Theoretical Limitations of Self-Attention in Neural Sequence Models

Michael Hahn
Stanford University
mhahn2@stanford.edu

Abstract

Transformers are emerging as the new workhorse of NLP, showing great success across tasks. Unlike LSTMs, transformers process input sequences entirely through self-attention. Previous work has suggested that the computational capabilities of self-attention to process hierarchical structures are limited. In this work, we mathematically investigate the computational power of self-attention to model formal languages. Across both soft and hard attention, we show strong theoretical limitations of the computational abilities of self-attention, finding that it cannot model periodic finite-state languages, nor hierarchical structure, unless the number of layers or heads increases with input length. These limitations seem surprising given the practical success of self-attention and the prominent role assigned to hierarchical structure in linguistics, suggesting that natural language can be approximated well with models that are too weak for the formal languages typically assumed in theoretical linguistics.

1 Introduction

Transformers are emerging as the new workhorse of NLP, achieving the state-of-the-art in tasks such as language modeling, machine translation, and creating pretrained contextualized word embeddings. Eschewing recurrent computations, transformers are entirely based on self-attention, performing their computations largely in parallel. This enables them to scale to very long sequences (Vaswani et al., 2017; Dai et al., 2019; Child et al., 2019). On the other hand, it has been suggested that this limits their expressiveness, as they cannot process input sequentially (Tran et al., 2018; Dehghani et al., 2019; Shen et al., 2018a; Chen et al., 2018; Hao et al., 2019). One aspect thought to be challenging for sequence models is hierarchical structure and recursion. Hierarchical structure is widely thought to be essential to modeling natural language, in particular its syntax (Everaert et al., 2015). Consequently, many researchers have studied the capability of recurrent neural network models to capture context-free languages (e.g., Kalinke and Lehmann (1998); Gers and Schmidhuber (2001); Grüning (2006); Weiss et al. (2018); Sennhauser and Berwick (2018); Korsky

and Berwick (2019)) and linguistic phenomena involving hierarchical structure (e.g., Linzen et al. (2016); Gulordava et al. (2018)). Some experimental evidence suggests that transformers might not be as strong as LSTMs at modeling hierarchical structure (Tran et al., 2018), though analysis studies have shown that transformer-based models encode a good amount of syntactic knowledge (e.g., Clark et al. (2019); Lin et al. (2019); Tenney et al. (2019)).

In this work, we examine these questions from a theoretical perspective, asking whether models entirely based on self-attention are theoretically capable of modeling hierarchical structures involving unbounded recursion. Formally, we study their ability to perform two computations that are thought to be essential to hierarchical structure: First, their ability to correctly **close brackets**, a basic problem underlying all nonregular context-free languages and formalized by the **DYCK** language (Chomsky and Schützenberger, 1963). Second, their ability to **evaluate iterated negation**, a basic component of the task of evaluating logical formulas, amounting to evaluating the **PARITY** of bitstrings. We show that neither of these problems can be solved by transformers and similar models relying entirely on self-attention, unless the number or size of parameters increases with the input length. Besides representing basic building blocks of hierarchical structure, these languages also represent large classes of regular and context-free languages, meaning that our results carry over to classes of other formal languages. Our results therefore also yield more generally limitations on the ability of self-attention to model finite-state languages and context-free languages.

While theoretical work has investigated the power of recurrent neural networks in depth (e.g., Siegelman and Sontag (1995); Bengio et al. (1994); Weiss et al. (2018); Miller and Hardt (2019); Merrill (2019)), the theoretical study of self-attention has begun only recently (Pérez et al., 2019; Hsieh et al., 2019). Our study provides the first theoretical results on limitations in the power of self-attention. We will provide results both for hard and soft attention settings, using different proof methods. Our results are strongest in the hard attention setting, holding without further assumptions on activation functions and parameter norms. In the soft attention settings, we still obtain results assuming smoothness of activation functions as used in practical implementations.

After discussing related work (Section 2), we introduce self-attention (Section 3) and two fundamental formal languages representing regular and context-free languages (Section 4). We then prove that self-attention cannot model these languages using either hard (Section 5) or soft (Section 6) attention. Finally, we discuss our results (Section 7).

2 Related Work

Prior Work on Self-Attention Transformers were proposed by Vaswani et al. (2017), previous related work using self-attention includes Cheng

et al. (2016); Parikh et al. (2016); Paulus et al. (2018); Lin et al. (2017). It has been a recurrent suggestion in the literature that transformers, relying entirely on self-attention, are restricted computationally, as they cannot process their input sequentially. Dehghani et al. (2019) suggested that transformers cannot compute functions that require sequential processing of input, without providing further details or proofs. Similarly, Shen et al. (2018a); Chen et al. (2018); Hao et al. (2019) have introduced extensions of transformers with recurrence, citing similar intuitions about limitations of transformers. Our results provide the first explicit formalization of these limitations.

A few studies have experimentally tested the abilities of transformers to learn structures. Most related to our work, Tran et al. (2018) compared the ability of transformers and LSTMs to learn hierarchical structure, specifically English subject-verb agreement and evaluating logical formulas. Their experimental results suggested that LSTMs are better at learning hierarchical structure. Yang et al. (2019) experimentally investigated the power of self-attention to extract word order information, finding differences between recurrent and self-attention models; however, these were modulated by the training objective. Lin et al. (2019) and Tenney et al. (2019) show that BERT (Devlin et al., 2019) encodes syntactic information.

Theoretical study of transformers was initiated by Pérez et al. (2019), who theoretically studied the ability of Seq2Seq transformers to emulate the computation of Turing machines. While we consider incremental modeling of sequences, where the number of computation steps is bounded by the input length n, they study the setting in which the transformer computes an unbounded number of autoregressive decoding steps, not bounded in the input length n. Even more recently, and more closely related to our interest here, Hsieh et al. (2019) studied the adversarial robustness of transformers. While they focused on experiments on NLP tasks, they also provided a theoretical analysis, showing that a single self-attention layer with a single head will be robust against input perturbations, assuming that input embeddings are drawn uniformly from the unit sphere. One of our results, Lemma 5, can be seen as considerably widening the scope of their result, both by avoiding distributional assumptions, and by applying to transformers with arbitrary numbers of heads and layers.

Investigating the Power of Sequence Modeling Architectures The computational power of recurrent neural networks has been a focus of study. A particular focus has been on their ability to learn non-regular context-free languages, thought to provide simple models of recursion and hierarchical structure as found in natural language.

A range of studies has experimentally examined the ability of recurrent networks to model counter languages such as $a^n b^n$ (Kalinke and Lehmann, 1998; Gers and Schmidhuber, 2001; Cartling, 2008; Weiss et al., 2018; Suzgun et al., 2019). Other work has experimentally studied the perfor-

mance of recurent architectures on learning to recognize well-bracketed strings, a similar but more challenging problem (Sennhauser and Berwick, 2018; Skachkova et al., 2018; Bernardy, 2018). Beyond modeling formal languages, another line of work has studied the ability of LSTMs to model hierarchical structure as occurring in realistic natural language data (Linzen et al., 2016; Gulordava et al., 2018).

Recently, Merrill (2019) and Korsky and Berwick (2019) theoretically studied several types of recurrent networks. Merrill (2019) showed that – in the finite precision setting – LSTMs recognize a subset of the counter languages, whereas GRUs and simple RNNs recognize regular languages. Korsky and Berwick (2019) showed, among other results, that arbitrary-precision RNNs can emulate pushdown automata, and can therefore recognize all deterministic context-free languages.

A related, though different, strand of research has investigated the power of neural networks to model Turing machines. A classical result (Siegelman and Sontag, 1995) states that – given unlimited computation time – recurrent networks can emulate the computation of Turing machines. Very recently, Pérez et al. (2019) have shown the same result for both (argmax-attention) Transformers and Neural GPUs. The crucial difference between these studies and studies of language recognition is that, in these studies, the networks are allowed to perform unbounded recurrent computations, arbitrarily longer than the input length.

