times}_{n,i}\boldsymbol{x} = \widetilde{\boldsymbol{M}}^{\times}_{n,i}\boldsymbol{x}'$, for the same value of $\boldsymbol{x}$. This means that for any $\boldsymbol{y}$. In particular, from Equation (93), both $\boldsymbol{x}$ and $\boldsymbol{x}'$ are maximizers, showing that Lemma E.1 is true in this case. We circumvent having to find such a vector $\boldsymbol{v}$ in this case.

Case II. $\boldsymbol{n}_1^{-1}\boldsymbol{n}$ has non-zero coordinates which are not all in $\{\pm 1\}$. In particular, at least one coordinate where this vector is strictly less than -1 or strictly greater than $+1$. In this case, observe that the sign vector $\widetilde{\boldsymbol{n}} = \mathrm{sign}(\boldsymbol{n}_1^{-1}\boldsymbol{n}) \in \mathcal{H}_k$ lies within, but is not a vertex of the convex hull of the set $\mathcal{H}_k \cup \{\boldsymbol{n}_1^{-1}\boldsymbol{n}\}$. The reason for this is simple to see when we assume that $\boldsymbol{n}_1^{-1}\boldsymbol{n}$ has only one coordinate which is not in $[-1, 1]$, say, the coordinate $j = 2$: here, $\widetilde{\boldsymbol{n}}$ can be written down as a convex combination (with non-zero coefficients) of $\boldsymbol{n}_1^{-1}\boldsymbol{n}$ and $\widetilde{\boldsymbol{n}}^{(2)}$; the latter vector is obtained by flipping coordinate 2 of $\widetilde{\boldsymbol{n}}$. When there is more than one coordinate not in $[-1, 1]$, we can peel away these large coordinates in $\boldsymbol{n}_1^{-1}\boldsymbol{n}$ by taking a convex combination of this vector with the vectors $\widetilde{\boldsymbol{n}}^{(j)}$ for the appropriate values of j, to return the sign vector $\widetilde{\boldsymbol{n}}$. Here, $\widetilde{\boldsymbol{n}}^{(j)}$ is the version of $\widetilde{\boldsymbol{n}}$ where the j^{th}-coordinate is flipped. This results in the following claim.

Claim 1. The sign vector $\widetilde{\boldsymbol{n}}$ lies within the convex hull of the points $\mathcal{H}_k \cup \{\boldsymbol{n}_1^{-1}\boldsymbol{n}\}$, but is not a vertex of this set.

In particular, we may write,

$$\widetilde{\boldsymbol{n}} = \alpha_0\boldsymbol{n}_1^{-1}\boldsymbol{n} + \sum_{j\in[n]} \alpha_j\widetilde{\boldsymbol{n}}^{(j)}. \tag{94}$$

where $\alpha_0 > 0$ and $\sum_{j=0}^n \alpha_i = 1$. By left-multiplying this on both sides by $\widetilde{\boldsymbol{M}}^{\times}_{n,i}$ and noting that $\boldsymbol{n}$ lies in the null-space of this matrix, we get,

$$\widetilde{\boldsymbol{M}}^{\times}_{n,i}\widetilde{n} = \sum_{j\in[n]} \alpha_j\widetilde{\boldsymbol{M}}^{\times}_{n,i}\widetilde{\boldsymbol{n}}^{(j)} \tag{95}$$

where note that $\sum_{j\in[n]}\alpha_j$ is strictly less than 1, since $\alpha_0 > 0$. We may write this vector as,

$$\begin{aligned}\widetilde{\boldsymbol{M}}^{\times}_{n,i}\widetilde{n} &= \alpha_0\boldsymbol{0} + \sum_{j\in[n]} \alpha_j\widetilde{\boldsymbol{M}}^{\times}_{n,i}\widetilde{\boldsymbol{n}}^{(j)} \\ &= \frac{\alpha_0}{2^k}\sum_{\boldsymbol{h}\in\mathcal{H}_k}\widetilde{\boldsymbol{M}}^{\times}_{n,i}\boldsymbol{h} + \sum_{j\in[n]} \alpha_j\widetilde{\boldsymbol{M}}^{\times}_{n,i}\widetilde{\boldsymbol{n}}^{(j)} \end{aligned} \tag{96}$$

