Parity, Sensitivity, and Transformers

Alexander Kozachinskiy
CENIA
alexander.kozachinskyi@cenia.cl

Tomasz Steifer
IPPT PAN
tsteifer@ippt.pan.pl

Przemysław Wałęga
Queen Mary University of London
p.walega@qmul.ac.uk

February 5, 2026

Abstract

The transformer architecture is almost a decade old. Despite that, we still have a limited understanding of what this architecture can or cannot compute. For instance, can a 1-layer transformer solve PARITY—or more generally—which kinds of transformers can do it? Known constructions for PARITY have at least 2 layers and employ impractical features: either a length-dependent positional encoding, or hardmax, or layernorm without the regularization parameter, or they are not implementable with causal masking.

We give a new construction of a transformer for PARITY with softmax, length-independent and polynomially bounded positional encoding, no layernorm, working both with and without causal masking. We also give the first lower bound for transformers solving PARITY— by showing that it cannot be done with only one layer and one head.

1 Introduction

Imagine that you are trying to train a neural network—or a related model such as transformer, Graph Neural Network etc.—on some task and you see that the accuracy oscillates and never reach a reasonable threshold. Perhaps the model is too big and overfits to the training dataset. Perhaps the model size is alright, but a rugged loss landscape makes it impossible to find the global minimum using gradient descent methods. But it may also be the case that model is not expressible enough—there is simply no assignment of the model weights that would allow it to compute the underlying true hypothesis. Such a scenario is the point of interest of the expressivity of neural networks (and related models), which is an important area of machine learning theory.

For instance, take a simple feed-forward neural network (FFN). It is well known that any Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ can be computed by a sufficiently large FFN with an appropriate activation function. However, a naive construction of requires the size of the network to grow exponentially with the length of the input. Clearly, this is not tight for all Boolean functions. Understanding which model properties (like size or specific activation function) are required to express a given function can guide model design and help explain failure in training.

In this paper we study expressivity of transformers with respect to one specific task, namely, the PARITY task. Let us define PARITY as a function which assigns 0 to all binary words with an even number of ones and 1 to all the rest. PARITY in an exemplary task that has been studied both in theoretical computer science (e.g. in the context of circuit complexity (Furst et al., 1984; Khrapchenko, 1971)) and in machine learning theory (see e.g. Regev (2009)). One of the reasons why PARITY got considerable attention is because, in a sense, it is the most sensitive Boolean function—flipping a single bit always changes the output of the function.

In the context of transformers, PARITY was also one of the first formal task which expressivity has been studied. Hahn (2020) considered a simplified model of transformer with unique hard attention (UHAT) and used random restriction method to prove that PARITY is not computable by a constant-depth UHAT transformer. Later Hao et al. (2022) strenghten this result by showing that any family of functions computable by a constant-depth UHAT transformer is in the class AC^0—which is known to not contain PARITY. Although concerned with a simplified model, these results suggested some explanation to why transformers struggle to generalize on tasks such as PARITY Bhattamishra et al. (2020).

Meanwhile, Chiang and Cholak (2022) gave an explicit construction of a 2-layer transformer with soft attention capable of computing PARITY. This result showed a stark contrast between hard and soft attention mechanisms, albeit with a caveat. The construction of Chiang and Cholak required a positional encoding $i \mapsto i/n$ that depends not only on the position i but also on the input length n. This differs from the standard approaches because transformers are usually run on inputs of varying depth. In fact, in NLP transformers often employ causal masking, where the i-th position can only access information from the positions with smaller indices (and so, most positions do not 'see' the input length).

This motivated Kozachinskiy and Steifer (2025) to give an alternative 3-layer construction based on a length-independent positional encoding. However, their construction works just in the the full-attention model and crucially requires that each token accesses the information from all the other positions. Moreover, they gave no bound on how fast their positional encoding has to grow.

Yang et al. (2025) gave yet another 2-layer construction that works with causal masking and no positional encoding (except a specialized beginning-of-sequence token). That being said, this construction required two features that differ from standard practice—hard attention and layer normalization with the regularization parameter ε set to zero. These design choices cannot be used in actual learning via standard gradient methods.

Interestingly, it remains open whether PARITY can be computed by a transformer in just one layer. All the constructions mentioned before use at least 2 layers. In general, proving a lower bound on the number of transformer layers needed to compute some task is hard, especially for the full-attention model (Chen et al., 2024) but some techniques for single layer transformers exist. Sanford et al. (2023) introduced a communication complexity technique (which works under the assumption of logarithmic precision). More recently, Kozachinskiy et al. (2025) gave a new technique, based on a notion of the Split-VC dimension, which works even for transformers with infinite precision. Both techniques require finding a partition of the input bits such that either (a) the communication complexity of the resulting problem, where Alice gets bits from one part of the partition and Bob from the other, is large; or (b) the VC dimension of the concept class, obtained by considering bits from one part of the partition as inputs and from the other as parameters, is large. For PARITY, however, both quantities are constant for any partition, making existing technique not applicable to this function.

Our results. We give a partial solution to this open problem and show that no 1-layer transformer with only 1 head can solve PARITY. Our argument relies on the notion of average sensitivity of a function f, which is defined as the number of input positions such that flipping them changes the value of f, averaged over all input words of the given length. In particular, we show that a function computed by a transformer with a single layer and single head, has the average sensitivity of $O(\sqrt{n})$. We contrast this with the observation that the average sensitivity of the PARITY grows linearly with the input length.

Our second technical contribution is a new construction of 4-layer transformer that computes PARITY, which improves on the existing constructions in the following aspects: it uses soft attention, works for both full-attention and causal masking architecture, uses only length-independent positional encoding and no layer normalization. Furthermore, its positional encoding is polynomially bounded, which makes it possible to implement on input lengths of practical size.

Related work. The expressive power of the transformer architectures (Vaswani et al., 2017) is heavily studied in the literature. Existing results analyse a wide range of settings leading to a rich landscape of expressiveness results. Some of the main architectural choices include different positional encodings; soft

attention or its idealizations: unique hard attention (UHAT) and averaging hard attention (AHAT); number of layers; the presence or absence of causal masking; the use or omission of layer normalization; finite versus infinite numerical precision; architectures with both encoder and decoder components or only one of them; and, more recently, the chain-of-thought setting.

Apart of the papers already mentioned, there exists a considerable collection of technical results on transformer expressivity. As already observed by Hao et al. (2022), transformers equipped with average hard attention (AHAT) can compute tasks outside AC^0 (such as MAJORITY), which shows a separation between different hard attention variants. Merrill et al. (2022) showed that AHAT transformers can solve only languages recognized by families of constant-depth threshold circuits (TC^0). Angluin et al. (2023) and Yang et al. (2024a) established a precise characterization showing that masked hard-attention transformers recognize exactly the star-free languages, which are equivalent to languages definable in first-order logic with linear order, linear temporal logic, and counter-free (aperiodic) automata. This connection to well-studied formal language classes provided deep insights into transformer limitations. A complementary result was obtained by Barcelo et al. (2023) who showed that UHAT cannot recognize some languages in AC^0, even with unbounded positional encoding. They also provided an upper bound, showing that such transformer can recognize any language definable in the first-order logic with arbitrary unary numerical predicates.

