On the Expressivity of Selective State-Space Layers: A Multivariate Polynomial Approach

Edo Cohen-Karlik * **Itamar Zimerman** * **Liane Galanti** *

Ido Andrew Atad **Amir Globerson** **Lior Wolf**

Blavatnik School of Computer Science and AI, Tel Aviv University

Abstract

Recent advances in efficient sequence modeling have introduced selective state-space layers, a key component of the Mamba architecture, which have demonstrated remarkable success in a wide range of NLP and vision tasks. While Mamba's empirical performance has matched or surpassed SoTA transformers on such diverse benchmarks, the theoretical foundations underlying its powerful representational capabilities remain less explored. In this work, we investigate the expressivity of selective state-space layers using multivariate polynomials, and prove that they surpass linear transformers in expressiveness. Consequently, our findings reveal that Mamba offers superior representational power over linear attention-based models for long sequences, while not sacrificing their generalization. Our theoretical insights are validated by a comprehensive set of empirical experiments on various datasets.

1 INTRODUCTION

Sequence modeling has been the focus of many works in recent years, with remarkable results enabling applications such as ChatGPT. To date, these models are largely based on the Transformer architecture (Vaswani et al., 2017). While transformers have proven extremely effective, they suffer from several drawbacks compared to traditional recurrent models, one of which is their computational complexity which scales quadratically with the input sequence length.

In attempt to mitigate the computational inefficiency of sequence modeling of transformers, a line of work has attempted to resurrect RNNs, Gu et al. (2021a); Gupta et al. (2022a); Gu et al. (2021b) have introduced a series of architectures called State Space Models (SSMs) which include a linear recurrence that admits efficient computations and special structure on the transition matrices.

While these architectures have demonstrated impressive performance in long sequence modeling, their performance on fundamental tasks such as language modeling fall short compared to transformers, mostly due to intrinsic superiority of transformers in modeling interactions between different elements of the input sequence. Recent work (Gu and Dao, 2023), has introduced Mamba, an SSM variant with Selective SSMs (S6) as its core block. In an S6 layer, the parameters are a function of the input, providing the SSM with content awareness. The empirical success of Mamba is undeniable, with applications spanning large-scale language modeling (Zuo et al., 2024; Lieber et al., 2024; Waleffe et al., 2024), image (Liu et al., 2024b), and video (Li et al., 2025b) processing, medical imaging (Liu et al., 2024a), tabular data analysis (Ahamed and Cheng, 2024), Reinforcement Learning (Lv et al., 2024), point-cloud analysis (Liang et al., 2024), graph processing (Wang et al., 2024), and N-dimensional sequence modeling (Li et al., 2024).

The success of Mamba models across various domains ignites interest in their theoretical properties. Establishing a comprehensive theoretical understanding is crucial, as it enhances our knowledge of these layers, promotes their adoption within the research community, and paves the way for future architectural advancements. Additionally, since S6 can be considered a variant of attention with linear complexity (Ali et al., 2025; Zimerman et al.), deeper theoretical insights could elucidate the relationships between gated RNNs, transformers, and SSMs, thereby advancing our knowledge of these interconnected architectures. Initial efforts to establish a theoretical framework for the expressiveness of selective (and non-selective) SSM layers have been undertaken by several researchers. Using Rough Path

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

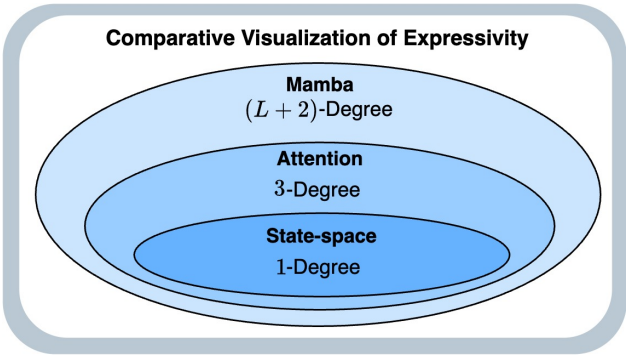

Figure 1: **Expressivity via Polynomial Degree:** Our characterization of SSMs, S6 layers, and causal self-attention via multivariate polynomials allows us to identify the expressiveness gap between these layers through maximal polynomial degree.

Theory, (Cirone et al., 2024) demonstrated that diagonal selective SSMs, such as Mamba, possess less expressive power than their non-diagonal counterparts. Additionally, (Merrill et al., 2024) investigated the expressiveness relationships between SSM variants and transformers using the lens of circuit complexity, revealing that both models share the same expressive power (belonging to TC^0). Finally, (Jelassi et al., 2024) examined the copying ability of various SSM variants compared to transformers. It concluded that from both theoretical and empirical perspectives SSMs struggle with this task. While these significant studies highlight the limited expressive capabilities of Mamba models compared to other architectures, our work introduces a different trend. We demonstrate the superior expressive power of S6 layers, using a theoretical framework based on multivariate polynomials.

In addition to exploring the expressiveness gap between transformers and S6 layers, we develop norm-based length-agnostic generalization bounds for S6 layers. Suggesting that the added expressivity of the selective mechanism does not hinder the generalization properties compared to traditional SSMs as studied in (Liu and Li, 2024), and that while polynomial expressivity increases with sequence length, it does not impact generalization.

Our main contribution encompasses the following aspects: (i) We present simplified polynomial variants of Mamba that exhibit comparable performance on NLP and vision tasks, promoting a simpler model that can serve as a foundation for other theoretical contributions, as well as shed light on the inner dynamics of Mamba and its critical components. (ii) By establishing the connection between multivariate polynomials and S6 layers, we theoretically prove that S6 layers are expressive as a linear self-attention with depth that scales logarithmically with the sequence length; that is, there are functions that can be modeled with a single S6 that would require a logarithmic number of linear attention layers. More generally, we establish that a Mamba model with only 4 layers is sufficient to represent the class of multivariate polynomials with bounded degree L. In contrast, a linear attention-based model requires a logarithmic number of layers in L to achieve the same representational capability. (iii) Through experiments on a synthetic dataset in a controlled environment designed to isolate expressivity issues, we empirically validate that our theory is reflected in practice. (iv) Finally, although S6 has better polynomial expressivity for long sequences, we show that this does not come at the cost of limited generalization, by proving the first length-agnostic generalization bound for S6 . This leads us to conclude that S6 layer possesses superior theoretical properties over linear attention for long-range tasks.

2 RELATED WORK

SSMs have gained a lot of traction due to their remarkable performance and computational efficiency. Their theoretical properties have been the focus of many recent works; Merrill et al. (2024) compare the expressive capacity of (non-selective) SSMs and transformers, concluding that both architectures belong to the same complexity class (TC^0).[1] Sarrof et al. (2024) conduct a more refined analysis and show that transformers and SSMs occupy different portions of TC^0. Another work showed that a single-layer Transformer with N heads can simulate any state space model with N channels (Zimerman and Wolf, 2023).

Cirone et al. (2024) show that the selective mechanism introduced in Mamba results with more expressive architectures compared to traditional (non-selective) SSMs. Ali et al. (2025) show that there are functions that S6 can implement while transformers cannot.