3 Self-Attention

Here we define self-attention as used in Transformers, following Vaswani et al. (2017), with some changes in the notation to simplify arguments in our proofs. We have an input $\mathbf{x} = x_1 \ldots x_n$, where all x_i come from some finite alphabet $\mathcal{V}$, and x_n is an end-of-sequence symbol. This input is then encoded into a sequence of input embeddings $v_1, \ldots, v_n$ using some embedding map $\mathcal{V} \to \mathbb{R}^k$. We furthermore have a sequence $p_1, p_2, \ldots$ of positional embeddings $p_i \in \mathbb{R}^k$. These are independent of the input $\mathbf{x}$, and can be computed through some predefined scheme, or could be learned for each position occurring in the training data (Vaswani et al., 2017). Input and positional embeddings are combined (e.g., via addition or concatenation) to vectors $y_i^{(0)} = f(v_i, p_i)$ $(i = 1, \ldots, n)$, which we will refer to as Layer 0.

A transformer has a fixed number L of **layers**; the **activations** $y_i^{(k)}$ at position i of the k-th layer $(k = 1, \ldots, L)$ are defined as follows. Each layer has a set of H **attention heads**; we first compute attention scores for the h-th head:

$$a_{i,j}^{(k,h)} = f_{k,h}^{att}\left(y_i^{(k-1)}, y_j^{(k-1)}\right) \qquad (1)$$

where $f_{k,h}^{att}$ combines the activations from the previous level into an attention score. This can be implemented e.g. using dot product or additive attention. Specifically, the implementation described by Vaswani et al. (2017, p. 5) linearly transforms the position-wise activations $y_i^{(k-1)}$ separately into

'query' vectors $Qy_i^{(k-1)}$ and 'key' vectors $Ky_i^{(k-1)}$ (for some parameter matrices K, Q); the attention score $a_{i,j}^{(k,h)}$ is then implemented as a scaled dot product of query $Qy_i^{(k-1)}$ and key $Ky_j^{(k-1)}$.

The activation of the head is computed by weighting according to attention weights $\hat{a}_{i,j}^{(k,h)}$:

$$b_{i,k,h} = \sum_{j=1}^{n} \hat{a}_{i,j}^{(k,h)} y_j^{(k-1)} \quad (2)$$

We note that the implementation described by Vaswani et al. (2017) first linearly transforms the activations $y_j^{(k-1)}$ into 'value vectors' before multiplying with $\hat{a}_{i,j}^{(k,h)}$; this is mathematically equivalent to applying this linear transformation to $b_{i,k,h}$ as part of the map f^{act} we describe below.

In the soft attention version, these weights $\hat{a}_{i,j}^{(k,h)}$ are obtained by the softmax operation: $\hat{a}_{i,\cdot}^{(k,h)} = \text{softmax}(a_{i,\cdot}^{(k,h)})$. In the hard attention variant (Pérez et al., 2019), one takes the actual maximum attention values: $\hat{a}_{i,j}^{(k,h)} = \delta_{j,\arg\max_{j'} a_{i,j'}^{(k,h)}}$.[1]

The per-position activations are then computed as

$$y_i^{(k)} := f^{act}(y_i^{(k-1)}, b_{i,k,1}, \ldots, b_{i,k,H}) \quad (3)$$

where f^{act} is implemented as a fully-connected feedforward network with a skip-connection (Vaswani et al., 2017) from $y_i^{(k-1)}$ to $y_i^{(k)}$.

Hard and Soft Attention There is a choice between soft attention and hard attention (Shen et al., 2018b; Pérez et al., 2019). The one prior theoretical study of transformers (Pérez et al., 2019) assumes hard attention. In practice, soft attention is easier to train with gradient descent; however, analysis studies suggest that attention often concentrates on one or a few positions in trained transformer models (Voita et al., 2019; Clark et al., 2019) and that the most important heads are those that clearly focus on a few positions (Voita et al., 2019), suggesting that attention often behaves like hard attention in practice. We will examine both hard (Section 5) and soft (Section 6) attention.

Formalizing Language Recognition We consider the problem of language recognition, the task of classifying input strings as belonging to or not belonging to a formal language. Following Weiss et al. (2018), we formalize this as the sequence-to-sequence task of mapping words to labels 1 ('in the language') and 0 ('not in the language'). Following the construction of transformers in sequence-to-sequence tasks (Vaswani et al., 2017), we compute a softmax probability vector for this label from the last activation $y_n^{(L)}$, obtained after reading the end-of-sequence symbol.

4 Regular and Context-Free Languages

We will analyze the ability of transformers to recognize regular and context-free languages, using two prominent representatives.

PARITY is the set of bit strings such that the number of 1s is even. This is a very simple regular language, recognized by a finite-state automaton with two states. The regular languages form the

[1] When there are multiple positions with maximal attention weight, we will assume that the one occurring first in the sequence is chosen. Our analysis also works under other schemes of resolving ties, such as random selection.

lowest layer of the Chomsky hierarchy, and even simple RNNs can compute all regular languages. Within the regular languages, a particularly basic class is formed by the *counter-free* or *star-free* languages (McNaughton and Papert, 1971), which can be expressed by regular expressions using only union, complementation, and concatenation. In some sense, PARITY is the simplest non-counter-free, or *periodic*, regular language. This means, if transformers cannot compute PARITY, they cannot recognize (almost)[2] any regular language that is not counter-free. In the context of natural language, PARITY naturally arises in the context of evaluating logical formulas: Evaluating iterated negations is tantamount to counting whether the number of nested negations is even or odd. If transformers cannot compute parity, they also cannot evaluate logical formulas accurately.

2DYCK is the language of correctly bracketed words consisting of two types of brackets ('(', ')' and '[', ']'). This language is a very simple model of hierarchical structure. The Chomsky-Schützenberger theorem (Chomsky and Schützenberger, 1963) states that any context-free language arises from a variant of 2DYCK with multiple types of parentheses through intersection with a regular language and homomorphisms. Consequently, the ability of LSTMs to model languages such as 2DYCK has been an object of experimental study (Sennhauser and Berwick, 2018; Skachkova et al., 2018; Bernardy, 2018). Our theoretical results will show that transformers are strongly limited in their ability to model 2DYCK, including variants with fewer or more types of parentheses.

[2]Inability to compute PARITY entails that they cannot recognize any regular language whose syntactic morphism is not quasi-aperiodic (Barrington et al., 1992, p. 488).

5 Results for Hard Attention

We will start our analysis with the study of hard attention (Pérez et al., 2019). We show that hard attention transformers cannot represent PARITY or 2DYCK. To keep the results maximally general, our analysis will use combinatorial arguments and make no assumption about, e.g., activation functions and the norms of parameter matrices. In fact, we do not even assume that the internal position-wise representations $y_j^{(k)}$ in each layer are vector-valued, as opposed to, say, discrete structures.

We aim to prove that no hard-attention transformer is capable of representing PARITY or 2DYCK, by constructing – for any given candidate transformer model – a set of input words that this model will have to misclassify. The basic idea (see Figure 1) behind the proof is that, by fixing a small fraction of the input symbols in a particular way, we can 'capture the attention' of the transformer in such a way that it ends up ignoring almost all remaining input symbols. This shows that the transformer could not have solved a problem such as PARITY, where every single input bit matters.