The exact relation between soft attention and its hard variants is not fully understood. For example, it is not known whether the soft attention can simulate UHAT or AHAT under the bounded precision assumption. Yang et al. (2024b) demonstrated that soft attention mechanisms can provably simulate hard attention if unbounded position encoding is allowed. This observation is significant as it suggests that some theoretical expressivity results for UHAT and AHAT transformers may transfer to more realistic architectures. For softmax attention models, the theoretical understanding is more limited and depends critically on precision assumptions. Merrill and Sabharwal (2023) proved that log-precision transformers have restricted expressivity, showing a fundamental trade-off between parallelism and precision. Chiang et al. (2023) proved tighter bounds on transformer expressivity by analysing the role of layer normalization and showing that even with polynomial precision, certain counting tasks remain difficult for bounded-depth transformers.

Sensitive functions in the context of transformers has been studied by Hahn and Rofin (2024). They have shown that, under certain assumptions, if a transformer computes a highly sensitive function, then the transformer itself is highly sensitive to random perturbation of its parameters. This partially explains why it is sometimes hard to reach a global minimum of the loss landscape when learning functions like parity.

The role of chain-of-thought setting significantly increases expressiveness of transformers. Wei et al. (2022) empirically demonstrated the effectiveness of chain-of-thought prompting. Liu et al. (2024) proved that that k chain-of-thought steps allow solving problems requiring k-fold function composition. Merrill and Sabharwal (2024) characterized the expressive power of AHAT transformers with chain-of-thought, showing that polynomial-length chains allow transformers to simulate polynomial-time computation under certain assumptions. Feng and Chen (2024) showed that transformers struggle with tasks requiring global coordination and proved that inductive scratchpad techniques can overcome some of these limitations. Recent work of Bavandpour et al. (2025) gave lower bounds on the number of chain-of-thought steps needed for UHAT transformers to solve certain tasks like PARITY. This complements the result of Barcelo et al. (2025) which gave exact characterization between the number of chain-of-thought steps needed for 1-layer UHAT transformer and Ehrenfeucht-Haussler tree rank.

Organization of the paper. Section 2 introduces all necesary definitions, related to transformers. In Section 3, the sensitivity lower bound on 1-layer 1-head transformers is established. Finally, Section 4 gives our new construction of a transformer for parity.

2 Transformers

Attention layers. Transformers are built upon attention layers. We consider two kinds of attention layers – full-attention and causually masked.

Definition 1. *A d-dimensional H-head full-attention layer is a function $L\colon (\mathbb{R}^d)^* \to (\mathbb{R}^d)^*$, given by*

- H query *matrices* $Q^{(k)} \in \mathbb{R}^{d\times d}$, H key *matrices* $K^{(k)} \in \mathbb{R}^{d\times d}$, *and* H value *matrices* $V^{(k)}$ *for* $k = 1,\ldots,H$,
- *a matrix* $W_O \in \mathbb{R}^{d\times(dH)}$ *(parameters of the linear transformation, mixing representations from different heads).*
- *two matrices* $W_1, W_2 \in \mathbb{R}^{d\times d}$ *and two vectors* $b_1, b_2 \in \mathbb{R}^d$ *(parameters of the position-wise feed-forward network in the end fo the layer).*

Given an input sequence of vectors $(\alpha_1,\ldots,\alpha_n) \in (\mathbb{R}^d)^n$, *the output sequence of vectors* $(\beta_1,\ldots,\beta_n) = L(\alpha_1,\ldots,\alpha_n) \in (\mathbb{R}^d)^n$ *is computed in the following steps:*

1. *for each* $k = 1,\ldots,H$, *and for* $i,j = 1,\ldots,n$, *compute the* ***attention weight*** *of the* j*-th position to the* i*-th position in the* k*-th head as:*

$$L_{ij}^{(k)} = \langle K^{(k)}\alpha_i, Q^{(k)}\alpha_j\rangle/\sqrt{d} \tag{1}$$

2. *for each head* $k = 1,\ldots,H$ *and position* $j = 1,\ldots,n$, *define the* ***value of the*** k***-th head in the*** j***-th position*** *as:*

$$h_j^{(k)} = \frac{\sum_{i=1}^{n}\exp\{L_{ij}\}V^{(k)}\alpha_i}{\sum_{i=1}^{n}\exp\{L_{ij}\}} \in \mathbb{R}^d; \tag{2}$$

3. *for each position* $j = 1,\ldots,n$, *combine the values of all the heads in this position via:*

$$h_j = W_O\begin{pmatrix} h_j^{(1)} \\ h_j^{(2)} \\ \vdots \\ h_j^{(H)} \end{pmatrix}; \tag{3}$$

4. *finally, for* $j = 1,\ldots,m$ *define the output in the* j*-th position as the result of applying a feed-forward network, given by matrices* W_1, W_2 *and bias vectors* b_1, b_2, *to* $h_j + \alpha_j$.

$$\beta_j = W_2 \cdot \mathrm{ReLU}\left(W_1(h_j+\alpha_j)+b_1\right)+b_2 \in \mathbb{R}^d. \tag{4}$$

Recall that $\mathrm{ReLU}((x_1,\ldots,x_d)) = (\max\{0,x_1\},\ldots,\max\{0,x_d\})$.

Causally masked attention layers are defined similarly, except that (2) is replaced by:

$$h_j^{(k)} = \frac{\sum_{i=1}^{j}\exp\{L_{ij}\}V^{(k)}\alpha_i}{\sum_{i=1}^{j}\exp\{L_{ij}\}} \in \mathbb{R}^d; \tag{5}$$

(the sum is only up to j, that is, the position j does not see positions ahead of it).

Transformers. In practice, transformers are defined as functions from sequences of *tokens* (elements of some finite alphabet) to probability distributions of tokens. Given a *prompt* (an input sequence of tokens), a transformer computes a distribution and then generates a token from it. We consider a deterministic version of transformers where instead of generation, we simply take as an output the token with the maximal probability.