In this work we compare the expressive power of S6 to those of transformers. We show that under certain assumptions which we justify empirically, a constant number of S6 layers are dense in the polynomial function space while transformers and non-selective SSMs are far less expressive. In addition to discussing expressivity, we also provide the first norm-based generalization bound for S6. Relevant related works are detailed in Appendix C.

3 BACKGROUND

In this section we present the technical details required for the theoretical analysis and provide relevant notations.

Notations Let $X \in \mathbb{R}^{D\times L}$ be an input sequence of length L with dimension size D, denote the element in the i-th channel and position j as X_{ij}. We denote the entire channel at a specific position j and the entire sequence at a specific

[1] TC^0 is the complexity class that can be decided by polynomial-sized Boolean circuits.

channel i as X_{*j} and X_{i*}, respectively. For simplicity, we denote a general single channel of X without channel index by $x := (x_1, x_2, \cdots, x_L)$ such that $x_i \in \mathbb{R}$.

Mamba Given these notations, a Mamba block, which is built on top of the S6 layer, is specified as follows:

$$X = \sigma(\text{Conv1D}(\text{Linear}(U)), \quad Z = \sigma(\text{Linear}(U)) \tag{1}$$

$$Y = \text{S6}(X), \quad \hat{Y} = \text{Linear}(Y \otimes Z) \tag{2}$$

Here, U is the input to the Mamba block, and X is the input to the S6 layers, $X, Z, Y, \hat{Y} \in \mathbb{R}^{L \times D}$, the linear layers operate independently for each sequence element, σ represents SiLU activation function, and $\otimes$ denotes element-wise multiplication with the gate branch in Eq. 2 (Right).

S6 An S6 layer is a recent variant of SSM. A standard diagonal SSM is parameterized by a diagonal transition matrix $A \in \mathbb{R}^{N \times N}$, input and output matrices $B, C \in \mathbb{R}^{N \times 1}$, and a timescale $\Delta \in \mathbb{R}$. An input scalar sequence x is mapped to an output scalar sequence y via the following recurrent rule:

$$h_t = \bar{A}\, h_{t-1} + \bar{B}\, x_t\,, \quad y_t = C\, h_t\,, \bar{A} = f_A(A, \Delta) \tag{3}$$

$$\bar{B} = f_B(B, \Delta)$$

where f_A, f_B are discretization functions, and the discrete system matrices are $\bar{A} \in \mathbb{R}^{N \times N}$ and $\bar{B} \in \mathbb{R}^{N \times 1}$. The recurrent rule in Eq. 3 can be computed efficiently in parallel on modern hardware accelerators using work-efficient parallel scans (Smith et al., 2022) or a simple scalar convolution via FFTs (Gu et al., 2021b). Note that Eq. 3 is a map from $\mathbb{R}^L$ to $\mathbb{R}^L$, and to process D channels, multiple independent instances are used.

Contrary to traditional SSMs, which utilize time-invariant system matrices and process each channel independently, S6 layers incorporate a data-dependent mechanism that is parameterized by $S_B, S_C \in \mathbb{R}^{N \times D}$, $A \in \mathbb{R}^{D \times N}$ and $S_\Delta \in \mathbb{R}^{1 \times D}$ to define the time-variant matrices as follows:

$$B_t = S_B X_{*t},\ C_t = S_C X_{*t},\ \Delta_t = \text{softplus}(S_\Delta X_{*t}) \tag{4}$$

$$\bar{A}_t = \exp(\Delta_t A), \quad B_t = \Delta_t B_t \tag{5}$$

The resulting time-variant recurrent rule is:

$$h_t = \bar{A}_t h_{t-1} + \bar{B}_t x_t, \quad y_t = C_t h_t \tag{6}$$

Our analysis focuses on the regime of 'many-to-one', which deals with models that operate on sequences and produce a single output after processing the entire input sequence.

4 THEORETICAL RESULTS

We begin by presenting our simplified model in Sec. 4.1, which forms the basis for our theory on the expressivity and generalization of S6 layers, discussed in Sec.4.2 and Sec. 4.3, respectively. In the experiments section we demonstrate that the simplified model used in our exposition achieves comparable results to those of the original S6 layers, encouraging further exploration of this variant.

4.1 Model Simplifications

The original S6 layer is parameterized by S_Δ, S_B, S_C and A, and it is defined in Eqs. 4 and 6.

Our approach utilizes a simplified model with a single channel described below:

$$\bar{B}_i = S_B x_i\,,\ C_i = S_C x_i\,,\ \Delta_i = p_1(S_\Delta x_i)\,, \tag{7}$$

$$\bar{A}_i = p_2(\Delta_i A) \tag{8}$$

where p_1 and p_2 represent polynomials that operate independently per element, for instance, a second-degree Taylor approximation for softplus and exponent accordingly. An equivalent model would be:

$$C_i = S_C x_i, \quad \bar{B}_i = S_B x_i, \quad \bar{A}_i = p_2\left(p_1\left(\frac{S_\Delta x_i}{\sqrt{D}}\right) A\right) \tag{9}$$

in which D is the width of the model, and $\frac{1}{\sqrt{D}}$ is a constant normalization factor. This model can be interpreted as state-space layer without discretization, with $\bar{A}$ being selective, and p_1 and p_2 are stabilizers designed to control the values of A, which must be positive. It is imperative to note that standard SSMs without discretization are both effective and simple (Gupta et al., 2022b).

For simplicity, we define an additional polynomial $p_A(x) = p_2\left(p_1\left(\frac{x}{\sqrt{D}}\right) A\right)$ which is parameterized by A and ties p_1 and p_2. Hence, we can denote $\bar{A}_i = p_A(S_\Delta x_i)$.

Alternatively we can utilize the following simplified non-polynomial model:

$$\bar{B}_i = S_B x_i, \quad C_i = S_C x_i, \tag{10}$$

$$\Delta_i = \text{softplus}(S_\Delta x_i), \quad \bar{A}_i = \exp(\Delta_i A) \tag{11}$$

4.2 Expressivity

We identify an expressivity gap between S6 and attention via multivariate polynomials. This gap is delineated through Thm. 1 and Thm. 2. The former establishes that attention models require $O(\log L)$ layers to represent L-degree multivariate polynomials, whereas Mamba models can express such polynomials within a single layer. Furthermore, Thm. 2 extends this finding by demonstrating that the expressiveness gap is not limited to anecdotal or edge-case polynomials. Rather, it encompasses a broad spectrum of polynomial functions.

Theorem 1. *(informal) Consider a simplified S6 layer (see Sec. 4.1) and single Transformer layer, both with hidden dimension N. For input sequences of length $L \geq 3$, a single layer of simplified S6-based Mamba is logarithmically more expressively efficient in depth compared to a single causal linear self-attention layer with a single head and polynomial activations instead of softmax.*

We consider this theorem relatively surprising, as transformers are considered highly expressive models, in contrast to state-space layers, which are traditionally constrained. The proof follows from Lemma 1 and Lemma 2 which are presented next.