In order to formalize the idea of 'fixing' a few input bits, we introduce the notion of input restrictions: An **input restriction** (short: **restriction**) ρ is a family of maps $\rho_n : \{1, \ldots, n\} \to \{*, 0, 1\}$

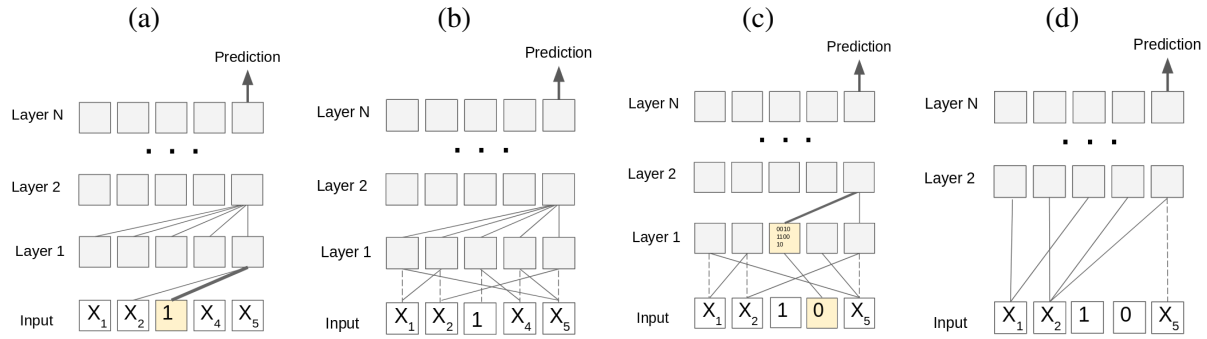

Figure 1: Iteratively reducing the layers of a transformer by fixing a few input symbols. (a) By applying a suitable input restriction, we fix a small number of input symbols, 'attracting' attention from the first layer to a few inputs. (b) After this step, Lemma 4 ensures that each activation in the first layer only depends on a small number of input symbols that it can attend to (solid connections), plus the input that feeds into it via a skip connection (dashed connections). (c) We again fix a few input symbols in such a way as to 'attract' attention of layer-2 heads to some layer-1 activations. As a result, each layer-2 activation only depends on a small number of layer-1 activations, again by Lemma 4. (d) After this step, each layer-1 activation only depends on a few inputs, and we can remove layer 1.

for $n \in \mathbb{N}$. An input restriction ρ is applied to a transformer by fixing, when the input length is n, the input symbol x_i to the value $\rho_n(i)$ whenever $\rho_n(i) \in \{0,1\}$. The output value of the resulting transformer only depends on those inputs x_i such that $\rho_n(i) = *$.

The idea of using such input restrictions has been successful in the theory of Boolean circuits (Furst et al., 1984; Hastad et al., 1994). In particular, Furst et al. (1984) famously used it to prove that polynomial-size bounded-depth Boolean circuits with $\wedge, \vee$, and $\neg$ gates cannot compute PARITY. We describe a new method to prove existence of suitable restrictions appropriate to transformers, as the proof approaches from the Boolean circuit literature do not seem to generalize to networks with real-valued activations.

The following result formalizes the claim that any transformer can be forced to ignore input bits by fixing some inputs in a particular way:

Theorem 1. *Let any hard attention transformer be given, and let* $C \in (0,1)$. *Then there is a restriction* ρ *and an integer* $c > 0$ *such that*

$$|\{i \leq n : \rho_n(i) = *\}| \geq Cn$$

(for all sufficiently large n) and such that the function computed by the transformer on the restricted input depends only on $\leq c$ *inputs, independent of input length n.*

We first show how this entails that transformers do not recognize the two formal languages:

Corollary 2. *Transformers with hard attention cannot model* PARITY *or* 2DYCK.

Proof. For PARITY, after applying a restriction, the transformer's output depends on c inputs. An input of sufficiently large size n thus has unrestricted inputs that do not influence the output. But flipping a single input bit changes the value, so the transformer's output cannot match membership in

PARITY beyond chance for such n.

For 2DYCK, we show that hard attention transformers cannot even solve the simpler variant 1DYCK with a single bracket type ('(', ')'). We first restrict the first $0.2n$ input positions to '(' and the last $0.2n$ positions to ')'. After then applying the restriction provided by the theorem with $C = 0.9$, the resulting restricted input will still be compatible with both well-bracketed and non-well-bracketed inputs, but the prediction will depend only on a bounded number of positions. As the prediction depends on only a bounded number of positions, this shows the transformer could not recognize 1DYCK, and thus also not 2DYCK. □

Discussion It may be instructive to compare to similar languages that *can* be modeled by hard-attention transformers. First, 1^* (over the alphabet $\{0,1\}$) is the regular language of words that have only ones and no zeroes; its minimal automaton has two states, like PARITY. A transformer can recognize this by having an attention head that attends to a position with zero input if it exists, and rejects if the head found such a position. Second, $a^n b^n$ is a very basic context-free language. It can be recognized using suitable positional embeddings by (1) having one head attend to the largest position n, (2) using this information to attend to any b at position $< n/2$ or any a at position $\geq n/2$. If such a symbol is found, the model rejects, else it accepts. A crucial difference between these languages and PARITY / 2DYCK is that fixing a few inputs in any part of an input string can easily force nonmembership, e.g. a single 0 for 1^*, and an a in the second half for $a^n b^n$. Therefore, such simple languages are immune to the depth reduction method, and indeed *can* be modeled perfectly with self-attention.

In general, the depth reduction method applies to languages that are sufficiently *sensitive*: If, for some $C \in (0,1)$, fixing Cn input symbols cannot force a word to be inside or outside of the language, then hard-attention transformers cannot recognize this language. Sensitivity of functions has been studied in computational complexity (Boppana, 1997; Gopalan et al., 2016) and more recently linked to generalization in feedforward networks (De Palma et al., 2018). We intend to investigate these connections in future work.

Proof Idea of the Theorem Our approach for proving Theorem 1 will be to construct input restrictions in a layerwise manner, starting from layer 1. In order for this construction to go through, the main challenge is to construct a suitable restriction at a given layer: As shown in Figure 2, this restriction should only affect a few input bits (about $(1-C^{1/L})n$ many input bits), while forcing each attention head in the first layer to ignore all but c input bits. Perhaps surprisingly, this is possible; the idea is to fix input bits that achieve high attention scores for several heads, so that input bits that cannot achieve such high attention scores will be ignored.

Once we have shown that such a restriction always exists, we can use this technique to itera-

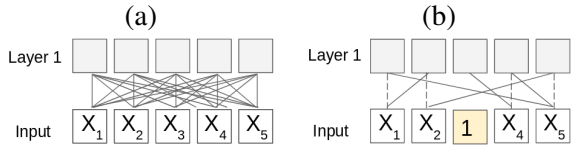

Figure 2: Finding a good input restriction: (a) Every attention head in the first layer could potentially attend to any input bit. (b) Perhaps surprisingly, one can fix a small number of input bits in such a way that each layer-1 attention head can only possibly attend to c (here, $c = 1$) inputs, and ignores all other inputs. Each activation vector $y_j^{(1)}$ in the first layer then only depends on the $H \cdot c$ inputs that its H (here, $H = 1$) attention heads can attend to, plus the input x_j that feeds into it via a skip-connection.

tively remove layers, as illustrated in Figure 1: After we have applied the first such restriction, each of the heads in the first layer will only depend on a bounded number c of input positions. In the second step, we apply the same argument to the heads in the second layer, so that each head in the second layer only depends on a bounded number c' of heads in the first layer. After this step, we can collapse the first layer into a collection of feedforward networks that transform a bounded number $\leq cc'$ of input positions into an activation $y_i^{(0)}$ of the lowest layer. After this step, the first layer has been entirely removed. Iterating this argument, we remove all layers until the prediction output only depends on a bounded number of input positions, bounded independently of input length.

We now make these ideas formal. After the removal of the first layer of a transformer, the resulting structure is not a transformer any more, as each head in the lowest layer now depends on a *combination* of input positions. We introduce a technical definition to make this concept precise:

Definition 3 (c-Transformer)**.** *Let c be a positive integer. A c-transformer with L layers is one in which the layer-0 activations $y_j^{(0)}$ depend on the embeddings not just at one position j, but are a function of the embeddings at $\leq c$ input positions:*

$$y_j^{(0)} = f_{n,j}^{inp}((v_{i_1^{j,n}}, p_{i_1^{j,n}}), \ldots, (v_{i_c^{j,n}}, p_{i_c^{j,n}})) \quad (4)$$

for some indices $i_s^{j,n} \in \{1, \ldots, n\}$ ($s = 1, \ldots, c$).