Definition 2. *An C-layer H-head d-dimensional full-attention transformer over a finite set $\mathcal{V}$ (a "vocabulary" whose elements are called "tokens"), containing a special symbol $\perp$, is a function $T\colon \mathcal{V}^* \to \mathcal{V}$, given by*

- *C H-head d-dimensional full-attention layers $L_1, \ldots, L_C$;*
- *the input embedding $\mathsf{E}\colon \mathcal{V} \times \mathbb{N}^2 \to \mathbb{R}^d$;*
- *the* output distribution *matrix $W \in \mathbb{R}^{\mathcal{V} \times d}$ (transforming d-dimensional —vectors into vectors whose coordinates are indexed by tokens).*

Given a sequence of tokens $x_1 \ldots x_n \in \mathcal{V}^n$, the output token $y = T(x_1, \ldots, x_n)$ is computed in the following steps:

1. *we transform the input sequence of tokens into a sequence of vectors via the input embedding:*

$$\alpha_1 = \mathrm{E}(x_1, 1, n), \ldots, \alpha_n = \mathrm{E}(x_n, n, n); \tag{6}$$

2. *we apply the sequence of attention layers to this sequence:*

$$(\beta_1, \ldots, \beta_n) = L_C \circ \ldots \circ L_1(\alpha_1, \ldots, \alpha_n); \tag{7}$$

3. *we transform the vector in the last position after the last layer into a vector in $\mathbb{R}^{\mathcal{V}}$:*

$$\mu = \mathrm{softmax}(W\beta_n). \tag{8}$$

4. *finally, we define:*

$$y = T(x_1, \ldots, x_n) := \arg\max_{x \in \mathcal{V}} \mu_x. \tag{9}$$

If there are multiple tokens, reaching the maximum, we set $T(x_1, \ldots, x_n) = \perp$.

We say that the input embedding $\mathrm{E}\colon \mathcal{V} \times \mathbb{N}^2 \to \mathbb{R}^d$ is of the *standard form* if it can be written as $\mathrm{E}(x, i, N) = \mathrm{TE}(x) + \mathrm{PE}(i, n)$, for $x \in \mathcal{V}, i, n \in \mathbb{N}$ amd some functions $\mathrm{TE}\colon \mathcal{V} \to \{0,1\}^d$ (token embedding) and $\mathrm{PE}\colon \mathbb{N}^2 \to \mathbb{R}^d$ (positional encoding). The positional encoding function $\mathrm{PE}\colon \mathbb{N}^2 \to \mathbb{R}^d$ is length-independent if $\mathrm{PE}(i, n)$ depends just on the first argument, that is, if $\mathrm{PE}(i, n) = g(i)$ for some $g\colon \mathbb{N} \to \mathbb{R}^d$ and all $i, n \in \mathbb{N}$.

Definition 3. *A transformer T computes an infinite sequence of Boolean functions $\{f_n\}_{n \in \mathbb{N}}$, where $f_n\colon \{0,1\}^n \to \{0,1\}$ for each n, if the vocabulary of T includes symbols 0, 1, and if*

$$T(x) = f_n(x) \qquad \forall n \in \mathbb{N}\ \forall x \in \{0,1\}^n.$$

By PARITY we mean the sequence of functions $\{x_1 \oplus \ldots \oplus x_n\}_{n=1}^{\infty}$.

3 Sensitivity Lower Bound

Let $f\colon \{0,1\}^n \to \{0,1\}$. Its *sensitivity* at input $x \in \{0,1\}^n$, denoted by $s_x(f)$, is the number of input positions $i \in \{1,\ldots,n\}$ such that flipping x_i changes the value of $f(x)$. The average sensitivity of f is defined as:

$$as(f) = \sum_{x\in\{0,1\}^n} s_x(f)/2^n. \tag{10}$$

Theorem 1. *Assume that a sequence of Boolean functions $\{f_n\}_{n=1}^{\infty}$ is computable by a 1-layer 1-head transformer. Then $as(f_n) = O(\sqrt{n})$ as $n \to \infty$.*

Note that for 1-layer transformers, there is no difference between the full attention and causally-masked attention models. In the last position, where the output token is computed, we attend all positions in both models. And with just a single layer, the computation of attention in other positions is not affecting the result yet.

of Theorem 1. Let $f_n^0, f_n^1\colon \{0,1\}^{n-1} \to \{0,1\}$ be the results of two possible fixations of the last input bit to f_n:

$$f_n^0(x) = f_n(x0), \qquad f_n^1(x) = f_n(x1)$$

for $x \in \{0,1\}^{n-1}$. It is enough to show that $as(f_n^0) = O(\sqrt{n})$ and $as(f^1) = O(\sqrt{n}_n)$. Indeed, for $x \in \{0,1\}^{n-1}$, we have:

$$s_{x0}(f_n) \le s_x(f_n^0) + 1, \qquad s_{x1}(f_n) \le s_x(f_n^1) + 1,$$

and hence:

$$\begin{aligned} as(f_n) &= \sum_{x\in\{0,1\}^n} s_x(f_n)/2^n = \sum_{x\in\{0,1\}^{n-1}} s_{x0}(f_n)/2^n + \sum_{x\in\{0,1\}^{n-1}} s_{x1}(f_n)/2^n \\ &\le \sum_{x\in\{0,1\}^{n-1}} (s_x(f_n^0)+1)/2^n + \sum_{x\in\{0,1\}^{n-1}} (s_x(f_n^1)+1)/2^n \\ &= \frac{as(f_n^0) + as(f_n^1)}{2} + 1. \end{aligned}$$

We now show that $as(f_n^0) = O(\sqrt{n})$, the argument for the bound $as(f_n^1) = O(\sqrt{n})$ is similar. We take a 1-layer, 1-head transformer, computing $\{f_n\}$, assuming the last input bit x_n is fixed to 0. Let d be the dimension of this transformer.

Below we use a well-known fact that the theory of reals with addition and order $(\mathbb{R}, +, <)$ admits a quantifier elimination (Ferrante and Rackoff, 1975). We assume that the language contains all real constants (it still admits a quantifier elimination as we simply can replace all occurrences of constants by fresh free variables, eliminate quantifiers, and then substitute back constants in place of fresh free variables).

Lemma 1. *Assume there is a 1-layer 1-head transformer, computing $\{f_n\}_{n=1}^{\infty}$. There exists a quantifier-free formula Φ in the interpretation $(\mathbb{R}, +, <)$ with $d+1$ free variables such that for all n there exists $d+1$ affine (over $\mathbb{R}$) functions*

$$\begin{aligned} l_0(x_1,\ldots,x_{n-1}) &= c_0^1 x_1 + \ldots + c_0^{n-1} x_{n-1} + c_0, \\ &\vdots \\ l_d(x_1,\ldots,x_{n-1}) &= c_d^1 x_1 + \ldots + c_d^{n-1} x_{n-1} + c_d, \end{aligned}$$

such that for any $x \in \{0,1\}^{n-1}$, we have $f_n^0(x) = 1$ if and only if $\Phi(l_0(x),\ldots,l_d(x)) = 1$.