Lemma 1. *There exists a function $f : \mathbb{R}^L \to \mathbb{R}$ that can be implemented by one channel of a simplified (see Sec. 4.1) S6 such that a single linear attention head would require at least $O(\log L)$ layers to express f.*

Lemma 2. *Any function that can be expressed by a single causal linear attention head can be expressed by a simplified (see Sec. 4.1) S6 layer.*

For the complete details and proof of Lemma 2, we refer the reader to Appendix B . As for Lemma 1, we present here a proof sketch for a simplified version (see Lemma 1 in the appendix for the complete proof). In this simplified case, our proof focuses exclusively on linear-attention. Additionally, we consider models that process scalar sequences.

Proof of Lemma 1 (without Softmax). Our proof relies on the characterization of the hypothesis classes that are realizable through S6 and self-attention via multivariate polynomials. We start by presenting this formulation for S6:

Single S6 Layer as Multivariate Polynomials One channel of the variant of the S6 layer we consider is described in detail in Eq. 9. Since we are interested in identifying the minimal polynomial degree required to characterize models that employ Eq. 9, we can assume that P_A is linear. Thus, we can incorporate the coefficients of P_A into S_Δ, namely, $\bar{A}_t = S_\Delta x_t$. Consequently, Eq.9 can be unrolled as follows:

$$y_t = S_C S_B \sum_{j=1}^{t} {S_\Delta}^{t-j-1} \left(\Pi_{k=j+1}^{t-1} x_k \right) x_t {x_j}^2 \tag{12}$$

Hence, we can characterize the last output y_L as a multivariate polynomial with at least L monomials, and a maximal degree of at least $L + 2$. Denote $c_j = S_C S_B S_\Delta^{L-j-1}$,

$$y_t = \sum_{j=1}^{t} c_j \left(\Pi_{k=j+1}^{t-1} x_k \right) x_t {x_j}^2 \tag{13}$$

Attention as Multivariate Polynomials Given the parameters of the layers $W_Q, W_K, W_V \in \mathbb{R}$, the last element in the output sequence can be computed by:

$$y = \frac{(xW_Q)(xW_K)^T}{\sqrt{d_k}}(xW_V) \tag{14}$$

$$y_t = \frac{W_Q W_K W_V}{\sqrt{d_k}} \sum_{j=1}^{t} {x_j}^2 x_t = \sum_{j=1}^{t} c_j {x_j}^2 x_t \tag{15}$$

where $c_j = \frac{W_Q W_K W_V}{\sqrt{d_k}}$. Hence, we can characterize the last output element y_L of a self-attention layer by a 3-degree multivariate polynomial with L monomials.

N-Stacked Attention Layers as Multivariate Polynomials In light of the characterization of one head of a self-attention layer, we can now proceed to establish the characterization of N-stacked self-attention layers. Since the composition of multivariate polynomials results in another multivariate polynomial, where the maximal degree of the resulting polynomial amounts to the product of the maximum degrees of the composed polynomials, we can prove by induction that N-Stacked layers can be represented by a polynomial with a maximal degree of 3^N.

Identify and Refine the Expressivity Gap The description of one head of a self-attention layer and one channel of an S6 via multivariate polynomials reveals the expressiveness gap in terms of the maximal degree, which is 3 for self-attention and $L + 2$ for S6, as depicted in Fig. 1. Thus, our formulation presents a broad family of functions that can be implemented by selective SSMs but cannot be realized by a single self-attention head. Furthermore, when processing a sequence of length L, it is clear that in order to express a function realized by S6 using N-stacked self-attention layers, $O(\log L)$ stacked layers are required. □

Single SSM as Multivariate Polynomials For completeness, we formalized a single standard SSM (not S6) via multivariate polynomials. Since SSMs can be represented as a single non-circular convolution between the input x and a kernel $\bar{K} = (C\bar{B}, C\bar{A}\bar{B}, \ldots, C\bar{A}^{L-1}\bar{B})$, it is evident that the last output element of the SSM can be expressed by a 1-degree multivariate polynomial consisting of L monomials.

Sharpening the Expressivity Gap Thm 1 establishes the existence of an expressivity gap. However, it remains unclear how many polynomials are encompassed within this gap, and whether they constitute a significant portion of the function class or are merely anecdotal instances. To refine our separation results , we now quantify the expressiveness gap by analyzing the number of layers required to represent the entire class of L-degree multivariate polynomials. As established previously, attention-based models necessitate $O(\log L)$ layers. In contrast, the following theorem demonstrate that a Mamba model with just 4-stacked layers and a sufficiently large number of channels can represent *any multivariate polynomial of arbitrary degree*, thereby highlighting the significant expressiveness gap, which constitute a significant portion of the class of L-degree polynomials.

Theorem 2. *Given an input scalar sequence $x \in \mathbb{R}^L$, a model with four stacked simplified S6-based Mamba layers, a sufficiently large number of channels, learnable positional encoding (PE), sufficient padding and a linear encoder layer at the first layer can express any multivariate polynomial with bounded degree of x.*

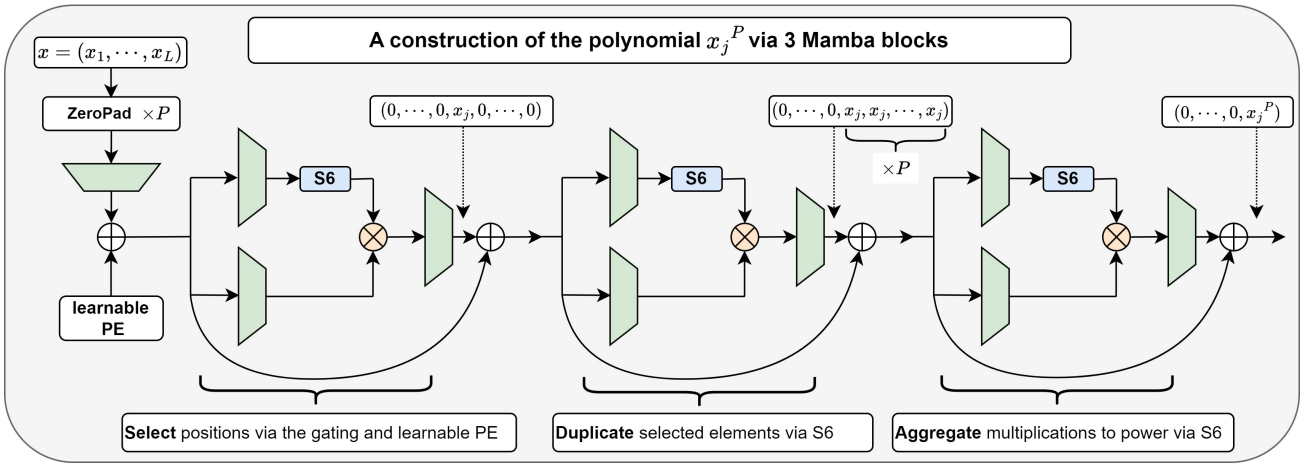

Figure 2: Visualization of 3-stacked Mamba layers expressing monomials of a univariate polynomial, as formulated in Lemma 3. To simplify the visualization, the Conv1D layer has been omitted.