With this technical notion, we show that we can reduce layers, iteratively removing the lowest layer until no self-attention layer is left:

Lemma 4 (Depth Reduction Lemma)**.** *Given a c-transformer with L layers, and some restriction ρ such that*

$$|\{i \leq n : \rho_n(i) = *\}| \geq Cn \quad (5)$$

($C \in (0, 1]$) for all sufficiently large n. Choose any $C' < C$. Then there is a restriction ρ' such that

$$|\{i \leq n : \rho'_n(i) = *\}| \geq C'n \quad (6)$$

for all sufficiently large n, and such that the resulting function is computed by a $(c \cdot (2^c kH + 1))$-transformer with $L - 1$ layers, for some integer k (depending on C'), where $H \geq 1$ is the number of attention heads at each layer and position.

The lemma implies Theorem 1:

Proof of Theorem 1. The output of the transformer is determined by the last activation $y_n^{(L)}$. Apply the Depth Reduction Lemma iteratively,

choosing the constants C' in the lemma appropriately, until only the zero-th layer remains. Then, after applying the resulting restriction, the final activation $y_n^{(L)}$ is now computed by $y_n^{(0)}$, which is determined by a bounded number of input bits. $\square$

5.1 Proving the Depth Reduction Lemma

In this section, we will prove the Depth Reduction Lemma. We construct the restrictions ρ'_n separately for each n, on the basis of the given restriction ρ_n. In this process, we will only *restrict additional bits*, that is, the only case in which $\rho'_n(i)$ can be different from $\rho_n(i)$ is that $\rho'_n(i)$ may be 0 or 1 where $\rho_n(i)$ was $*$. The construction proceeds in three stages $\rho_n^{(1)}$, $\rho_n^{(2)}$, and $\rho_n^{(3)} = \rho'_n$, which all may restrict additional bits. At the end, we verify that the conclusion of the Depth Reduction Lemma is satisfied for the resulting restriction ρ'_n.

Throughout the proof, we will need a few parameters independent of n: First, we need an integer k that has to be sufficiently large for the proof to succeed, and will be fixed later in the proof. Second, we need parameters $\eta \in (0, \frac{1}{2})$, $q \in (0,1)$ and $\delta > 0$; we will also fix the specific values later in the proof.

Stage 1 We start from ρ_n and first modify it into a restriction $\rho_n^{(1)}$ such that each input bit serves as an input to at most $\leq \frac{1}{\eta} c/C$ many different layer-0 heads, when applying $\rho_n^{(1)}$. Assume the number of input bits feeding into more than $\frac{1}{\eta} c/C$ different layer-0 activations is $\geq \eta Cn$. Then the number of pairs of input bits and depending layer-0 activations is $> \eta Cn \cdot \frac{1}{\eta} c/C = nc$. But there are at most nc such pairs, because there are n layer-0 activations, each of which depends on $\leq c$ inputs. So the number of input bits with $> \frac{1}{\eta} c/C$ depending layer-0 heads is $\leq \eta Cn$. We can obtain $\rho_n^{(1)}$ from ρ_n by restricting these input bits to some fixed value in $\{0,1\}$ (it doesn't matter which one), and the set $\{i \leq n : \rho_n^{(1)}(i) = *\}$ still has at least $(1-\eta)Cn$ elements, for all sufficiently large n.

Stage 2 We now describe the second stage. We write (h,i) for a layer-1 attention head h ($h = 1,\ldots,H$) at position i ($i = 1,\ldots,n$). Fix such a head (h,i). As $y_i^{(0)}$ depends on $\leq c$ input bits, it can take on at most $\leq 2^c$ possible values. For each possible value z, and each position $j \in \{1,\ldots,n\}$, we compute the maximum possible attention value that can be achieved for this pair:

$$\max_{x_1 \ldots x_n \,:\, y_i^{(0)} = z} f_{1,h}^{att}(z, y_j^{(0)}) \tag{7}$$

considering only inputs $x_1 \ldots x_n$ that are compatible with the restriction $\rho_n^{(1)}$ constructed at Stage 1. For each value z, we order the positions $\{1,\ldots,n\}$ downwards by this value, obtaining a sequence $j_1^{(z)},\ldots,j_n^{(z)}$ for each layer-1 attention head h at a position i and each possible value z of $y_i^{(0)}$ (In the case of ties, we order by position, by Footnote 1). For each layer-1 attention head and each z, we select a sequence $1 \leq i_1^{(z)} < i_2^{(z)} < \cdots < i_k^{(z)} \leq n$ such that (1) for each $i_s^{(z)}$, there is at least one input x_q that only feeds into the activation at position $j^{(z)}_{i_s^{(z)}}$ and no $j^{(z)}_{i_{s'}^{(z)}}$ ($s \neq s'$), and (2) $i_k^{(z)}$ is minimal, i.e.

there is no subsequence with smaller $i_k^{(z)}$ that also satisfies (1). This construction is visualized in an example in Figure 3. Such a subsequence exists unless $n \leq ck$, in which case the Depth Reduction Lemma is already satisfied for this input length n.

If z is a possible value of the activation $y_i^{(0)}$, then we say that a pair $((i,h),z)$, of a head h at position i and a possible value z of $y_i^{(0)}$, is **satisfied** if one of the layer-0 activations $y_{i_s^{(z)}}^{(0)}$ ($s \in \{1,\dots k\}$) is fixed by $\rho_n^{(1)}$ to the value achieving the maximum attention value (7). Also, we say that (h,i) is satisfied if each $((h,i),z)$ is. The idea behind this definition is: If $((h,i),z)$ is satisfied, then there are at most k different layer-0 heads that this head could attend to when applying ρ_n', assuming that $y_i^{(0)}$ takes the value z. As a consequence, a satisfied head can only depend on $c \cdot (2^c k + 1)$ many input bits. Our aim will be to construct ρ_n' so that each layer-1 head is satisfied.

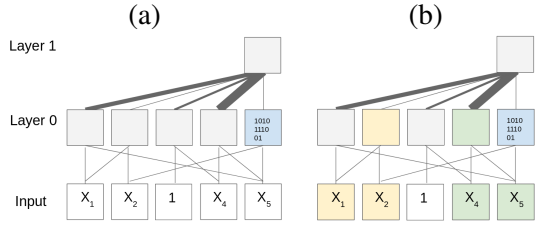

Figure 3: Selecting the sequence $i_1^{(z)} \dots i_k^{(z)}$: We have a c-transformer with $c = 2$, i.e., each Layer 0 activation only depends on at most two input bits. (a) We fix a head in Layer 1 at position i (here, $i = 5$), and some value z for $y_i^{(0)}$ (blue activation). For each other Layer-0 activation $y_j^{(0)}$, we compute the maximal possible attention value between that activation and the Layer 1 head, assuming the fixed value z for $y_i^{(0)}$ – these maximum attention values are visualized by the thickness of the different lines. (b) We select $k = 2$ activations from Layer 0, marked in yellow and green. For each of these, there is at least one (in fact, two in the example) input bits (also marked in yellow and green) that feed into this one and no other selected activation.

A layer-1 head k-**depends** on some input x_i if $\rho_n(i) = *$ and x_i appears as an input to some $j_r^{(z)}$ for $r \leq i_k^{(z)}$, for some z. Because $i_k^{(z)}$ is minimal, a layer-1 head k-depends on an input if and only if that input appears as an input to some $j_{i_s^{(z)}}$ ($s \leq k$). In particular, a layer-1 head k-depends only on $\leq 2^c ck$ input variables. Two layer-1 head are k-**neighbors** if some $j_{i_s^{(z)}}$ for one and $j_{i_{s'}^{(z')}}$ for the other both k-depend on some input bit x_l.

We will construct ρ_n' using probabilistic arguments over randomly chosen restrictions. For this approach to succeed, we require a sufficient amount of independence between the activations of different heads in layer 1. We thus need to ensure that the number of k-neighbors of each head is bounded. Recall $\eta \in (0, \frac{1}{2})$, and let H be the number of attention heads in each position of layer 1.