Proof. Let $h_n = W_O h_n^{(1)}$ and $\alpha_n = \mathrm{E}(0,n,n)$ be as in (1–4) (we have $h_n^{(k)}$ just for $k = 1$ because our transformer has just 1 head). Denote $\gamma_n = h_n + \alpha_n \in \mathbb{R}^d$. Observe that h_n depends on the input $x \in \{0,1\}^{n-1}$ while α_n does not (as the n-th token is fixed to 0). We first show that there exists a quantifier-free formula Ψ

in the interpretation $(\mathbb{R}, +, <)$ with d free variables such that for all n and $x \in \{0,1\}^{n-1}$, we have $\Psi(\gamma_n) = 1$ if and only if $f_n^0(x) = 1$.

The output of our attention layer at the n-th position is computed as:

$$\beta_n = W_2 \cdot \mathrm{ReLU}\left(W_1 \gamma_n + b_1\right) + b_2. \tag{11}$$

Here W_1, W_2, b_1, b_2 are two fixed $d \times d$ matrices and two fixed d-dimensional vectors. Note that an equality $\mathrm{ReLU}(x) = y$ is expressible in $(\mathbb{R}, +, <)$ via:

$$(x < 0 \to y = 0) \wedge (x \geq 0 \to y = x).$$

Hence, (11) where components of β_n and γ_n are treated as variables, is expressible in $(\mathbb{R}, +, <)$. Using this, we define Ψ as follows. We write $\exists(\beta_n)_1 \exists(\beta_n)_2 \ldots \exists(\beta_n)_d$ such that the formula, expressing equality (11), is true. We also add a condition that in the output distribution of our transformer, obtain from the vector $\beta_n = ((\beta_n)_1, \ldots, (\beta_n)_d))$, the token 1 has the maximal probability. This can be written as linear inequalities of the form $(W\beta_n)_1 > (W\beta_n)_x$ for $x \in \mathcal{V} \setminus \{1\}$, where $W \in \mathbb{R}^{\mathcal{V} \times d}$ is the output distribution matrix of our transformer. Finally, we eliminate quantifiers to obtain the required formula Ψ.

We now turn Ψ into Φ with the properties, stated in the lemma. The vector $\gamma_n 0$ expresses as follows:

$$\gamma_n = W_O h_n^{(1)} + \alpha_n = \frac{\sum\limits_{i=1}^{n} \exp\{L_{in}\} \left(\alpha_n + W_O V^{(1)} \alpha_i\right)}{\sum\limits_{i=1}^{n} \exp\{L_{in}\}}$$

The term in the numerator, corresponding to $i = n$, is a vector $\theta_n = \exp\{L_{nn}\} \left(\alpha_n + W_O V^{(1)} \alpha_n\right) \in \mathbb{R}^d$, not depending on $x \in \{0,1\}^{n-1}$ (recall that the input in the last token is fixed to 0). Likewise, the term in the denominator for $i = n$ is a number $\rho_n \in \mathbb{R}$, not depending on $x \in \{0,1\}^{n-1}$. In turn, for every $i = 1, \ldots, n-1$, the i-th term in the sum of the denominator is a vector:

$$\begin{aligned}
&\exp\{L_{in}\} \left(\alpha_n + W_O V^{(1)} \alpha_i\right) = \exp\{\langle K^{(1)} \alpha_i, Q^{(1)} \alpha_n \rangle\} \left(\alpha_n + W_O V^{(1)} \alpha_i\right) \\
&= \exp\{\langle K^{(1)} \cdot \mathrm{E}(x_i, i, n), Q^{(1)} \cdot \mathrm{E}(0, n, n) \rangle\} \left(\mathrm{E}(0, n, n) + W_O V^{(1)} \mathrm{E}(x_i, i, n)\right),
\end{aligned}$$

determined just by i, n and the input bit x_i. Hence, it can be written as $(1 - x_i)\theta_{in}^0 + x_i \theta_{in}^1$ for some $\theta_{in}^0, \theta_{in}^1 \in \mathbb{R}^d$. Similarly, the i-th term of the sum in the denominator, for $i = 1, \ldots, n-1$, can be written as $(1 - x_i)\rho_{in}^0 + x_i \rho_{in}^1$ for some $\rho_{in}^0, \rho_{in}^1 \in \mathbb{R}$. Overall, we obtain:

$$\gamma_n = \frac{\theta_n + \sum\limits_{i=1}^{n-1} \left((1 - x_i)\theta_{in}^0 + x_i \theta_{in}^1\right)}{\rho_n + \sum\limits_{i=1}^{n-1} \left((1 - x_i)\rho_{in}^0 + x_i \rho_{in}^1\right)} = \begin{pmatrix} l_1(x) \\ l_2(x) \\ \vdots \\ l_d(x) \end{pmatrix} / l_0(x),$$

where $l_0, l_1, \ldots, l_d$ are some affine functions in $x = (x_1, \ldots, x_{n-1})$. To obtain Φ, we introduce $d+1$ variables $\tau_0, \ldots, \tau_d$, and replace $(\gamma_n)_i$ with τ_i / τ_0 for $i = 1, \ldots, d$ in Ψ. All atomic formulas in Ψ will be linear inequalities/equalities of the form:

$$c_1 \tau_1 / \tau_0 + \ldots + c_d \tau_d / \tau_0 \geq / = c_0.$$

To obtain Φ, we multiply all these atomic formulas by τ_0, obtaining linear expressions of the form:

$$c_1 \tau_1 + \ldots + c_d \tau_d \geq / = c_0 \tau_0.$$

Note that we are only care about the cases when τ_0 is strictly positive, because in the statement of the lemma, we substitute τ_0 with $l_0 = \sum\limits_{i=1}^{n} \exp\{L_{in}\}$, taking strictly positive value for every $x \in \{0,1\}^{n-1}$. Hence, under for this kind of substitutions, the multiplication by τ_0 gives an equivalent formula. $\square$

We finish the proof of the theorem. Let us call an edge of the $(n-1)$-dimensional Boolean hypercube *sensitive* if f_n^0 takes different values on its ends. The average sensitivity of f_n^0 is twice the number of sensitive edges, divided by 2^n (the factor of 2 appears because in the formula for average sensitivity (10), every sensitive edge is counted twice). In turn, the value of f_n^0 is determined by the value of $\Phi(l_0(x), \ldots, l_d(x))$. The formula Φ is quantifier-free, and there is some constant (independent on n) number of atomic sub-formulas. Each of these atomic sub-formulas becomes a linear equality or inequality in $x_1, \ldots, x_{n-1}$ when we substitute linear functions $l_0(x), \ldots, l_d(x)$ in place of fresh variables of Φ. In order for an edge to be sensitive for f_n^0, one of these $O(1)$ equalities or inequalities has to give different results on the ends of this edge. Geometrically, this edge has to be cut by the hyperplane, defining a linear equality or inequality. It remains to use a result of O'Neil (1971) that a hyperplane can cut at most $O(\sqrt{n}2^n)$ edges of a hypercube.