For simplicity, we assume that the state size $N = 1$ and the SiLU activations in Mamba are removed. We justify these decisions as they only restrict our model. The proof is presented in appendix B, and it follows a technical construction that demonstrates how to express a general polynomial using Mamba. The core argument is built on the following lemma, which is also visualized in Fig. 2:

Lemma 3. *Given an input scalar sequence $x \in \mathbb{R}^L$, for any j, a model M with 3-Mamba layers, a sufficiently large number of channels, learnable PE, and a linear encoder in the first layer can express any monomial of a univariate polynomial in x_j. Specifically, for any constants $c \in \mathbb{R}$ and $P \in \mathbb{R}$, there exists a configuration of M such that the output of the k-th channel $M(x)_k = c \cdot x_j^P$ for $k > P + j$.*

For a detailed proof of Lemma 3, we refer the reader to the appendix. Here, we provide an intuitive explanation of the proof, which hinges on the following two key capability of the Mamba architecture:

(i) Per-position selection: By utilizing Mamba's auxiliary components, including the gating branch, linear layers, and learnable PE, each Mamba layer can isolate a specific channel k at a given position j. Notably, the output of the first Mamba block can effectively filter out all unnecessary positions, producing a sequence mask with zeros at every position except $i = j$, which contains x_j at position j. This is done by setting the S6 to the identity function ($\bar{A} = 0, \bar{B} = \bar{C} = 1$), ensuring x_j is not modified. Additionally, mask the other positions achieved by set the parameters of a learnable PE at one of the channels to the indicator function $\mathbb{I}_{=j}$, which is 1 only when focusing on the j-th position, and ensuring this PE passed into the gate branch through the linear layers.

(ii) Express powers by aggregate multiplications of duplicated elements: Given an input $x = (0, \cdots, x_j, 0, \cdots)$, which can be obtained from the first layer, the second Mamba block can duplicate the value of x_j exactly $P - 2$ times. This duplication holds even if $P > L$, thanks to the ZeroPad component. The duplication process is achieved by setting the linear layers to identity mappings and utilizing a degenerate single SSM channel where the system matrices always equal $\bar{A}, \bar{B}, \bar{C} = 1$. Therefore, if the k-th channel receives an input sequence $x = (0, \cdots, x_j, 0, \cdots)$, it will output $x = (0, \cdots, x_j, \cdots, x_j)$. Through the gating mechanism, learnable PE (which can pass through skip-connections to the subsequent layers), and biases in the linear layer, the entire block can then produce a filtered output $z = (1, \cdots, 1, x_j, \cdots, x_j, 1, \cdots, 1)$, ensuring that there are exactly $P - 2$ copies of x_j. Next, the S6 layer in the third block can multiply the sequence elements in z via the operation described in Eq. 13, which include the term $\Pi_{k=j+1}^{t} \bar{A}_k$. This term yields an output sequence with the values $(1, \cdots, 1, x_j^3, \cdots, x_j^P, \cdots)$. To specifically isolate $x_j{}^P$, we begin by generating all unwanted elements by applying the same SSM to the sequence z, introducing an additional zero at the initial occurrence of x_j denoted by z'. We then subtract the outputs from these two SSM channels at the final linear layer of the block. This subtraction yields a telescoping series $SSM(z) - SSM(z')$:

$$\sum_{j=1}^{t} c_j \left(\Pi_{k=j+1}^{t-1} x_k \right) {x_t}^2 {x_j}^2 - \sum_{j=2}^{t} c_j \left(\Pi_{k=j+1}^{t-1} x_k \right) {x_t}^2 {x_j}^2 \quad (16)$$

that effectively eliminates any term except for $x_j{}^P$.

Finally, it is crucial to highlight that incorporating auxiliary components into the Transformer, such as learnable PE and gating, does not mitigate the logarithmic increase in depth required with L. This limitation arises because the expressible degree within each block remains unchanged, thereby leading to the observed asymptotic behavior.

4.3 Generalization

Our theoretical analysis in Sec. 4.2 demonstrates the superior expressive power of S6 compared to linear attention,

particularly in terms of multivariate polynomial degree and long-range processing capabilities. Furthermore, Thm. 2 and Lemma 1 establishes that the hypothesis class associated with Mamba models with just four layers is significantly larger than that of transformers, even with greater depth. While increased expressivity can often come at the cost of generalization, in this section, we show that S6 layers do not suffer from this trade-off. To do so, we provide a generalization bound for simplified S6 layers, as defined in Eq. 42, which is a generalization of Eq. 10. Our analysis is based on a classifier $f : \mathbb{R}^{D\times L} \to \mathbb{R}^{\mathcal{C}}$, parameterized by (A, S_B, S_C, S_Δ). The classifier f for a specific class $c \in [\mathcal{C}]$ denoted by $f^c(X_{*1}, ..., X_{*L})$ is computed as:

$$\sum_{d=1}^{D} W_{c,d} \left(S_C X_{*L}\right)^T \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \bar{A}_{dk} \right) S_B X_{*i} X_{di} \qquad (17)$$

where $\mathcal{C}$ is the number of classes, $\bar{A}_{dk} = \exp(\Delta_k A_d)$ (A_d is the dth row of A) as defined in Eq. 4 and $W \in \mathbb{R}^{\mathcal{C}\times D}$ represents a linear projection from the output to the number of classes. We denote the parameters of a classifier by $w = (A, S_B, S_C, S_\Delta, W)$ and the corresponding function induced by a specific instance of w by f_w. The class of functions taking on different parameter instances w, is denoted by $\mathcal{F}$. As customary in the derivation of Rademacher complexity based bounds (e.g. (Golowich et al., 2018)), our analysis takes into account the (different) norms of the parameters, for which we denote:

$$\begin{aligned} \rho_W(w) = ||W||_F,\ \rho_A(w) = \|A\|_{\max},\ \rho_B(w) = \|S_B\|_{2,\infty}, \\ \rho_C(w) = \|S_C^{(h)}\|_F,\ \rho_\Delta(w) = \|S_\Delta\|_2, \\ \Gamma(w) = \rho_W(w) \cdot \rho_A(w) \cdot \rho_B(w) \cdot \rho_C(w) \cdot \rho_\Delta(w) \end{aligned} \qquad (18)$$

Equipped with these notations, we are now ready to state our main generalization bound.