We modify $\rho_n^{(1)}$ into $\rho_n^{(2)}$ so that each layer-0 head has at most $\leq 2^c kH$ many k-depending unsatisfied layer-1 heads. Assume that indeed some layer-0 head has more than $2^c kH$ many k-depending unsatisfied layer-1 heads. By fixing $\leq c$ input bits and appealing to the Pigeonhole principle, we can fix this head to a value that achieves the maximum attention value for at least $> kH$ many of these k-depending layer-1 heads. Let $\rho_n^{(2)}$ be the restriction resulting from adding this to $\rho_n^{(1)}$. Once we have done this, $\{i \leq n : \rho_n^{(2)}(i) = *\}$ still has at least $(1-\eta)Cn - c$ elements, and more than kH many additional pairs $((h,i),z)$ are now also satisfied. We then repeat the selection of the sequence $j_1^{(z)}, \dots, j_n^{(z)}$ (substituting $\rho_n^{(2)}$ for $\rho_n^{(1)}$

in the definition), and repeat the construction described here, to restrict additional input bits in $\rho_n^{(2)}$. We iterate this procedure until no layer-0 head has $> 2^c kH$ many k-depending unsatisfied layer-1 heads (h,i). This procedure can be iterated at most until each layer-1 head is satisfied, that is, at most $\frac{2^c Hn}{kH} = \frac{2^c n}{k}$ times. Let U be the number of times this procedure is iterated ($U \leq \frac{2^c n}{k}$). At the end, $\{i \leq n : \rho_n^{(2)}(i) = *\}$ has at least $(1-\eta)Cn - cU \geq \left((1-\eta)C - \frac{2^c c}{k}\right) n$ elements. By choosing k so large that $\frac{2^c c}{k} \leq \eta$, we find that $\{i \leq n : \rho_n^{(2)}(i) = *\}$ still has at least $(1-2\eta)Cn$ many elements. Once this is completed, each layer-0 head has at most $\leq 2^c kH$ many k-depending unsatisfied layer-1 heads. Thus each input bit now has at most $\leq \frac{2^c}{\eta} kcH/C$ many k-depending unsatisfied layer-1 heads. Consequently, every unsatisfied layer-1 head has at most $f \leq \frac{2^{2c}}{\eta} c^2 k^2 H/C$ many unsatisfied k-neighbors.

Stage 3 In order to construct the third and final restriction $\rho_n^{(3)}$, we apply the "probabilistic method": We define a probability distribution over restrictions $\rho_n^{(3)}$, and show that the probability assigned to restrictions of the type we require is strictly greater than zero, showing that such a restriction exists. For each input length n, define the distribution over restrictions $\rho_n^{(3)}$ that independently assigns to each input position $i \in \{1,\ldots,n\}$ the symbol 1 or 0 with probability $q/2$ each ($q \in (0,1)$ chosen later), and $*$ with probability $1-q$. On those input bits where $\rho_n^{(2)}(i) \neq *$, we restrict this random restriction to agree with $\rho_n^{(2)}(i)$. For an layer-1 attention head (h,i) and for each value z (there are at most 2^c such), define $X_{i,h}^{(z)}$ to be the event that, for this head, none of $y^{(0)}_{j_{i_1^{(z)}}}, \ldots, y^{(0)}_{j_{i_k^{(z)}}}$ are fixed to the value that produces the highest attention weight. Define X_0 to be the event that more than $(1+\delta)q$ of the input bits that $\rho_n^{(2)}$ maps to $*$ are set to $0/1$ by $\rho_n^{(3)}$ (where $\delta \in (0,1)$, to be fixed later). Our goal will be to show that a nonzero amount of probability mass is assigned to restrictions ρ'_n avoiding all events $\{X_0\} \cup \{X_{i,h}^{(z)} : i,z\}$.

First, a Chernoff bound gives (Mitzenmacher and Upfal, 2017, Theorem 4.4)

$$\mathbb{P}(X_0) \leq \exp\left(-\delta^2 q(1-2\eta)Cn/3\right) \qquad (8)$$

since $\rho_n^{(2)}$ had $\geq (1-2\eta)Cn$ unrestricted input bits after Stage 2.

Second, we show that the probability of $X_{i,h}^{(z)}$ ($i = 1,2,\ldots,n$, $h = 1,\ldots,H$) decays exponentially in k. First, if $((h,i),z)$ is already satisfied after Stage 2, then $\mathbb{P}(X_{i,h}^{(z)}) = 0$. Else, fixing z for ease of notation, let $Y_{i,h}^t$ ($t = 1,\ldots,k$) be the event that the layer-0 activation $y^{(0)}_{j_{i_t^{(z)}}}$ is not fixed to the value that produces the highest attention weight, for the given attention head (h,i). Note that $X_{i,h}^{(z)} = \bigcap_t Y_{i,h}^t$. We have $\mathbb{P}(Y_{i,h}^s) \leq 1-(q/2)^c \in (0,1)$. Any $Y_{i,h}^s$ can be statistically dependent on at most $c \cdot \frac{1}{\eta} c/C = \frac{1}{\eta} c^2/C$ other events $Y_{i,h}^{s'}$, because each input bit has at most $\frac{1}{\eta} c/C$ depending layer-0 heads. Therefore, there is a set of $\geq \frac{k}{\frac{1}{\eta}c^2/C}$ independent events among these. Call these $Y_{i,h}^{t_1}, \ldots, Y_{i,h}^{\frac{k}{\frac{1}{\eta}c^2/C}}$. Then

$X_{i,h}^{(z)} \subseteq \bigcap_{s=1}^{\frac{k}{\frac{1}{\eta}c^2/C}} Y_{i,h}^{t_s}$, and thus

$$\mathbb{P}(X_{i,h}^{(z)}) \leq \prod_{s=1}^{\frac{k}{\frac{1}{\eta}c^2/C}} \mathbb{P}(Y_{i,h}^{t_s}) \leq (1-(q/2)^c)^{\frac{k}{\frac{1}{\eta}c^2/C}} \quad (9)$$

for each $i = 1,2,\ldots,n$ and $h = 1,\ldots,H$.

In order to conclude that there is a restriction $\rho_n^{(3)}$ avoiding all events $\{X_0\} \cup \{X_{i,h}^{(z)} : i,h,z\}$, we apply the Lovász Local Lemma (Mitzenmacher and Upfal, 2017, Theorem 6.17). Each event $X_{i,h}^{(z)}$ is statistically independent of the set $\{X_{(j,h')}^{(z')} :$ heads (j,h') and (i,h) are not k-neighbors$\}$. The complement of this set has cardinality $\leq f = \frac{2^{2c}}{\eta}c^2k^2H/C$, as concluded at the end of Stage 2. Set $A := \frac{1}{k^2}$, $B := \frac{1}{2}$. By the Lovász Local Lemma, it is sufficient show the following:

$$\mathbb{P}(X_{i,h}^{(z)}) \leq A(1-B)(1-A)^f \quad (10)$$

$$\mathbb{P}(X_0) \leq B(1-A)^{2^cHn} \quad (11)$$

The Lovász Local Lemma then guarantees that there is some input restriction $\rho_n^{(3)}$ that avoids all events $\{X_0\} \cup \{X_{i,h}^{(z)} : i,h,z\}$. For (10), we need

$$D \leq A^{1/k}(1-B)^{1/k}(1-A)^{f/k} \quad (12)$$

where $D = (1-(q/2)^c)^{\frac{1}{\frac{1}{\eta}c^2/C}} \in (0,1)$. For the first term on the right,

$$\lim_{k\to\infty} A^{1/k} = \lim_{k\to\infty} \exp\left(-\log(k^2)/k\right) = 1$$