We still require some clarification because O'Neil assumes that a hyperplane cuts an edge if goes strictly between its endpoints, while in our setting, an edge can be sensitive also if a hyperplane passes through one of the endpoints (but not through the other). However, one can easily deduce from the O'Neil's result that a hyperplane can cut at most $O(\sqrt{n}2^n)$ edges including non-strict cutting. Indeed, we can move a hyperplane slightly towards a hyperspace with a at least half of non-strictly cut edges. The resulting hyperplane will strictly cut at least half of the edges that were cut (possibly, non-stricty) by the initial hyperplane. □

Corollary 1. *No 1-layer 1-head transformer computes PARITY.*

The upper bound on sensitivity in Theorem 1 is tight as there exists a sequence of Boolean functions, having average sensitivity $\Omega(\sqrt{n})$ and computable by a 1-layer 1-head transformer. For instance, this holds for the sequence of *majority functions*, $\{\mathrm{maj}_n\}_{n\in\mathbb{N}}$, where

$$\mathrm{maj}_n(x_1, \ldots, x_n) = \begin{cases} 1 & x_1 + \ldots + x_n > n/2, \\ 0 & \text{otherwise.} \end{cases}$$

Almost all inputs to maj_n have 0 sensitivity, except of $\Omega(2^n/\sqrt{n})$ inputs from 2 adjacent layers of the Boolean cube (where maj_n changes its value) that all have sensitivity $\Omega(n)$. This implies that $as(\mathrm{maj}_n) = \Omega(\sqrt{n})$. On the other hand, a 1-layer 1-head transformer is able to compute this function by computing the expression: $\frac{x_1+\ldots+x_n}{n} - \frac{1}{2n} - \frac{1}{2}$, where the first term comes from the average of the input bits with uniform attention weights, and the second term comes from the positional encoding. This quantity is positive for inputs with value of maj_n equal to 1, and negative for inputs with value of maj_n equal to 0. It remains to put this quantity in the output distribution to the token 1, and minus this quantity to the token 0.

4 A New Transformer for Parity

We require the following fact (a generalization of Faulhaber's formulas to real powers), proved in Appendix for completeness.

Lemma 2. *For $\alpha \in [5, 100]$, and $n \in \mathbb{N}$, we have $1^\alpha + \ldots + n^\alpha = \frac{n^{\alpha+1}}{\alpha+1} + \frac{n^\alpha}{2} + \frac{\alpha n^{\alpha-1}}{12} + O(n^{\alpha-2})$.*

We now establish our main result of this section.

Theorem 2. *Both in the full attention and the causally-masked attention models, there is a 4-layer transformer for PARITY with a standard-form input embedding, whose positional encoding is length-independent and polynomially bounded (the latter meaning that the l_∞-norm of the positional encoding in position i is bounded by some polynomial in i).*

Proof. We will need to compute attention just in the last token, from the rest of the tokens we need just positional encoding and input bits. Thus, our construction will work both in the full attention and causally-masked attention models.

Let $x_1 x_2 \ldots x_n \in \{0,1\}^n$ be the input word and let $\Sigma = x_1 + \ldots + x_n$. Let us give a proof assuming that $1 \le \Sigma \le cn$ for some universal constant $c > 0$ to be defined later. Under this assumption, we will require just 3 layers.

Our general plan is to use attention to obtain the following value at some point:

$$z = \sum_{i=1}^{n} e^{L_{i,n}}(-1)^i / \left(\sum_{i=1}^{n} e^{L_{i,n}} \right),$$

and a guarantee that a) $L_{i,n}$ is maximized at the position $i = \Sigma$; b) $L_{\Sigma,n}$ is much larger than $L_{j,n}$ whenever $j \neq \Sigma$—larger enough to guarantee that z is positive if Σ is even and negative otherwise.

At the first layer, we compute the weighted sum of inputs bits, where the weight of the positions with 1 is 1, and the weight of the positions with 0 is α/n, for some constant $\alpha \in (0, 1)$ to be specified later. Indeed, we can have $\ln n$ at position n from the positional encoding. Thus, we can get attention weights of the form $L_{i,n} = (-\ln(n) + \delta)(1 - x_i)$, where δ is such that $e^\delta = \alpha$. This will allow us to compute the following expression in the first layer:

$$\gamma = \frac{10\Sigma}{\Sigma + (\alpha/n)(n - \Sigma)} = \frac{10}{1 + \rho}, \qquad \text{where } \rho = \alpha(1/\Sigma - 1/n).$$

Note that $0 \leq \rho \leq \alpha < 1$, meaning that $5 \leq \gamma \leq 10$

At the second layer, using the positional encoding $i \mapsto (\ln i, i^{10})$, and the already computed value of γ, we can compute the following expression:

$$\Gamma = \frac{1^\gamma \cdot 1^{10} + \ldots + n^\gamma \cdot n^{10}}{1^\gamma + \ldots + n^\gamma}$$

(using attention weights $L_{i,n} = \ln(i) \cdot \gamma$).

Lemma 3.

$$\Gamma = \tau_n \cdot f(\rho) \cdot \left(1 + O\left(\frac{\rho}{n^2} + \frac{1}{n^3}\right)\right),$$

where $\tau_n = n^{10}\left(1 + \frac{5}{n} - \frac{5}{3n^2}\right)$ *and* $f(\rho) = \frac{11+\rho}{21+11\cdot\rho}$.

Proof. Elaborating on the expression for Γ with the use of Lemma 2, since $\gamma, \gamma + 11 \in [5, 100]$, we get:

$$\begin{aligned}
\Gamma &= \frac{\frac{n^{\gamma+11}}{\gamma+11}\left(1 + \frac{\gamma+11}{2n} + \frac{(\gamma+11)(\gamma+10)}{12n^2} + O\left(\frac{1}{n^3}\right)\right)}{\frac{n^{\gamma+1}}{\gamma+1}\left(1 + \frac{\gamma+1}{2n} + \frac{(\gamma+1)\gamma}{12n^2} + O\left(\frac{1}{n^3}\right)\right)} \\
&= n^{10} \cdot \frac{\gamma+1}{\gamma+11} \cdot \frac{\left(1 + \frac{\gamma+11}{2n} + \frac{(\gamma+11)(\gamma+10)}{12n^2} + O\left(\frac{1}{n^3}\right)\right)}{\left(1 + \frac{\gamma+1}{2n} + \frac{(\gamma+1)\gamma}{12n^2} + O\left(\frac{1}{n^3}\right)\right)}.
\end{aligned}$$

Observe that $\frac{\gamma+1}{\gamma+11} = \frac{\frac{10}{1+\rho}+1}{\frac{10}{1+\rho}+11} = \frac{11+\rho}{21+11\rho} = f(\rho)$. Let us now work separately with the fraction in the last

expression.