Theorem 3. *Let P be a distribution over $\mathbb{R}^{D\times L} \times [C]$ and $\delta > 0$. Let $S = \{(X_{(j)}, y_{(j)})\}_{j=1}^m$ be a dataset of i.i.d. samples selected from P, where each $X_{(j)} = (X_{(j)_{*1}}, \ldots, X_{(j)_{*L}}) \in \mathbb{R}^{D\times L}$. Assume that $\forall j \in [m]$: $||X_{(j)}||_{\max} \leq 1$. Additionally, suppose $\forall k \in [L], d \in [D]$: $||\bar{A}_{dk}||_{\max} < K < 1$. Then, with probability at least $1-\delta$ over the selection of S, for any $f_w \in \mathcal{F}$,*

$$\begin{aligned} &\text{err}_P(f_w) - \frac{1}{m}\sum_{j=1}^{m} \mathbb{I}[\max_{c\neq c'}(f_w^c(X_{(j)})) + \gamma \geq f_w^{c'}(X_{(j)})] = \\ &\text{err}_P(f_w) - \text{err}_S^\gamma(f_w) \leq \frac{2\sqrt{2}}{\gamma m}(\Gamma(w) + \frac{1}{D^2N^2})D^2 \cdot \\ &(1 + \sqrt{2\log(4LCD^4N)}) \sqrt{\max_{t,k} \sum_{j=1}^{m} (X_{(j)_{tk}})^2} \frac{K}{(K-1)^2} + \\ &3\sqrt{\frac{\log(2/\delta) + 2\log(D^2N^2\Gamma(w)+2)}{2m}}, \end{aligned} \qquad (19)$$

where the maximum is taken over $t \in [D]$, $k \in [L]$.

See Appendix 5 for the complete details and proof. A uniform convergence bound for a class $\mathcal{F}$ is an upper bound on the generalization gap that uniformly holds for all $f \in \mathcal{F}$. The more direct Rademacher complexity bound presented in Lemma 4 is an example of a uniform convergence bound. However, these bounds become ineffective when a function $f \in \mathcal{F}$ exists that can perfectly fit any labeling of the dataset. In such situations, uniform convergence bounds fail to provide meaningful insights and are considered vacuous.

In contrast, the bound derived in Thm. 3 is also based on Rademacher complexity, but it is not a uniform convergence bound and hence it is not inherently vacuous. In the proof, we partition the class $\mathcal{F}$ into subsets $\mathcal{F}_\rho$, where the partitioning criterion ρ is based on the norm of the S6 matrices, and apply our Rademacher bound (see Appendix 4) within each subset. We then apply a union bound to combine these results, obtaining a bound individualized for each $f_w \in \mathcal{F}$.

To understand our bound, we analyze its terms. First note that from the standard assumption that the data is normalized, we have,

$$\sqrt{\max_{t,k} \sum_{j=1}^{m} (X_{(j)_{tk}})^2} \leq \sqrt{m} \qquad (20)$$

Regarding the term $\sqrt{\log(\mathcal{C}DNL)}$, even when choosing exceptionally large values for D or L, such as 2^{100}, the impact on the bound remains minimal. The second term in the bound (see Thm 3) is typically smaller than the first term, as it scales with $\sqrt{\log(DN\Gamma(w))/m}$. Therefore, we conclude that our generalization bound scales with $\mathcal{O}\left(\frac{1}{\sqrt{m}} D^2 \Gamma(w) \cdot \frac{K}{(K-1)^2}\right)$. This implies that the bound is largely unaffected by the sequence length L, making it applicable to various sequence lengths without being significantly impacted by them. Note that since $K < 1$, when K is small, the term $\frac{1}{1-K}$ approaches 1, further reducing its impact on the bound. This implies that for very small K, the generalization bound becomes even tighter.

5 EXPERIMENTS

In this section, we extensively validate our theorems and assumptions through empirical analysis. First, in Sec.5.1, we demonstrate that our simplified variant of the Mamba layer achieves performance comparable to the original Mamba layer when incorporated into standard settings and deep networks, thereby justifying the exploration of this variant. Next, in Sec.5.2, we validate our theory on expressiveness by showing that self-attention struggles to learn high-degree multivariate polynomials, which S6 can model effectively.

5.1 Model Justification

Our theoretical study employs the simplified S6 variant described in Eq. 9. We conduct experiments in both the NLP and vision domains, evaluating this variant when integrated into the Mamba backbone, with the goal of showing that it performs similarly to the original S6 layer.

NLP In NLP, we trained variants of our simplified S6 layer within Mamba backbones on the Wikitext-103 dataset using a self-supervised scheme for *Next Token Prediction*. Our models feature 12 layers with a hidden dimension size of 386 and were trained with a context length of 1024 tokens. The final results are detailed in right panel of Tab. 1 and in the right panel of Fig. 3, which illustrates the evolution of test-perplexity across epochs. Evidently, our simplified S6 variant performs well in the NLP domain, with only a slight reduction in perplexity with respect to the original model. Specifically, the polynomial variant achieved a perplexity score of 26.42, 0.69 points lower than its original baseline score of 25.73. In contrast to the polynomial variant, the other simplified variants that employ $\bar{B}_i = B_i$ achieve a slightly lower perplexity compared to the baseline, highlighting the significance of this aspect in the architecture.

Vision Image classification experiments are conducted on the ImageNet-100 benchmark. We built upon the Vision-Mamba (ViM) architecture (Liu et al., 2024b), replacing the S6 layers with our simplified variant while maintaining the same training procedures and hyper-parameters. The left panel of Tab. 1 presents the results: the simplified variant from Eq. 9 achieves a accuracy of 78.62%, which is 2.4% lower than the original model which achieve a score of 81.02. For reference, we include the results of DeiT (Touvron et al., 2021), which achieved a top-1 accuracy of 78.21% for the same model size (Baron et al., 2024).

To identify which aspects of our simplification most significantly impact performance, we compare two additional variations. First, we use a polynomial S6 variant without omitting the discretization. Second, we run a vanilla non-polynomial model where be $\bar{B}_i = B_i$. Our empirical analysis reveals that the polynomial model performs remarkably well, achieving an accuracy score of 80.28, just 0.74 points below the original model and 0.92 points above the non-polynomial simplified model. To provide a comprehensive view, the training curves are presented in left panel of Fig. 3, which also empirically analyzes several variants of the simplified model compared to the baseline. The full set of hyper-parameters can be found in the Appendix at Tab. 3.

5.2 Learning Polynomials

In this section, we empirically validate our theoretical claims concerning the expressiveness of S6 and self-attention layers from Thms. 1 and 2. Since our analysis of the expressiveness gap between those layers relies on their characterization via multivariate polynomials, we focus on learning such functions over synthetic data. To isolate factors other than expressiveness, we employ a control setup with small NNs, comprising up to four layers with narrow widths (2 or 4 channels) and an additional output linear projection head. For each architecture, we used the standard implementations: (i) self-attention with softmax and positional encoding, and

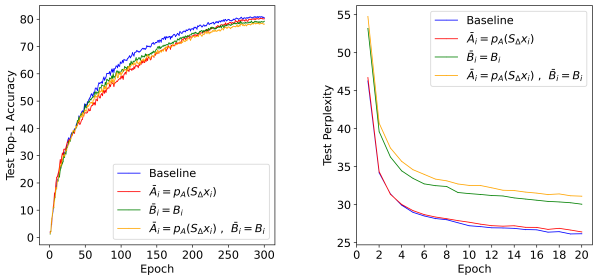

Figure 3: **Model justifications & ablations:** In the **left** panel, we present the top-1 accuracy score for image classification via the ImageNet-100 benchmark, while the **right** panel displays the perplexity score for language modeling using the WikiText-103. The y-axis represents the model's score across different epochs. In both figures, the blue curve represents the baseline, the yellow curve corresponds to Eq.9, the green curve illustrates Eq.10, and the red curve depicts the polynomial variant using standard discretization.