Since $\alpha_0 > 0$, this equation implies that the image of $\widetilde{\boldsymbol{n}}$ under $\widetilde{\boldsymbol{M}}^{\times}_{n,i}$ itself falls within $\mathrm{conv}(\widetilde{\boldsymbol{M}}^{\times}_{n,i}\mathcal{H}_k)$, but is itself not a vertex of this set. This means that $\widetilde{\boldsymbol{n}}$ can never be a maximizer of $f_{n,i}\big(\cdot, \boldsymbol{y}\big)$ for any $\boldsymbol{y}$, and in particular when $\boldsymbol{y} = \widetilde{\boldsymbol{n}}$, thereby proving Lemma E.1. □

E.2. L-layer attention-only transformers with 1 head per layer: Proof of Theorem 6.3

Proof. The proof largely tracks the 2-layer case, with the main exception that we keep track of how the maximum possible rank of the matrix $\boldsymbol{M}_{n,i}^{\times}$ grows as a function of the depth of the transformer. In the case the 2-layer transformer, we show that it cannot exceed 2. With the addition of more layers, we show that it cannot exceed 2^{L-1}.

Recall from the notation in Equation (86) that the output of the first attention layer is,

$$\boldsymbol{x}_n^{(2)} = \boldsymbol{m}_n^{(1)} + \boldsymbol{M}_n^{(1)} \begin{bmatrix} x'_n & x'_{n-1} & \cdots & x'_1 \end{bmatrix}^T \tag{97}$$

where $\boldsymbol{M}_n^{(1)} \in \mathbb{R}^{d \times n}$ has rank at most 2. Let us rewrite this as,

$$\boldsymbol{x}_n^{(2)} = \boldsymbol{m}_n^{(1)} + \overline{\boldsymbol{M}}_n^{(1)} \begin{bmatrix} x'_T & x'_{T-1} & \cdots & x'_1 \end{bmatrix}^T \tag{98}$$

where $\boldsymbol{M}_n^{(1)} \in \mathbb{R}^{d \times T}$ is causally masked to be 0's when it operates on x_i for all indices $i > n$. Note that even with this causal masking, $\overline{\boldsymbol{M}}_n^{(1)}$ has rank at most 2, as discussed in Equation (86).

By induction, assume that the output of the $(\ell-1)^{\text{th}}$ attention layer is of the form,

$$\boldsymbol{x}_n^{(\ell)} = \boldsymbol{m}_n^{(\ell-1)} + \overline{\boldsymbol{M}}_n^{(\ell-1)} \boldsymbol{x}_{1:T} \tag{99}$$

where $\boldsymbol{x}_{1:T} \triangleq \begin{bmatrix} x'_T & x'_{T-1} & \cdots & x'_1 \end{bmatrix}^T$. Passing $\boldsymbol{x}_n^{(\ell)}$ through the ℓ^{th} attention layer, we get,

$$\begin{aligned}
\boldsymbol{x}_n^{(\ell+1)} &= \boldsymbol{x}_n^{(\ell)} + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \left(\boldsymbol{x}_i^{(\ell)} + \boldsymbol{p}_{n-i}^{(\ell),V} \right) \qquad (100)\\
&= \boldsymbol{m}_n^{(\ell-1)} + \overline{\boldsymbol{M}}_n^{(\ell-1)} \boldsymbol{x}_{1:T} + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \boldsymbol{m}_i^{(\ell-1)} + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \overline{\boldsymbol{M}}_i^{(\ell-1)} \boldsymbol{x}_{1:T} \\
&\qquad + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \boldsymbol{p}_{n-i}^{(\ell),V} \qquad (101)
\end{aligned}$$

Define,

$$\boldsymbol{m}_n^{(\ell)} = \boldsymbol{m}_n^{(\ell-1)} + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \boldsymbol{m}_i^{(\ell-1)} + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \boldsymbol{p}_{n-i}^{(\ell),V}, \text{ and,} \tag{102}$$

$$\overline{\boldsymbol{M}}_n^{(\ell)} = \overline{\boldsymbol{M}}_n^{(\ell-1)} + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \overline{\boldsymbol{M}}_i^{(\ell-1)} \tag{103}$$