Also, $\lim_{k\to\infty}(1-A)^{f/k}$ equals

$$\lim_{k\to\infty}\left(1-\frac{1}{k^2}\right)^{\frac{2^{2c}}{\eta}c^2kH/C} = \lim_{k\to\infty}\left(1-\frac{E^2}{k^2}\right)^k = 1$$

for $E := \frac{2^{2c}}{\eta}c^2H/C$. So, if we choose k large enough (independently of n), the RHS of (12) can be made arbitrarily close to 1, in particular, greater than D. In order to also satisfy (11), we need

$$\exp\left(-\delta^2 q(1-2\eta)C/3\right) \leq B^{1/n}(1-A)^{2^cH}$$

which holds for n, k large enough (again, choosing k independent of n). In conclusion, there exists, for each sufficiently-large n, a restriction $\rho_n^{(3)}$ that avoids all events $\{X_0\} \cup \{X_{i,h}^{(z)} : i,z\}$, for some k independent of n. For such a $\rho^{(3)}$, we have

$$|\{i \leq n : \rho_n^{(3)}(i) = *\}| \geq (1-2\eta)\cdot(1-(1+\delta)q)Cn$$

for all sufficiently large n. Then choose $\eta \in (0,\frac{1}{2})$ small, $q \in (0,1)$, and $\delta > 0$ (such that $(1+\delta)q \in (0,1)$) in such a way as to achieve $(1-2\eta)\cdot(1-(1+\delta)q) = C'/C$.

After applying $\rho_n^{(3)}$, every layer-1 head $b_{j,1,h}$ depends only on (1) the c input bits feeding into $y_j^{(0)}$, and (2) the $\leq c2^ck$ input bits that the head k-depends on. Thus, each layer-1 activation $y_j^{(1)}$ only depends on $\leq c\cdot(2^ckH+1)$ input bits: There are $\leq H\cdot c\cdot 2^c\cdot k$ input bits that the H different attention heads k-depend on, plus a skip-connection from $y_j^{(0)}$, which itself depends on $\leq c$ input bits. We can thus remove layer 0, convert layer-1 activations $y_j^{(1)}$ into layer-0 activations $y_j^{(0)}$, and ob-

tain a $(c \cdot (2^c kH + 1))$-transformer performing the same computation as before when $\rho^{(3)}$ is applied. This concludes the proof of the Depth Reduction Lemma.

6 Results for Soft Attention

In the previous section, we showed that transformers using hard attention are not able to recognize a range of core formal languages. In this section, we study soft attention. It turns out that proving limitations as strong as what we found in the hard attention setting would settle a major open problem in computational complexity, and may therefore be extremely hard to attain with currently available mathematical methods.[3] This barrier prevents us from proving bounds on the *accuracy* that soft attention transformers can achieve; nevertheless, we will be able to prove limitations on the achievable *cross-entropy* in modeling distributions over the formal languages. We will use the smoothness of the operations used in transformers to show that any transformer, as inputs get longer, will not be able to robustly model such distributions. The idea behind the proof is that the impact of any single input symbol on the output of the transformer is small if the input is long:

Lemma 5. *Let a soft attention transformer be given, and let n be the input length. If we exchange one input symbol x_i ($i < n$), then the change in the resulting activation $y_n^{(L)}$ at the decoder layer is bounded as $O(\frac{1}{n})$ with constants depending on the parameter matrices.*

This contrasts with recurrent networks: Changing a single input can have nonnegligible impact on the final state even for very long input. E.g., an RNN recognizing PARITY through a hidden state that encodes parity of the current prefix will flip its hidden state if a single input bit is flipped.

Lemma 5 entails that, as inputs become longer, soft attention transformers cannot achieve good cross-entropies on prediction problems that are very sensitive to individual input symbols: A Lipschitz-continuous prediction function, such as a ReLU MLP with a softmax output, will not be able to make very different predictions for inputs that are encoded into similar activations $y_n^{(L)}$.

To make all our assumptions explicit, we will assume the following setting, though the results do not depend on the specific details. For PARITY, we consider the distribution over bitstrings generated by a two-state automaton that – if the number of 1s emitted so far is even – terminates with probability p, and otherwise emits a 1 or 0 with equal probability each. Given a prefix of a string drawn from this distribution, we ask the transformer to predict the next symbol from $\Sigma = \{0, 1, \text{ENDOFSEQUENCE}\}$. Note that the next symbol can be ENDOFSEQUENCE if and only if the prefix has an even number of 1s. For 2DYCK, we follow the experimental study of Skachkova et al. (2018) and take the distribution generated by a PCFG that expands $S \to (S)S$ or $S \to [S]S$ with

[3] Showing that soft attention transformers cannot achieve perfect accuracy on evaluating Boolean formulas would separate the complexity classes LTC^0 and NC^1, a widely conjectured but long-open problem in computational complexity.

probability $p/2$ each, and $S \to \varepsilon$ with probability $1-p$. We ask the model to predict the next character among $\Sigma = \{(,),[,],\text{EndOfSequence}\}$.

Theorem 6. *Let a soft attention transformer be given for* Parity *or* 2Dyck. *As $n \to \infty$, cross-entropy on predicting the next symbol converges to unigram chance level (*Parity*), or is at least separated from the optimal cross-entropy by some constant $\varepsilon > 0$ (*2Dyck*).*

Proof. First, let us consider Parity. Exchanging a single bit flips membership in Parity. Thus, for any $x \in$ Parity, there is a string $x' \notin$ Parity, differing only in one bit. As x and x' differ only in one bit, the transformer's output activations differ by $O(\frac{1}{n})$. Therefore, a Lipschitz-continuous prediction function cannot robustly assign different next-symbol probabilities after even and odd numbers of 1s, and cross-entropy will converge to unigram chance level.

For 2Dyck, consider a string x of length n, known to be the prefix of a word generated by the PCFG. One can show that there is a constant $P_0 \in (0,1)$ (dependent on p but not n), such that x both ends with a closing bracket, and is unbalanced, with probability $\geq P_0$.[4] After such an x, the next symbol is a closing bracket with constant nonzero probability $(1-p)$. If x can be followed by, say, ')' but not ']', then there is a string x', differing only in one input position, but for which the next symbol can be ']' but not ')'. As x was assumed to end with a closing bracket, the exchanged symbol is not the last symbol of x, and thus the transformer's predictions on x and x' differ only by $O(\frac{1}{n})$. We can decompose the prediction task into predicting (1) whether an opening or closing bracket, or EndOfSequence, follows, and (2) whether a round or square bracket follows, in case a bracket follows. The cross-entropy loss is the sum of the cross-entropies incurred on these two successive predictions tasks. Therefore, when such a prefix x is followed by the correct closing bracket, say ')', the model will incur, as $n \to \infty$, a cross-entropy loss on the second task of at least $\log 2$, reflecting at-chance performance in choosing between the two possible closing brackets. In contrast, optimal cross-entropy loss on (2) would be 0, as the bracket type (round or square) is actually fully determined by x. Thus, the overall cross-entropy on all prefixes x of length n is, asymptotically as $n \to \infty$, at least $P_0 \cdot (1-p) \cdot \log 2 > 0$ more than the optimal cross-entropy. □

We proceed to proving Lemma 5.

[4] One can show this formally using a Markov chain argument. Let the **height** $H(x)$ of a word x be the number of opening brackets minus the number of closing brackets in x. When iteratively sampling a symbol sequence using a pushdown automaton for 2Dyck, the height H_n of the prefix x up to length n forms a Markov chain taking values in $\mathbb{N}$. The prefix x is unbalanced if and only if $H_n > 0$, this is always the case whenever n is odd. Restricting to even n, the chain $\{H_n : n = 0,2,4,...\}$ is aperiodic and takes values in $\{0,2,4,\dots\}$. It is also positive recurrent, as words sampled from the PCFG have finite expected length (Skachkova et al., 2018, 2.2.1). Therefore, the Markov chain $\{H_n : n = 0,2,4,...\}$ converges to its stationary distribution (Mitzenmacher and Upfal, 2017), which – by positive recurrence – must assign some nonzero weight π_{2i} to each height $2i$ ($i \geq 0$). Hence, even when n is even, the prefix x is unbalanced with nonzero probability $1-\pi_0$ asymptotically independent of n. Also, since the transition probabilities $P(H_{n+1}|H_n)$ are independent of n, there is an asymptotically constant nonzero probability that $x_1 \dots x_{|x|-1}$ has height larger than x, i.e., the last bracket of x is a closing one.