(ii) the original S6 architecture (Gu and Dao, 2023) with and without PE. The experiments examine the clean implementation of these components, without additional elements such as Conv1D, activations, or FFNs. We conduct experiments on two tasks: classification and regression.

Classification Our dataset consists of binary random sequences of length $L = 20$. The labels are uniformly distributed between 0 and L, determined using the "Count in Row" function (Ali et al., 2025), defined as follows:

Definition 1. *The count in row problem: Given a binary sequence $x_1, x_2, \dots x_L \in \{0,1\}^L$ such that $x_i \in \{0,1\}$, the "count in row" function f is defined to produce an output sequence $y_1, y_2, \dots, y_L$, where each $y_i = f(x_1, .., x_i)$ is determined based on the contiguous subsequence of 1s to which x_i belongs. Formally:*

$$y_i = \max_{0 \le j \le i} \left(\{ i - j + 1 \mid \prod_{k=j}^{i} [x_k > 0] = 1 \} \cup \{0\} \right) \tag{21}$$

where $[x_k > 0]$ is the Iverson bracket, equaling 1 if $x_k > 0$ and 0 otherwise.

The top part in Tab. 2 presents the results. Remarkably, even a single layer of selective SSMs, both with and without PE, outperforms attention models with double the number of layers and channels, all while utilizing significantly fewer parameters, as suggested by Thm 1.

Regression We synthetically construct the dataset $S = \{(x_i, y_i)\}_{i=1}^m$ by first randomly selecting a polynomial denoted by P. For each example in the dataset, we then generate x values uniformly at random and compute the corresponding labels using this P.

$$c_i \sim \mathbb{U}([-2, 2]),\ p_{i,j} \sim \mathbb{U}([L]),\ x_j \sim \mathbb{U}([0.1, 2]) \tag{22}$$

Table 1: Ablations of our simplified S6 variants, with vision tasks on the left and NLP on the right. 'S' for simplified variants. Results for Transformer models are provided as a reference point.

Model	Top-1	# Parameters
ViM (baseline)	81.02	6.2M
ViM (S., $\bar{B}_i = B_i$)	79.36	6.2M
ViM (S., $\bar{A}_i = p_A(S_\Delta(x_i))$)	80.28	6.2M
ViM (S., Eq. 9)	78.62	6.2M
Transformer (DeiT)	78.21	6.2M

Model	PPL	# Parameters
Mamba (baseline)	25.73	30.1M
Mamba (S., $\bar{B}_i = B_i$)	29.49	30.1M
Mamba (S., $\bar{A}_i = p_A(S_\Delta(x_i))$)	26.42	30.1M
Mamba (S., Eq. 9)	31.12	30.1M
Transformer	28.31	31.4M

Table 2: **Learning multivariate polynomials over synthetic data.** Classification results are presented on the left, and regression results on the right. Best results for each model depth are in bold.

Classification			
Model	# Layers	Accuracy	# Parameters
S6 w/ PE ($D = 2$)	1	83.1	35
S6 w/o PE ($D = 2$)	1	**84.8**	35
S6 w/ PE ($D = 2$)	2	93.4	63
S6 w/o PE ($D = 2$)	2	**97.1**	63
Self-Attention($D = 2$)	1	32.9	29
Self-Attention($D = 2$)	2	41.1	51
Self-Attention($D = 2$)	4	44.8	95
Self-Attention($D = 4$)	1	36.2	176
Self-Attention($D = 4$)	2	44.2	244
Self-Attention($D = 4$)	4	55.8	380

Regression			
Model	D	MSE	# Parameters
S6 w/ PE	4	12.67	101
S6 w/o PE	4	**11.81**	101
S6 w/ PE	6	12.45	157
S6 w/o PE	6	**11.04**	157
S6 w/ PE	8	12.17	377
S6 w/o PE	8	**9.057**	377
Self-Attention	4	19.22	81
Self-Attention	6	19.10	157
Self-Attention	8	19.048	257

$$y = P(x) = \sum_{i=1}^{3} c_i \Pi_{j=1}^{L} {x_j}^{p_{i,j}} \tag{23}$$

Our models consist of a single layer with either 4 or 8 channels, processing sequences of length $L = 5$. As demonstrated in the bottom part of Tab. 2, both S6 variants, with and without PE, significantly outperform traditional self-attention layers across both model sizes. For example, while a self-attention model with 8 channels obtains an MSE score of 19.05, all S6 variants achieve an MSE below 12.67. These experiments demonstrate that at least in controlled environments with small models, S6 layers outperform traditional self-attention layers in approximating polynomials where the total multivariate degree exceeds the sequence length.

6 DISCUSSION

To understand the implications of our theory, we first explain why analyzing Softmax-free attention is realistic. Then, we discuss the consequences for standard attention models.

Transformers Without Softmax The softmax function is primarily associated with optimization and stability, as it normalizes attention scores to the [0, 1] range and prevents numerical instabilities. However, transformer variants without softmax have proven effective in several domains, including reducing latency (Hua et al., 2022; Lu et al., 2021; Ramapuram et al., 2024) and in applications such as vision (Wortsman et al., 2023), NLP (Ma et al., 2022), and other areas (Zimerman et al., 2023). Additionally, these models have recently become even more practical, as researchers have successfully scaled linear attention far beyond 7B parameters (Li et al., 2025a), enabling LLMs to extend their context window to 4 million tokens while matching the performance of GPT-4o and Claude-3.5 Sonnet. Several additional linear attention-based LLMs were presented in (Shen et al., 2024; Qin et al., 2023; Sun et al., 2023). Since these models achieve near-SoTA, focusing on a softmax-free attention model is well justified.

Transformers With Softmax While the softmax function can theoretically be expressed as an infinite-degree polynomial, we provide careful considerations. The softmax function involves both exponentiation and proportional normalization. The former can be well approximated using low-degree polynomials (Zhang et al., 2024), while the latter primarily serves to normalize the scores. We refine our assumption by analyzing transformers that apply exponentiation to each attention score, assuming this can be approximated by a polynomial of degree P. The resulting model expresses higher-degree polynomials within each layer, but it remains based on pairwise interactions via Key, Query, and Values, leading to an maximal polynomial degree of $3P+1$, independent of the sequence length L. This supports the validity of our argument in more common regimes.

Interpretation and Intuition Our characterization of S6 layers through the lens of polynomials offers a novel perspective on the semantic capabilities of Mamba. Specifically, we extend the concept of polynomial degree to quantify the

number of tokens involved in each interaction within a layer of an model. For instance, low-degree polynomials correspond to interactions involving only a few tokens, while high-degree polynomials represent dependencies spanning many tokens. This analysis highlights the unique strength of S6 layers in modeling continuous, multi-token interactions, such as counting and recurrent operations. In contrast, transformers, are naturally biased toward sparser and more fragmented representations such as induction heads.