$$\frac{\left(1+\frac{\gamma+11}{2n}+\frac{(\gamma+11)(\gamma+10)}{12n^2}+O\left(\frac{1}{n^3}\right)\right)}{\left(1+\frac{\gamma+1}{2n}+\frac{(\gamma+1)\gamma}{12n^2}+O\left(\frac{1}{n^3}\right)\right)}=\left(1+\frac{\gamma+11}{2n}+\frac{(\gamma+11)(\gamma+10)}{12n^2}+O\left(\frac{1}{n^3}\right)\right)\cdot$$
$$\left(1-\frac{\gamma+1}{2n}-\frac{(\gamma+1)\gamma}{12n^2}+\frac{(\gamma+1)^2}{4n^2}+O\left(\frac{1}{n^3}\right)\right)$$
$$=1+\frac{5}{n}+\frac{1}{12n^2}\Big((\gamma+11)(\gamma+10)-3(\gamma+1)(\gamma+11)-(\gamma+1)\gamma+3(\gamma+1)^2\Big)+O\left(\frac{1}{n^3}\right)$$
$$=1+\frac{5}{n}+\frac{80-10\gamma}{12n^2}+O\left(\frac{1}{n^3}\right)=1+\frac{5}{n}+\frac{80-\frac{100}{1+\rho}}{12n^2}+O\left(\frac{1}{n^3}\right)$$
$$=1+\frac{5}{n}+\frac{80-100+O(\rho)}{12n^2}+O\left(\frac{1}{n^3}\right)=1+\frac{5}{n}-\frac{5}{3n^2}+O\left(\frac{\rho}{n^2}+\frac{1}{n^3}\right)$$
$$=\left(1+\frac{5}{n}-\frac{5}{3n^2}\right)\left(1+O\left(\frac{\rho}{n^2}+\frac{1}{n^3}\right)\right),$$

and the lemma follows. □

Lemma 4. *Let $f(\rho)=\frac{11+\rho}{21+11\cdot\rho}$ be the function from Lemma 3. For $i\in\{1,\ldots,n\}$, define*

$$W_i=-\left(f(\rho)-f(0)-f'(0)\cdot\alpha(1/i-1/n)\right)^2.$$

There for all small enough $\alpha\in(0,1)$, for all n and $\Sigma\in\{1,2,\ldots,n\}$, we have:

- $W_\Sigma\geq -O(\alpha^4/\Sigma^4)$;
- $W_i\leq -\Omega\left(\alpha^2(1/i-1/\Sigma)^2\right)$ *for all* $i\neq\Sigma$.

Proof. The function $f(\rho)$ is infinitely differentiable at $(-1,+\infty)$, meaning that

$$f(\rho)=f(0)+f'(0)\rho+O(\rho^2)\qquad\text{as }\rho\to 0.$$

Importantly, $f'(0)\neq 0$ as a direct calculation shows that $f'(0)=-100/441$. Recall that $\rho=\alpha(1/\Sigma-1/n)\leq\alpha/\Sigma)$. Thus, $W_\Sigma=-(O(\rho^2))^2=-O(\alpha^4/\Sigma^4)$. In turn, for any $i\neq\Sigma$, we obtain:

$$-W_i=-\left(f'(0)(\frac{\alpha}{\Sigma}-\frac{\alpha}{i})+O(\alpha^2/\Sigma^2)\right)^2=-\left(\Omega(\alpha(1/i-1/\Sigma))+O(\alpha^2/\Sigma^2)\right)^2.$$

For small enough α, the term $\Omega(\alpha(1/i-1/\Sigma))=\Omega(\alpha/\Sigma^2)$ dominates the $O(\alpha^2/\Sigma^2)$ term, and the lemma follows. □

Our plan now is to devise, at the third layer, attention weights $L_{i,n}$ that are proportional to W_i, multiplied by a large factor (so that attention will be mostly concentrated at the position $i=\Sigma$).

Note that

$$W_i=-(f(\rho)+C/i+A_n)^2$$

for some absolute constant C and some expression A_n, depending only on n. Elaborating on this further, we get:

$$W_i=-2f(\rho)\cdot(C/i)-(C/i)^2-2A_n\cdot(C/i)+B_{n,\rho}$$

for some expression $B_{n,\rho}$ that does not depend on i.

We will get attention weights that are very close to this expression without $B_{n,\rho}$, multiplied by the factor $\tau_n=n^{10}\left(1+\frac{5}{n}-\frac{5}{3n^2}\right)$ from Lemma 3:

$$L'_{i,n}=\tau_n(W_i-B_{n,\rho})=\tau_n(-2f(\rho)\cdot(C/i)-(C/i)^2-2A_n\cdot(C/i))$$
$$=-2\tau_n\cdot f(\rho)\cdot(C/i)-\tau_n(C/i)^2-2\tau_nA_n\cdot(C/i).$$

Will not be able to get attention weights exactly $L'_{i,n}$ in the dot-product attention. The problem is with the term $\tau_n f(\rho)$ which is not yet computed. However, at the n-th position we have computed Γ, which, by Lemma 3, satisfies $\Gamma = \tau_n \cdot f(\rho) \cdot (1 + O(\frac{\rho}{n^2} + \frac{1}{n^3}))$ and thus is really close to $\tau_n \cdot f(\rho)$. In turn, there will be no problem with terms $\tau_n(C/i)^2$ and $2\tau_n A_n \cdot (C/i)$ as these can be obtained from the dot-product attention using the positional encoding $i \mapsto (1/i, 1/i^2, \tau_i, \tau_i A_i)$. That is, our attention weights at the third layer will be:

$$L_{i,n} = -2\Gamma \cdot (C/i) - \tau_n (C/i)^2 - 2\tau_n A_n \cdot (C/i).$$

Recall that we assume that $\Sigma \leq cn$ for some absolute constant $C > 0$ to be chosen later. To finish the proof, we just need to show

Lemma 5. *There exist $\alpha > 0, c > 0$ such that for all large enough n and all Σ, we have that $L_{\Sigma,n} \geq L_{i,n} + \Omega(n^6)$ for all $i \neq \Sigma$.*

Proof. Note that

$$\begin{aligned} L_{i,n} &= -2\tau_n \cdot f(\rho)(1 + O(\rho/n^2 + 1/n^3)) \cdot (C/i) - \tau_n (C/i)^2 - 2\tau_n A_n \cdot (C/i) \\ &= L'_{i,n} + O(\tau_n \cdot \left(\frac{\rho}{n^2 \cdot i} + \frac{1}{n^3 i} \right)) = \tau_n (W_i - B_{n,\rho}) + O(\tau_n \cdot \left(\frac{\rho}{n^2 \cdot i} + \frac{1}{n^3 i} \right)). \end{aligned}$$

By Lemma 4, for all $\alpha > 0$ small enough we get:

$$L_{\Sigma,n} \geq \tau_n B_{n,\rho} - O(\tau_n \alpha^4/\Sigma^4) + O(\tau_n \cdot \left(\frac{\rho}{n^2 \cdot \Sigma} + \frac{1}{n^3 \Sigma} \right)) = \tau_n B_{n,\rho} - O(\tau_n E_1) + O(\tau_n E_2), \tag{12}$$

$$L_{i,n} \leq \tau_n B_{n,\rho} - \Omega(\tau_n \frac{\alpha^2 (i-\Sigma)^2}{i^2 \Sigma^2}) + O(\tau_n \cdot \left(\frac{\rho}{n^2 \cdot i} + \frac{1}{n^3 i} \right)) = \tau_n B_{n,\rho} - \Omega(\tau_n E_3) + O(\tau_n E_4). \tag{13}$$

We show that by taking α, c to be small enough, we can make $E_3/E_1, E_3/E_2, E_3/E_4$ arbitrarily large. This will imply that $L_{\Sigma,n}$ is larger by at least $\Omega(\tau_n E_3) = \Omega((1/\Sigma - 1/i)^2 \tau_n) = \Omega(\tau_n/\Sigma^4) = \Omega(n^6)$, as required.