Proof of Lemma 5. We compare the activations at the decoder layer for two inputs that only differ in the input at the i-th position. Let $D = \|v_i - v_i'\|_2$ the norm of the difference of the input embeddings at this position. We show by induction over $k = 1, \ldots, L$ that, for some $C > 0$ (chosen below) the differences between the resulting activations $y_j^{(k)}$, $y_j^{(k)'}$ are bounded as:

$$\|y_i^{(k)} - y_i^{(k)'}\| \leq C^{2k} D = O(1)$$
$$\|y_j^{(k)} - y_j^{(k)'}\| \leq \frac{H^k C^{2k} D}{n} = O(1/n) \quad (j \neq i)$$

Once we have shown this, we know that the influence of any individual input on the final prediction is $O(\frac{1}{n})$, with constants depending on the norms of parameter matrices and the number of layers.

At this point, it is worth remarking that a key property of transformers for this proof is that the number L of layers is bounded independently of the input length. A similar proof strategy can also be applied to other fixed-depth architectures that combine unboundedly many inputs in a smooth manner, such as 1D temporal convolutions with average pooling.

For $k = 0$, $\|y_i^{(0)} - y_i^{(0)'}\| \leq D$, and $\|y_j^{(0)} - y_j^{(0)'}\| = 0$ for $j \neq i$. For $k > 0$, we first note that $\|y_j^{(k)}\|_2 \leq 2C_{f^{act}}^L(\|p_j\| + \|v_j\|)$, where $C_{f^{act}} < \infty$ depends on the norms of the parameter matrices of f^{act}, which is implemented as a ReLU MLP (Vaswani et al., 2017). We'll write F for this upper bound for $\|y_j^{(k)}\|_2$. Attention logits are bounded by $A := F^2 C_{f^{att}}$ in the case of dot-product attention, and $A := 2FC_{f^{att}}$ in the case of additive attention. Then any attention weight $\widehat{a}_{j,i} = \exp(a_i)/\sum_j \exp(a_j)$ is upper bounded by $\frac{\exp(A)}{\exp(A)+(n-1)\exp(-A)} \leq \frac{\exp(2A)}{n-1}$.

Choose $C := 2 \cdot (1 + \exp(2A) + L_{f^{act}})$, where $L_{f^{act}}$ is the Lipschitz constant of the ReLU MLP f^{act}. Recall that activations $y_i^{(k)}$ are defined as $f^{act}(y_i^{(k-1)}, b_{i,k,1}, \ldots, b_{i,k,H})$, where $b_{i,k,h}$ equals $\sum_{j=1}^n \hat{a}_{i,j}^{k,h} y_j^{(k-1)}$. We first calculate

$$\|b_{j,k,h} - b'_{j,k,h}\| \leq \sum_{w=1}^{n} \hat{a}_{j,w}^{k,h} \|y_w^{(k-1)} - y_w^{(k-1)'}\|$$
$$= \hat{a}_{j,i}^{k,h} \|y_i^{(k-1)} - y_i^{(k-1)'}\| + \sum_{w \neq i} \hat{a}_{j,w}^{k,h} \|y_w^{(k-1)} - y_w^{(k-1)'}\|$$

which, using the induction hypothesis, is at most:

$$\frac{\exp(2A)}{n-1} C^{2(k-1)} D + \frac{H^{k-1} C^{2(k-1)} D}{n} \leq \frac{H^{k-1} C^{2k-1} D}{n}$$

Plugging this into the definition of $y_i^{(k)}$, the difference $\|y_j^{(k)} - y_j^{(k)}\|$ is at most

$$L_{f^{act}} \cdot \left(\|y_j^{(k-1)} - y_j^{(k-1)'}\| + \sum_{q=1}^{H} \|b_{i,k,q} - b'_{i,k,q}\| \right)$$

First, if $j = i$, this is bounded by (as $n \to \infty$)

$$\leq L_{f^{act}} \cdot \left(C^{2(k-1)} D + o(1) \right) \leq C^{2k} D$$

Second, if $j \neq i$, it is bounded above by

$$(1/n) \cdot L_{f^{act}} \cdot \left(H^{(k-1)} C^{2(k-1)} D + H^k C^{2k-1} D \right)$$

which is bounded by $\leq \frac{H^k C^{2k} D}{n}$. This proves the inductive step for $k > 0$. □

7 Discussion

We have shown that, even with infinite precision, transformers cannot robustly model non-counter-free regular languages, nor basic hierarchical structure. In the hard attention setting, our results hold independently of activation functions and the magnitude of the parameters, and show that no transformer can accurately classify strings as belonging to such languages. In the soft attention setting, our results are slightly less general, but still show that transformers cannot achieve perfect cross-entropies when modeling distributions over these formal languages.

Our results are asymptotic, in the sense that they show that any transformer will make mistakes on modeling PARITY and 2DYCK when the input is *sufficiently long*. A transformer may nonetheless be able to perform well on on short inputs; indeed, given any bound N on the input length, it is possible to construct a transformer that will achieve perfect accuracy or cross-entropy on all examples of length $n \leq N$; our results show that the number of heads and layers, or the parameter norms, will have to increase with N. Practical implementations of transformers might thus be able circumvent such asymptotic limitations by using large numbers of layers and heads, in relation to the sentence lengths typically occurring in natural language. Therefore, pending tighter nonasymptotic bounds, the results reported here need not constitute conclusive evidence for practical limitations of real-world NLP systems.

We believe that the most imminent implications of our results are theoretical in nature. They showcase mathematical techniques for analyzing the capabilities of self-attention, an architecture at the heart of recent advances in NLP. These tools provide theoretical understanding of differences between self-attention and theoretically more well-studied recurrent architectures: Recurrent networks such as LSTMs can perfectly emulate finite-state automata, and therefore can model any finite state language with optimal cross-entropy, as long as the state transition and symbol emission distributions are Markovian. In particular, PARITY of i.i.d. bitstrings can be predicted with perfect accuracy and cross-entropy, independent of the input length. Furthermore, infinite-precision RNNs and LSTMs can model stacks (Tabor, 2000; Grüning, 2006; Kirov and Frank, 2012) and thus are theoretically capable of modeling 2DYCK and other deterministic context-free languages perfectly. The results presented here thus theoretically confirm the intuition that models entirely built on self-attention may have restricted expressivity when compared to recurrent architectures (Tran et al., 2018; Dehghani et al., 2019; Shen et al., 2018a; Chen et al., 2018; Hao et al., 2019). Complementing the asymptotic methods developed here with empirical studies or non-asymptotic extensions is an interesting avenue for future research.

While finite languages are sufficient to model language up to any finite bound on sequence length, it has typically been argued that asymp-

totically more powerful formalisms at the level of context-free grammars or beyond are necessary to properly capture generalizations about the syntax and meaning of natural language (e.g., Chomsky (1957); Shieber (1985)). Our results entail that self-attention is limited in its ability to model context-free languages or evaluate logical formulas. In particular, self-attention cannot in general emulate stacks or arbitrary finite-state automata. Whether this hinders its capacity for syntactic generalization in practice is an interesting question; empirical research suggests that models with strong quantitative performance – both recurrent and transformer models – continue to struggle with syntactic generalization and that quantitative performance metrics such as perplexity can partly be dissociated from syntactic knowledge displayed on more challenging benchmarks (e.g., Kuncoro et al. (2018); Marvin and Linzen (2018); Tran et al. (2018); McCoy et al. (2019)).