This perspective can also shed light on the remarkable performance of hybrid models that combine modern RNNs and attention by leveraging their complementary strengths (Lieber et al., 2024; De et al., 2024). While a formal characterization of their trade-offs is yet to be established, our analysis suggests that S6 and attention capture fundamentally distinct types of interactions, characterized by the number of tokens involved in each interaction.

Extension to multi-dimensional inputs While our analysis is presented for a single channel, it extends directly to inputs $X \in \mathbb{R}^{D\times L}$. In Mamba, the multiplicative interactions that give rise to high-degree polynomials occur only along the sequence (time) axis, and not across channels. Although the parameters at each time step depend on the full input vector X_{*t}, this dependence is pointwise in time and does not introduce multiplicative interactions across channels. Consequently, the model realizes multivariate polynomials whose degree is governed by sequential compositions along the sequence length L.

7 CONCLUSIONS

This study explores the expressivity of Mamba models. By reducing the S6 layer to a polynomial form and composing an associated theory, we have established a novel connection between S6 layers and high-degree multivariate polynomials. This connection enables us to identify the expressivity gap between S6 layers and attention mechanisms comprehensively. We show that although the S6 layer has better theoretical expressivity than linear attention for long sequences, this does not negatively impact generalization. We provide a length-agnostic generalization bound to support this result, allowing us to conclude that the S6 layer has superior theoretical properties compared to linear attention for long-range tasks.

Finally, an interesting direction for future work is to extend our analysis to additional modern sequence models, such as Gated DeltaNet (Yang et al., 2024) and other gated or hybrid architectures. It would also be valuable to further investigate how faithfully the polynomial framework reflects the practical behavior of these models, and to what extent it captures their effective inductive biases in real-world settings.

8 LIMITATIONS

In this paper, we provide a new perspective on the expressivity gap between S6 layers and self-attention, the core layers of Mamba models and transformers. While our analysis leverages multivariate polynomial degrees as measures of expressiveness and offers insightful results, it does not formally connect these measures to widely-used expressivity metrics in the literature, such as Rademacher complexity or VC dimension. Establishing such connections remains an open challenge.

Additionally, our work focuses on a simplified architecture, omitting several components of the original models. While we empirically justify these simplifications and highlight the opportunity to use simpler models by identifying the key components responsible for the performance gap, analyzing the full complexity of softmax-based Transformer and Mamba architectures is an important direction for future research.

Finally, although expressiveness is a critical property to explore, LLMs with billions of parameters involve additional factors that influence their capacity and performance. These include optimization challenges, gradient behavior, implicit biases, and training stability. Addressing these aspects is beyond the scope of this study, which is centered on a theoretical characterization of expressiveness. We believe these topics represent promising avenues for future work.

References

Md Atik Ahamed and Qiang Cheng. Mambatab: A simple yet effective approach for handling tabular data. *arXiv preprint arXiv:2401.08867*, 2024.

Ameen Ali Ali, Itamar Zimerman, and Lior Wolf. The hidden attention of mamba models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1516–1534, 2025.

Zeyuan Allen-Zhu and Yuanzhi Li. Can sgd learn recurrent neural networks with provable generalization? *Advances in Neural Information Processing Systems*, 32, 2019.

Ethan Baron, Itamar Zimerman, and Lior Wolf. A 2-dimensional state space layer for spatial inductive bias. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=BGkqypmGvm.

Nicola Muca Cirone, Antonio Orvieto, Benjamin Walker, Cristopher Salvi, and Terry Lyons. Theoretical foundations of deep selective state-space models. *arXiv preprint arXiv:2402.19047*, 2024.

Edo Cohen-Karlik, Avichai Ben David, Nadav Cohen, and Amir Globerson. On the implicit bias of gradient descent for temporal extrapolation. In *International Conference on Artificial Intelligence and Statistics*, pages 10966–10981. PMLR, 2022a.

Edo Cohen-Karlik, Itamar Menuhin-Gruman, Raja Giryes, Nadav Cohen, and Amir Globerson. Learning low dimensional state spaces with overparameterized recurrent neural nets. *arXiv preprint arXiv:2210.14064*, 2022b.

Soham De, Samuel L Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.

Melikasadat Emami, Mojtaba Sahraee-Ardakan, Parthe Pandit, Sundeep Rangan, and Alyson K Fletcher. Implicit bias of linear rnns. In *International Conference on Machine Learning*, pages 2982–2992. PMLR, 2021.

Tomer Galanti, Mengjia Xu, Liane Galanti, and Tomaso Poggio. Norm-based generalization bounds for sparse neural networks. *Advances in Neural Information Processing Systems*, 36, 2024.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Conference On Learning Theory*, pages 297–299. PMLR, 2018.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021a.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34:572–585, 2021b.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. *Advances in Neural Information Processing Systems*, 35: 22982–22994, 2022a.

Ankit Gupta, Harsh Mehta, and Jonathan Berant. Simplifying and understanding state space models with diagonal linear rnns. *arXiv preprint arXiv:2212.00768*, 2022b.

Moritz Hardt, Tengyu Ma, and Benjamin Recht. Gradient descent learns linear dynamical systems. *Journal of Machine Learning Research*, 19(29):1–44, 2018.

Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer quality in linear time. In *International Conference on Machine Learning*, pages 9099–9117. PMLR, 2022.

Samy Jelassi, David Brandfonbrener, Sham M Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. *arXiv preprint arXiv:2402.01032*, 2024.

Michel Ledoux and Michel Talagrand. *Probability in Banach spaces*. Springer Berlin, Heidelberg, 1991.

Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025a.

Kunchang Li, Xinhao Li, Yi Wang, Yinan He, Yali Wang, Limin Wang, and Yu Qiao. Videomamba: State space model for efficient video understanding. In *European Conference on Computer Vision*, pages 237–255. Springer, 2025b.

Shufan Li, Harkanwar Singh, and Aditya Grover. Mamband: Selective state space modeling for multi-dimensional data. *arXiv preprint arXiv:2402.05892*, 2024.

Dingkang Liang, Xin Zhou, Xinyu Wang, Xingkui Zhu, Wei Xu, Zhikang Zou, Xiaoqing Ye, and Xiang Bai. Pointmamba: A simple state space model for point cloud analysis. *arXiv preprint arXiv:2402.10739*, 2024.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

Fusheng Liu and Qianxiao Li. From generalization analysis to optimization designs for state space models. *arXiv preprint arXiv:2405.02670*, 2024.

Jiarun Liu, Hao Yang, Hong-Yu Zhou, Yan Xi, Lequan Yu, Yizhou Yu, Yong Liang, Guangming Shi, Shaoting Zhang, Hairong Zheng, et al. Swin-umamba: Mamba-based unet with imagenet-based pretraining. *arXiv preprint arXiv:2402.03302*, 2024a.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model. *arXiv preprint arXiv:2401.10166*, 2024b.

Jiachen Lu, Jinghan Yao, Junge Zhang, Xiatian Zhu, Hang Xu, Weiguo Gao, Chunjing Xu, Tao Xiang, and Li Zhang. Soft: Softmax-free transformer with linear complexity. *Advances in Neural Information Processing Systems*, 34: 21297–21309, 2021.