Then, we can write down,

$$\boldsymbol{x}_n^{(\ell+1)} = \boldsymbol{m}_n^{(\ell)} + \overline{\boldsymbol{M}}_n^{(\ell)} \boldsymbol{x}_{1:T} \tag{104}$$

We also inductively assume that for every $i \leqslant n$,

(i) $\overline{\boldsymbol{M}}_i^{(\ell-1)}$ has rank $R \leqslant 2^{\ell-1}$, and,

(ii) $\overline{\boldsymbol{M}}_i^{(\ell-1)}$ can be factorized in the form $\sum_{r=1}^R \boldsymbol{u}_r \cdot \boldsymbol{v}_{i,r}^T$, where only the $\boldsymbol{v}_{i,r}$'s depend on i, but the $\boldsymbol{u}_r$'s do not depend on i.

Both of these conditions are true when $\ell - 1 = 1$ as evidenced by the structure of $\boldsymbol{M}_i^{(1)}$ in Equation (87) and noting that $\overline{\boldsymbol{M}}_i^{(1)}$ is obtained from $\boldsymbol{M}_i^{(1)}$ by right multiplying by a diagonal mask matrix. Using the recursion in Equation (102), we prove that the induction hypotheses (i) and (ii) are true at layer ℓ as well. In particular using the decomposition in (ii), observe that,

$$\overline{\boldsymbol{M}}_n^{(\ell)} = \sum_{r=1}^R \boldsymbol{u}_r \cdot \boldsymbol{v}_{n,r}^T + \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)} \boldsymbol{W}_V^{(\ell)} \sum_{r=1}^R \boldsymbol{u}_r \cdot \boldsymbol{v}_{i,r}^T \tag{105}$$

$$= \sum_{r=1}^{R} \boldsymbol{u}_r \cdot \boldsymbol{v}_{n,r}^T + \sum_{r=1}^{R} \boldsymbol{W}_V^{(\ell)} \boldsymbol{u}_r \cdot \left(\textstyle\sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)}\, \boldsymbol{v}_{i,r} \right)^T \tag{106}$$

$$= \sum_{r=1}^{2R} \boldsymbol{u}_r \cdot \boldsymbol{v}_{n,r}^T \tag{107}$$

where for $r' \in [R]$, $\boldsymbol{u}_{R+r'} \triangleq \boldsymbol{W}_V^{(\ell)} \boldsymbol{u}_r$ and $\boldsymbol{v}_{n,r'} \triangleq \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(\ell)}\, \boldsymbol{v}_{i,r}$. Since $\boldsymbol{M}_n^{(\ell)}$ is the sum of $2R$ rank 1 matrices and therefore has rank at most $2R \leqslant 2^\ell$, proving both parts of the induction hypothesis.

By induction, at the end of the $(L-1)^{\text{th}}$ layer, we have an output which looks like,

$$\boldsymbol{x}_n^{(L)} = \boldsymbol{m}_n^{(L-1)} + \overline{\boldsymbol{M}}_n^{(L-1)} \boldsymbol{x}_{1:T} \tag{108}$$

where $\boldsymbol{M}_n^{(L-1)}$ has rank at most 2^{L-1}. More importantly, note that by the causal masking, even though it appears to depend on the whole input sequence through $\boldsymbol{x}_{1:T}$, note that $\boldsymbol{x}_n^{(L)}$ only depends on $x_1, \cdots, x_n$ and not on the future inputs to this time n. In particular, by a similar argument as in the 2-layer case (cf. Equation (86) to Equation (91)), for any $i \leqslant n-k$ we can decompose the dot-product of the key and query vectors at the L^{th} layer as a bilinear form which looks like,