Nonetheless, the success of transformers across NLP tasks suggests that many aspects of natural language can be modeled well with methods that are formally too weak for the formal languages typically assumed in theoretical linguistics. Beyond general limitations of asymptotic analysis, a possible reason for this phenomenon is that language uses recursive structure only in restricted ways due to cognitive factors. For instance, it has long been noted that center embeddings, syntactic structures exhibiting iterated bracketing, are very challenging for humans to process (Miller and Chomsky, 1963; Gibson and Thomas, 1999). Intriguingly, self-attention bears some resemblance to psycholinguistic models of memory in human sentence processing that assume that humans, while processing a word, attend to chunks that were stored in memory when processing some previous words (Lewis and Vasishth, 2005; Parker et al., 2017). Such processing models predict difficulty with center embedding because they cannot count brackets (Lewis and Vasishth, 2005), akin to what we have shown theoretically for neural network models based on self-attention.

8 Conclusion

We formally investigated the capabilities of self-attention in modeling regular languages and hierarchical structure. We showed that transformers cannot model periodic regular languages or basic recursion, either with hard or soft attention, and even if infinite precision is allowed. This entails that self-attention cannot in general emulate stacks or general finite-state automata. Our results theoretically confirm the idea that self-attention, by avoiding recurrence, has quite limited computational power.

Acknowledgments

I thank Dan Jurafsky and the members of the Stanford NLP Group for helpful comments.

References

David A Mix Barrington, Kevin Compton, Howard Straubing, and Denis Thérien. 1992.

Regular languages in NC1. *Journal of Computer and System Sciences*, 44(3):478–499.

Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Jean-Philippe Bernardy. 2018. Can recurrent neural networks learn nested recursion? *LiLT (Linguistic Issues in Language Technology)*, 16(1).

Ravi B Boppana. 1997. The average sensitivity of bounded-depth circuits. *Information processing letters*, 63(5):257–261.

Bo Cartling. 2008. On the implicit acquisition of a context-free grammar by a simple recurrent neural network. *Neurocomputing*, 71(7-9):1527–1537.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86, Melbourne, Australia. Association for Computational Linguistics.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas. Association for Computational Linguistics.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Noam Chomsky. 1957. *Syntactic structures*. Mouton, The Hague.

Noam Chomsky and Marcel P Schützenberger. 1963. The algebraic theory of context-free languages. In *Studies in Logic and the Foundations of Mathematics*, volume 35, pages 118–161. Elsevier.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? An analysis of BERT's attention. In *Proceedings of BlackboxNLP 2019*.

Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Giacomo De Palma, Bobak Toussi Kiani, and Seth Lloyd. 2018. Deep neural networks are biased towards simple functions. *arXiv preprint arXiv:1812.10156*.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Martin BH Everaert, Marinus AC Huybregts, Noam Chomsky, Robert C Berwick, and Johan J Bolhuis. 2015. Structures, not strings: linguistics as part of the cognitive sciences. *Trends in cognitive sciences*, 19(12):729–743.

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

Felix A Gers and Jürgen Schmidhuber. 2001. LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6):1333–1340.

Edward Gibson and James Thomas. 1999. Memory limitations and structural forgetting: The perception of complex ungrammatical sentences as grammatical. *Language and Cognitive Processes*, 14(3):225–248.

Parikshit Gopalan, Noam Nisan, Rocco A Servedio, Kunal Talwar, and Avi Wigderson. 2016. Smooth boolean functions are easy: Efficient algorithms for low-sensitivity functions. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 59–70. ACM.

André Grüning. 2006. Stack-like and queue-like dynamics in recurrent neural networks. *Connection Science*, 18(1):23–42.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Jie Hao, Xing Wang, Baosong Yang, Longyue Wang, Jinfeng Zhang, and Zhaopeng Tu. 2019. Modeling recurrence for transformer. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1198–1207, Minneapolis, Minnesota. Association for Computational Linguistics.

Johan Hastad, Ingo Wegener, Norbert Wurm, and Sang-Zin Yi. 1994. Optimal depth, very small size circuits for symmetrical functions in AC0. *Information and Computation*, 108(2):200–211.

Yu-Lun Hsieh, Minhao Cheng, Da-Cheng Juan, Wei Wei, Wen-Lian Hsu, and Cho-Jui Hsieh. 2019. On the robustness of self-attentive models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1520–1529, Florence, Italy. Association for Computational Linguistics.

Yvonne Kalinke and Helko Lehmann. 1998. Computation in recurrent neural networks: From counters to iterated function systems. In *Australian Joint Conference on Artificial Intelligence*, pages 179–190. Springer.

Christo Kirov and Robert Frank. 2012. Processing of nested and cross-serial dependencies: an automaton perspective on SRN behaviour. *Connection Science*, 24(1):1–24.

Samuel A. Korsky and Robert C. Berwick. 2019. On the computational power of RNNs. *arXiv preprint arXiv:1906.06349*.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. Lstms can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436.

Richard L Lewis and Shravan Vasishth. 2005. An activation-based model of sentence processing as skilled memory retrieval. *Cognitive science*, 29(3):375–419.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open Sesame: Getting inside BERT's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253, Florence, Italy. Association for Computational Linguistics.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *International Conference on Learning Representations*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

R Thomas McCoy, Junghyun Min, and Tal Linzen. 2019. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance. *arXiv preprint arXiv:1911.02969*.

Robert McNaughton and Seymour A Papert. 1971. *Counter-Free Automata (MIT research monograph no. 65)*. The MIT Press.

William Merrill. 2019. Sequential neural networks as automata. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 1–13, Florence. Association for Computational Linguistics.

George A Miller and Noam Chomsky. 1963. Finitary models of language users. In R. Duncan Luce, Robert R. Bush, and Eugene Galanter, editors, *Handbook of Mathematical Psychology*, pages 419–492. John Wiley.

John Miller and Moritz Hardt. 2019. Stable recurrent models. In *International Conference on Learning Representations*.

Michael Mitzenmacher and Eli Upfal. 2017. *Probability and Computing*, 2nd edition. Cambridge University Press, Cambridge.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Dan Parker, Michael Shvartsman, and Julie A Van Dyke. 2017. The cue-based retrieval theory of sentence comprehension: New findings and new challenges. *Language processing and disorders*, pages 121–144.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Jorge Pérez, Javier Marinković, and Pablo Barceló. 2019. On the Turing completeness of modern neural network architectures. In *International Conference on Learning Representations*.

Luzi Sennhauser and Robert Berwick. 2018. Evaluating the ability of LSTMs to learn context-free grammars. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 115–124.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018a. Disan: Directional self-attention network for RNN/CNN-free language understanding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Sen Wang, and Chengqi Zhang. 2018b. Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. In *IJCAI'18 Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4345–4352.

Stuart M Shieber. 1985. Evidence against the context-freeness of natural language. In *Philosophy, Language, and Artificial Intelligence*, pages 79–89. Springer.

Hava Siegelman and Eduardo D Sontag. 1995. On the computational power of neural nets. *Journal of Computer and System Sciences*, 50:132–150.

Natalia Skachkova, Thomas Trost, and Dietrich Klakow. 2018. Closing brackets with recurrent neural networks. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 232–239.

Mirac Suzgun, Yonatan Belinkov, and Stuart M Shieber. 2019. On evaluating the generalization of LSTM models in formal languages. *Proceedings of the Society for Computation in Linguistics (SCiL)*, pages 277–286.

Whitney Tabor. 2000. Fractal encoding of context-free grammars in connectionist networks. *Expert Systems*, 17(1):41–56.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4593–4601.

Ke Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of being recurrent for modeling hierarchical structure. In *Proceedings of the 2018 Conference on Empirical Methods*

in Natural Language Processing, pages 4731–4736, Brussels, Belgium. Association for Computational Linguistics.

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

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5797–5808.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. On the practical computational power of finite precision rnns for language recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 740–745.

Baosong Yang, Longyue Wang, Derek F. Wong, Lidia S. Chao, and Zhaopeng Tu. 2019. Assessing the ability of self-attention networks to learn word order. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3635–3644.