We first fix α so that E_3 is any given constant time larger than E_1. This is possible because E_3 is at least $\Omega(\alpha^2/\Sigma^4)$ while $E_1 = O(\alpha^4/\Sigma^4)$.

We now consider α as fixed. Then $E_3 = \Omega(1/\Sigma^4)$. In turn, $E_2 = O(\left(\frac{\rho}{n^2 \cdot \Sigma} + \frac{1}{n^3 \Sigma}\right)) = O(1/(n^2 \Sigma^2))$ (recall that $\rho \leq \alpha/\Sigma = O(1/\Sigma)$). Thus, $E_3/E_1 = \Omega(n^2/\Sigma^2)$. Choosing c in $\Sigma \leq cn$ small enough makes the fraction E_3/E_1 arbitrarily large.

Likewise, considering E_3/E_4, since $E_4 = O(\left(\frac{\rho}{n^2 \cdot i} + \frac{1}{n^3 i}\right)) = O(1/n^2 \Sigma i)$ as $\rho = O(1/\Sigma)$, we get up to a fixed constant factor:

$$E_3/E_4 \geq ((i-\Sigma)^2/(i^2\Sigma^2)/(1/(n^2 \Sigma i)) = (i-\Sigma)^2 \cdot \frac{n^2}{\Sigma i} \geq n/\Sigma,$$

where the latter is bacause $(i-\Sigma)^2 \geq 1, n/i \geq 1$. Again, by choosing c sufficiently small, we can make this fraction arbitrarily large. □

Hence, the maximum of $L_{i,n}$ is achieved at $i = \Sigma$, with all the other values being $\Omega(n^6)$ smaller. We then are able to compute the expression:

$$z = \sum_{i=1}^{n} e^{L_{i,n}} (-1)^i / \left(\sum_{i=1}^{n} e^{L_{i,n}} \right),$$

which will be, say, 0.1-close to $(-1)^\Sigma$. In particular, it will be positive for even Σ and negative for odd Σ. Thus, in the output distribution, it remains to put value z to the token 0, and value $-z$ to the token 1.

Finally, we explain how to get rid of the assumption $0 < \Sigma \leq cn$ for some small constant $c > 0$. We take an even integral number $M > 2/c$. Given an input $x \in \{0,1\}^n$, in the first layer we compute strings

$x^0, \ldots, x^{M-1}$, where x^r coincides with x on positions i with $i \equiv r \pmod M$ and is equal to 0 elsewhere, except of the position $r+1$ where it has 1. Thus, we have the following expressions for the bits of x^r:

$$x^r_i = \mathrm{ReLU}\Big(x_i + \mathbb{I}\{i \equiv r \pmod M\} - 1\Big) + \mathbb{I}\{i = r+1\},$$

which can be computed via FFNs of the first layer (indicators can be taken from the positional encoding). Note that $PARITY(x) = PARITY(x^0) \oplus \ldots \oplus PARITY(x^{M-1})$ because M is even. Moreover, for each $r = 0, \ldots, M-1$, the number of 1s in x^r is at least 1 and at most $1 + n/M < cn$. Hence, in the next 3 layers we can compute the parities of $x^0, \ldots, x^{M-1}$ in parallel, using M attention heads and the construction above. More precisely, we can compute M numbers $z^0, \ldots, z^{M-1}$ such that z^r is ϵ-close to 1 if $PARITY(x^r) = 0$, and ϵ-close to -1 if $PARITY(x^r) = 1$ (here $\epsilon > 0$ can be made arbitrarily small if n is large enough). Thus, the parity of x will be 0 if and only if there is an even number of numbers close to -1 among $z^0, \ldots, z^{M-1}$. In the FFN of the final layer, it now suffices to sum up expressions of the form $\mathrm{ReLU}(-z^0 - \ldots - z^{M-1} - M + 0.1)$ and all the similar ones where the number of minuses before z^r's is even. This sum will be at least some positive constant if the parity of x is 0, and 0 otherwise. □

References

Angluin, D., Chiang, D., and Yang, A. (2023). Masked hard-attention transformers and Boolean RASP recognize exactly the star-free languages. *CoRR*, abs/2310.13897.

Barcelo, P., Kozachinskiy, A., Lin, A. W., and Podolskii, V. (2023). Logical languages accepted by transformer encoders with hard attention. In *The Twelfth International Conference on Learning Representations*.

Barcelo, P., Kozachinskiy, A., and Steifer, T. (2025). Ehrenfeucht-haussler rank and chain of thought. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pages 2968–2977. PMLR.

Bavandpour, A. A., Huang, X., Rofin, M., and Hahn, M. (2025). Lower bounds for chain-of-thought reasoning in hard-attention transformers. In *Forty-second International Conference on Machine Learning*.

Bhattamishra, S., Ahuja, K., and Goyal, N. (2020). On the ability and limitations of transformers to recognize formal languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7096–7116.

Chen, L., Peng, B., and Wu, H. (2024). Theoretical limitations of multi-layer transformer. *arXiv preprint arXiv:2412.02975*.

Chiang, D. and Cholak, P. (2022). Overcoming a theoretical limitation of self-attention. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Chiang, D., Cholak, P., and Pillay, A. (2023). Tighter bounds on the expressivity of transformer encoders. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pages 5544–5562.

Feng, G. and Chen, Y. (2024). How far can transformers reason? the globality barrier and inductive scratchpad. In *NeurIPS 2024*.

Ferrante, J. and Rackoff, C. (1975). A decision procedure for the first order theory of real addition with order. *SIAM Journal on Computing*, 4(1):69–76.

Furst, M., Saxe, J. B., and Sipser, M. (1984). Parity, circuits, and the polynomial-time hierarchy. *Mathematical systems theory*, 17(1):13–27.

Hahn, M. (2020). Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171.

Hahn, M. and Rofin, M. (2024). Why are sensitive functions hard for transformers? *arXiv preprint arXiv:2402.09963*.