$$\begin{aligned}&\left\langle \boldsymbol{W}_K^{(L)} \big(\boldsymbol{x}_i^{(L)} + \boldsymbol{p}_{n-i}^{(L),K}\big), \boldsymbol{W}_Q^{(L)} \boldsymbol{x}_n^{(L)} \right\rangle \\ &\qquad = \boldsymbol{x}^T \boldsymbol{M}_{n,i}^{\times} \boldsymbol{y} + \boldsymbol{y}^T \big(\overline{\boldsymbol{M}}_{n,i}^{\times}\big) \boldsymbol{y} + (\boldsymbol{m}_{n,i}^{\times})^T \boldsymbol{x} + (\overline{\boldsymbol{m}}_{n,i}^{\times})^T \boldsymbol{y} + m_{n,i}^{\times} \triangleq f_{n,i}^{(L+1)}(\boldsymbol{x}, \boldsymbol{y})\end{aligned} \tag{109}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are defined as $\begin{bmatrix} x_n' & \cdots & x_{n-k+1}' \end{bmatrix}^T$ and $\begin{bmatrix} x_{i-1}' & \cdots & x_{i-k}' \end{bmatrix}^T$ respectively, and $\boldsymbol{M}_{n,i}^{\times}$ has rank at most that of $\boldsymbol{M}_n^{(L-1)}$, which is 2^{L-1}. In particular, if $2^{L-1} \leqslant k-2$, by Lemma E.1 the proof concludes. □

E.3. The general case: Transformers with H_ℓ heads in layer ℓ: Proof of Theorem 6.5

The h^{th} head of the first layer of the attention-only transformer learns patterns of the form,

$$\widetilde{\boldsymbol{x}}_n^{(1,h)} = \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(1,h)}\, \boldsymbol{W}_V^{(1,h)} \texttt{Emb}(x_i) + \sum_{i \leqslant n} \boldsymbol{W}_V^{(1,h)} \boldsymbol{p}_{n-i}^{V,(1,h)} \tag{110}$$

$$= \sum_{i \leqslant n} \mathrm{att}_{n,i}^{(1,h)} \left(\frac{\phi^h(0) + \phi^h(1)}{2} + x_i' \cdot \frac{\phi^h(1) - \phi^h(0)}{2} \right) + \sum_{i \leqslant n} \boldsymbol{W}_V^{(1,h)} \boldsymbol{p}_{n-i}^{V,(1,h)} \tag{111}$$

where the last equation assumes a binary input sequence, defines $x_i' = 2x_i - 1$ and uses the notation $\phi^h(0) = \boldsymbol{W}_V^{(1,h)} \texttt{Emb}(0)$ and $\phi^h(1) = \boldsymbol{W}_V^{(1,h)} \texttt{Emb}(1)$. We can further rewrite this as,

$$\widetilde{\boldsymbol{x}}_n^{(1,h)} = \boldsymbol{m}_n^{(1,h)} + \boldsymbol{M}_n^{(1,h)} \boldsymbol{x}_{1:T} \tag{112}$$

where each $\boldsymbol{M}_n^{(1,h)} \in \mathbb{R}^{d \times T}$ is rank 1 and applies a causal mask on the inputs x_i for $i > n$. Recall that the output of the first attention layer applies a projection matrix on the concatentation of $\widetilde{\boldsymbol{x}}_n^{(1,h)}$ across $h \in [H_1]$ and then adds a residual connection. The output can be written down as,

$$\widetilde{\boldsymbol{x}}_n^{(2)} = \texttt{Emb}(x_n) + \boldsymbol{W}_O^{(1)} \begin{bmatrix} \boldsymbol{m}_n^{(1,1)} \\ \vdots \\ \boldsymbol{m}_n^{(1,H_1)} \end{bmatrix} + \boldsymbol{W}_O^{(1)} \begin{bmatrix} \boldsymbol{M}_n^{(1,1)} \\ \vdots \\ \boldsymbol{M}_n^{(1,H_1)} \end{bmatrix} \boldsymbol{x}_{1:T} \tag{113}$$

$$= \boldsymbol{m}_n^{(1)} + \boldsymbol{M}_n^{(1)} \boldsymbol{x}_{1:T}, \tag{114}$$

where,

$$\boldsymbol{M}_n^{(1)} = \left(\frac{\texttt{Emb}(1) + \texttt{Emb}(0)}{2} \right) e_n^T + \boldsymbol{W}_O^{(1)} \begin{bmatrix} \boldsymbol{M}_n^{(1,1)} \\ \vdots \\ \boldsymbol{M}_n^{(1,H_1)} \end{bmatrix}, \text{ and,} \tag{115}$$

$$\boldsymbol{m}_n^{(1)} = \left(\frac{\texttt{Emb}(1) - \texttt{Emb}(0)}{2}\right) + \boldsymbol{W}_O^{(1)} \begin{bmatrix} \boldsymbol{m}_n^{(1,1)} \\ \vdots \\ \boldsymbol{m}_n^{(1,H_1)} \end{bmatrix} \tag{116}$$

Notice that the rank of the matrix $\boldsymbol{M}_n^{(1)}$ is at most $H_1 + 1$. This is because the concatenation operation can increase the rank at most additively, and since each of the $\boldsymbol{M}_n^{(1,h)}$ matrices are rank at most 1.

Following through the proof in Appendix E.2 for the L-layer case, we can prove inductively that at any layer ℓ, the output looks like,

$$\boldsymbol{x}_n^{(\ell)} = \boldsymbol{m}_n^{(\ell)} + \boldsymbol{M}_n^{(\ell)} \boldsymbol{x}_{1:T} \tag{117}$$

where the rank of $\boldsymbol{M}_n^{(\ell)}$ is $\prod_{i=1}^{\ell}(H_i + 1)$. Invoking Lemma E.1, if $\prod_{i=1}^{L-1}(H_i + 1) \leqslant k - 2$, the attention-only transformer cannot realize a k^{th}-order induction head at layer L.

F. Model architecture and hyper-parameters

The experiments were run on one $8 \times A100$ GPU node.

Table 2: Parameters in the transformer architecture with their shape.

Parameter	Matrix shape
transformer.wte	$2 \times d$
transformer.wpe	$N \times d$
transformer.h.ln_1 $(\times \ell)$	$d \times 1$
transformer.h.attn.c_attn $(\times \ell)$	$3d \times d$
transformer.h.attn.c_proj $(\times \ell)$	$d \times d$
transformer.h.ln_2 $(\times \ell)$	$d \times 1$
transformer.h.mlp.c_fc $(\times \ell)$	$4d \times d$
transformer.h.mlp.c_proj $(\times \ell)$	$d \times 4d$
transformer.ln_f	$d \times 1$

Table 3: Settings and parameters for the transformer model used in the experiments.

Dataset	k-th order binary Markov source
Architecture	Based on the GPT-2 architecture as implemented in (Pagliardini, 2023)
Batch size	Grid-searched in $\{8, 16\}$
Accumulation steps	1
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
Learning rate	0.001
Scheduler	Cosine
# Iterations	Up to 25000
Weight decay	1×10^{-3}
Dropout	0
Sequence length	Grid-searched in $\{32, 64, 128, 256, 512, 1024\}$
Embedding dimension	Grid-searched in $\{16, 32, 64\}$
Transformer layers	Between 1 and 8
Attention heads	Up to k
Repetitions	3

G. Additional experimental results

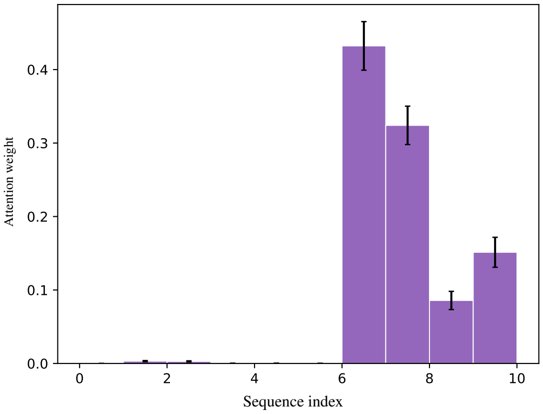

Figure 9: First head

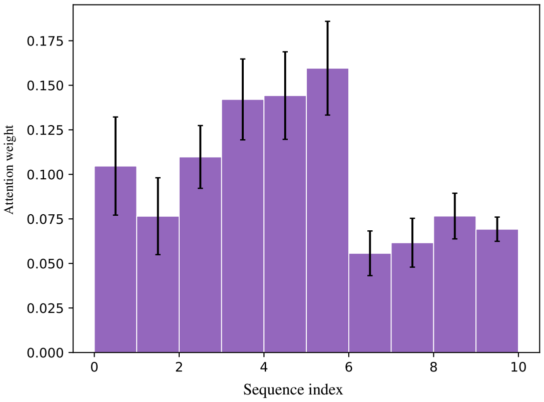

Figure 10: Second head

Figure 11: Mean attention for column $n = 10$ of the two heads of the first attention layer, for a 2-layer 2-head transformer model trained on an order-2 Markov process, averaged across 100 input sequences of length 128.