Hao, Y., Angluin, D., and Frank, R. (2022). Formal language recognition by hard attention transformers: Perspectives from circuit complexity. *Transactions of the Association for Computational Linguistics*, 10:800–810.

Khrapchenko, V. M. (1971). Complexity of the realization of a linear function in the class of π-circuits. *Matematicheskie Zametki*, 9(1):35–40.

Kozachinskiy, A. and Steifer, T. (2025). A completely uniform transformer for parity. *arXiv preprint arXiv:2501.02535*.

Kozachinskiy, A., Urrutia, F., Jimenez, H., Steifer, T., et al. (2025). Strassen attention: Unlocking compositional abilities in transformers based on a new lower bound method. *arXiv preprint arXiv:2501.19215*.

Liu, Z., Wang, H., and Ma, T. (2024). Chain of thought empowers transformers to solve inherently serial problems. In *ICLR 2024*.

Merrill, W. and Sabharwal, A. (2023). The parallelism tradeoff: Limitations of log-precision transformers. *Transactions of the Association for Computational Linguistics*, 11:531–545.

Merrill, W. and Sabharwal, A. (2024). The expressive power of transformers with chain of thought. *arXiv preprint arXiv:2310.07923*.

Merrill, W., Sabharwal, A., and Smith, N. A. (2022). Saturated transformers are constant-depth threshold circuits. *Transactions of the Association for Computational Linguistics*, 10:843–856.

O'Neil, P. E. (1971). Hyperplane cuts of an n-cube. *Discrete Mathematics*, 1(2):193–195.

Regev, O. (2009). On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40.

Sanford, C., Hsu, D. J., and Telgarsky, M. (2023). Representational strengths and limitations of transformers. *Advances in Neural Information Processing Systems*, 36:36677–36707.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wei, J. et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS 2022*.

Yang, A., Chiang, D., and Angluin, D. (2024a). Masked hard-attention transformers recognize exactly the star-free languages. In *NeurIPS 2024*.

Yang, A., Strobl, L., Chiang, D., and Angluin, D. (2024b). Simulating hard attention using soft attention. *arXiv preprint arXiv:2412.09925*.

Yang, A., Watson, C., Xue, A., Bhattamishra, S., Llarena, J., Merrill, W., Ferreira, E. D. S., Svete, A., and Chiang, D. (2025). The transformer cookbook. *arXiv preprint arXiv:2510.00368*.

A Proof of Lemma 6

The proof is via a series of lemmas.

Lemma 6. *For $\alpha \in [0, 100]$, and $n \in \mathbb{N}$, we have:*

$$1^\alpha + \ldots + n^\alpha = \frac{n^{\alpha+1}}{\alpha+1} + O(n^\alpha).$$

Proof. Observe that:

$$\frac{n^{\alpha+1}}{\alpha+1} = \int_0^n x^\alpha dx \leq 1^\alpha + \ldots + n^\alpha \leq \int_1^{n+1} x^\alpha dx \leq \frac{(n+1)^{\alpha+1}}{\alpha+1}$$

(using monotonicity of the function under integral since $\alpha \geq 0$). It remains to observe that

$$(n+1)^{\alpha+1} - n^{\alpha+1} = n^{\alpha+1}\left(\left(1+\frac{1}{n}\right)^{\alpha+1} - 1\right) = n^{\alpha+1} \cdot \left(\frac{\alpha+1}{n} + O(1/n^2)\right) = O(n^\alpha).$$

□

Lemma 7. *For $\alpha \in [2, 100]$, and $n \in \mathbb{N}$, we have:*

$$1^\alpha + \ldots + n^\alpha = \frac{n^{\alpha+1}}{\alpha+1} + \frac{n^\alpha}{2} + O(n^{\alpha-1}).$$

Proof. Observe that:

$$\begin{aligned}\frac{n^{\alpha+1}}{\alpha+1} &= \sum_{i=1}^n \frac{i^{\alpha+1} - (i-1)^{\alpha+1}}{\alpha+1} = \sum_{i=1}^n \frac{i^{\alpha+1} \cdot \left(1 - \left(1 - \frac{1}{i}\right)^{\alpha+1}\right)}{\alpha+1} \\ &= \sum_{i=1}^n \frac{i^{\alpha+1}\left(\frac{\alpha+1}{i} - \frac{(\alpha+1)\alpha}{2i^2} + O\left(\frac{1}{i^3}\right)\right)}{\alpha+1} = \sum_{i=1}^n i^\alpha - \frac{\alpha}{2}\sum_{i=1}^n i^{\alpha-1} + O\left(\sum_{i=1}^n i^{\alpha-2}\right)\end{aligned}$$

Using Lemma 6 for $\alpha - 1$ and $\alpha - 2$, we get:

$$1^\alpha + \ldots + n^\alpha = \frac{n^{\alpha+1}}{\alpha+1} + \frac{n^\alpha}{2} + O(n^{\alpha-1}),$$

as required. □

We finally get to the proof of Lemma 2. Similarly to the previous proof, we get:

$$\begin{aligned}\frac{n^{\alpha+1}}{\alpha+1} &= \sum_{i=1}^n \frac{i^{\alpha+1} \cdot \left(1 - \left(1 - \frac{1}{i}\right)^{\alpha+1}\right)}{\alpha+1} \\ &= \sum_{i=1}^n \frac{i^{\alpha+1} \cdot \left(\frac{\alpha+1}{i} - \frac{(\alpha+1)\alpha}{2i^2} + \frac{(\alpha+1)\alpha(\alpha-1)}{6i^3} + O\left(\frac{1}{i^4}\right)\right)}{\alpha+1} \\ &= S_\alpha - \frac{\alpha}{2}S_{\alpha-1} + \frac{\alpha(\alpha-1)}{6}S_{\alpha-2} + O(S_{\alpha-3}),\end{aligned}$$

where $S_\beta = 1^\beta + \ldots + n^\beta$. Using previous lemmas, we get:

$$\begin{aligned} S_\alpha &= \frac{n^{\alpha+1}}{\alpha+1} + \frac{\alpha}{2} S_{\alpha-1} - \frac{\alpha(\alpha-1)}{6} S_{\alpha-2} + O(S_{\alpha-3}) \\ &= \frac{n^{\alpha+1}}{\alpha+1} + \frac{\alpha}{2}\left(n^\alpha/\alpha + n^{\alpha-1}/2 + O(n^{\alpha-2})\right) \\ &- \frac{\alpha(\alpha-1)}{6}(n^{\alpha-1}/(\alpha-1) + O(n^{\alpha-2})) + O(n^{\alpha-2}) \\ &= \frac{n^{\alpha+1}}{\alpha+1} + \frac{n^\alpha}{2} + \frac{\alpha n^{\alpha-1}}{12} + O(n^{\alpha-2}). \end{aligned}$$