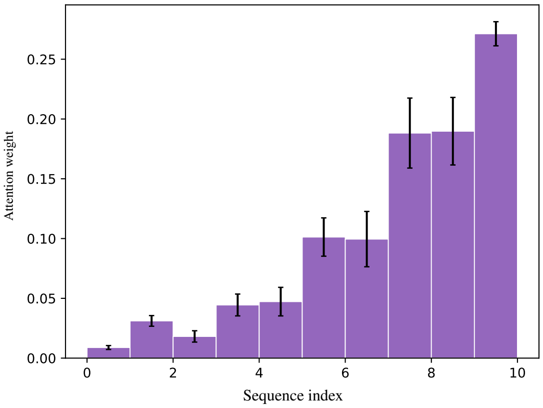

Figure 12: First head

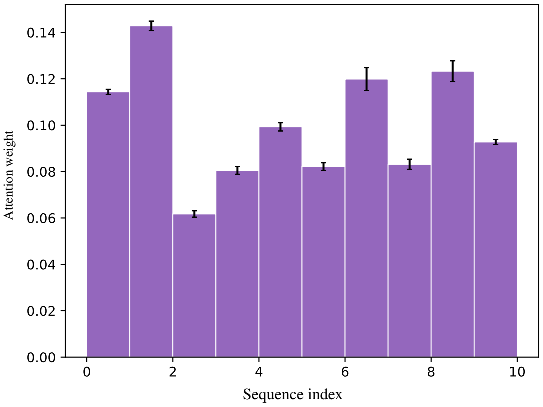

Figure 13: Second head

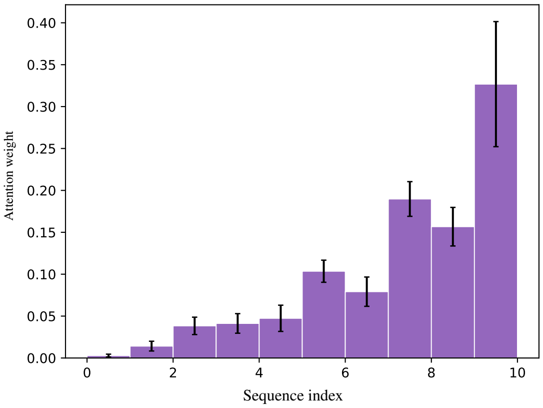

Figure 14: Third head

Figure 15: Mean attention for column $n = 10$ of the three heads of the first attention layer, for a 2-layer 3-head transformer model trained on an order-3 Markov process, averaged across 100 input sequences of length 128.

Assumption 6.2 suggests that the attention patterns $\mathrm{att}_{n,i}^{(\ell)}$ in layers $\ell = 1, 2, \cdots, L-1$, as learnt by an L-layer attention-only transformers may only be a function of only the position indices n, i. In this section we run some additional experiments to test this conjecture. We train a 2 layer attention-only transformer with k heads in the first layer, on data drawn from a randomly sampled k^{th}-order Markov process, and focus on the learnt attention patterns as a function of in the input sequence. Figure 11 plots the results of this experiment for $k = 2$ and Figure 15 for $k = 3$. While in both cases there is some variance in the attention patterns learnt by the transformer in some of the heads, we believe that this is a consequence of the iteration budget of the transformer, and specifically the fact that even if the test loss appears to have converged, the transformer may still continue changing in the parameter space. Furthermore, when the attention patterns have some non-zero but small variance as a function of the input, a relaxation of Assumption 6.2, we also believe that the results we proved in Theorems 6.3 to 6.5 should carry over approximately and leave this as an interesting question for future work. Conditional lower bounds of this nature, reliant on structural assumptions the transformer appears to demonstrate in practice are an interesting area of future research.