Qi Lv, Xiang Deng, Gongwei Chen, Michael Yu Wang, and Liqiang Nie. Decision mamba: A multi-grained state space model with self-evolution regularization for offline rl. *arXiv preprint arXiv:2406.05427*, 2024.

Xuezhe Ma, Chunting Zhou, Xiang Kong, Junxian He, Liangke Gui, Graham Neubig, Jonathan May, and Luke Zettlemoyer. Mega: moving average equipped gated attention. *arXiv preprint arXiv:2209.10655*, 2022.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. *arXiv preprint arXiv:2404.08819*, 2024.

Zhen Qin, Dong Li, Weigao Sun, Weixuan Sun, Xuyang Shen, Xiaodong Han, Yunshen Wei, Baohong Lv, Xiao Luo, Yu Qiao, et al. Transnormerllm: A faster and better large language model with improved transnormer. 2023.

Jason Ramapuram, Federico Danieli, Eeshan Dhekane, Floris Weers, Dan Busbridge, Pierre Ablin, Tatiana Likhomanenko, Jagrit Digani, Zijin Gu, Amitis Shidani, et al. Theory, analysis, and best practices for sigmoid self-attention. *arXiv preprint arXiv:2409.04431*, 2024.

Yash Sarrof, Yana Veitsman, and Michael Hahn. The expressive capacity of state space models: A formal language perspective. *arXiv preprint arXiv:2405.17394*, 2024.

Xuyang Shen, Dong Li, Ruitao Leng, Zhen Qin, Weigao Sun, and Yiran Zhong. Scaling laws for linear complexity language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16377–16426, 2024.

Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2022.

Gábor Lugosi Stéphane Boucheron, Olivier Bousquet. Theory of classification: a survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 3 2010. URL `http://eudml.org/doc/104340`.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, et al. An empirical study of mamba-based language models. *arXiv preprint arXiv:2406.07887*, 2024.

Chloe Wang, Oleksii Tsepa, Jun Ma, and Bo Wang. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *arXiv preprint arXiv:2402.00789*, 2024.

Mitchell Wortsman, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. Replacing softmax with relu in vision transformers. *arXiv preprint arXiv:2309.08586*, 2023.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024.

Jiawen Zhang, Jian Liu, Xinpeng Yang, Yinghao Wang, Kejia Chen, Xiaoyang Hou, Kui Ren, and Xiaohu Yang. Secure transformer inference made non-interactive. *Cryptology ePrint Archive*, 2024.

Itamar Zimerman and Lior Wolf. On the long range abilities of transformers. *arXiv preprint arXiv:2311.16620*, 2023.

Itamar Zimerman, Ameen Ali Ali, and Lior Wolf. Explaining modern gated-linear rnns via a unified implicit attention formulation. In *The Thirteenth International Conference on Learning Representations*.

Itamar Zimerman, Moran Baruch, Nir Drucker, Gilad Ezov, Omri Soceanu, and Lior Wolf. Converting transformers to polynomial form for secure inference over homomorphic encryption. *arXiv preprint arXiv:2311.08610*, 2023.

Jingwei Zuo, Maksim Velikanov, Dhia Eddine Rhaiem, Ilyas Chahed, Younes Belkada, Guillaume Kunsch, and Hakim Hacid. Falcon mamba: The first competitive attention-free 7b language model. *arXiv preprint arXiv:2410.05355*, 2024.

Checklist

1. For all models and algorithms presented, check if you include:

(a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes

(b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Not Applicable

(c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. Yes

2. For any theoretical claim, check if you include:

(a) Statements of the full set of assumptions of all theoretical results. Yes

(b) Complete proofs of all theoretical results. Yes

(c) Clear explanations of any assumptions. Yes

3. For all figures and tables that present empirical results, check if you include:

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). Yes

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets. Yes

(b) The license information of the assets, if applicable. Not Applicable

(c) New assets either in the supplemental material or as a URL, if applicable. Not Applicable

(d) Information about consent from data providers/curators. Not Applicable

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. Not Applicable

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable

A Experimental Setup

All training experiments were performed on public datasets using a single A100 40GB GPU for a maximum of two days. All experiments were conducted using PyTorch, and results are averaged over three seeds. All hyperparameters are detailed in Tab. 4 and Tab. 3.

Table 3: Hyperparameters for image-classification via Vision Mamba variants

Parameter	**Value**
Model-width	192
Number of layers	24
Number of patches	196
Batch-size	512
Optimizer	AdamW
Momentum	$\beta_1, \beta_2 = 0.9, 0.999$
Base learning rate	$5e-4$
Weight decay	0.1
Dropout	0
Training epochs	300
Learning rate schedule	cosine decay
Warmup epochs	5
Warmup schedule	linear
Degree of Taylor approx. (Eq. 9)	3

Table 4: Hyperparameters for language modeling via Mamba-based LMs

Parameter	**Value**
Model-width	386
Number of layers	12
Context-length (training)	1024
Batch-size	32
Optimizer	AdamW
Momentum	$\beta_1, \beta_2 = 0.9, 0.999$
Base learning rate	$1.5e-3$
Weight decay	0.01
Dropout	0
Training epochs	20
Learning rate schedule	cosine decay
Warmup epochs	1
Warmup schedule	linear
Degree of Taylor approx. (Eq. 9)	3

B Proofs

This section details our proofs.

B.1 Expressivity

Lemma 1. *There exists a function $f : \mathbb{R}^L \to \mathbb{R}$ that can be realized by one channel of S6 such that a single attention head would require at least $O(\log(L))$ layers to express this function.*

Proof of Lemma 1. The proof begins by characterizing the functions that an S6 layer can implement as multivariate polynomials with a maximal degree that scales linearly with the sequence length L. It then demonstrates that this property does not hold for self-attention layers.

Single S6 Layer as Multivariate Polynomials Let f be a function implemented by a S6 layer with the parameters A, S_Δ, S_B and S_C.

Recall that we deal with the following polynomial variant of S6, which is defined as follows:

$$C_i = (S_C X_i)^T, \quad \bar{B}_i = S_B X_i, \quad \bar{A}_i = P_A(S_\Delta X_i) \tag{24}$$

$$H_i = \bar{A}_t H_{i-1} + \bar{B}_i X_i, \quad Y_k = C_i X_i \tag{25}$$

Since we are interested in identifying the minimal polynomial degree required to characterize S6 models, we can assume that P_A is linear, namely $\bar{A}_i = S_\Delta X_i + A$. By plugging the zero matrix into A we get:

Now we can write Eq. 24 by:

$$H_i = S_\Delta X_i H_{i-1} + \bar{B}_i X_i, \quad Y_i = C_i X_i \tag{26}$$

Now, assuming S_Δ, S_B and S_C are sparse matrices such that they have zeros at all elements except a single column, namely, the time-variant matrices are controlled only by the first channel of the sequence X. Hence, this channel can be defined by:

$$h_i = S_\Delta x_i h_{i-1} + \bar{B}_i x_i, \quad y_i = C_i x_i \tag{27}$$

This exact recurrent rule is discussed in Eq. 9 (Single S6 layer as multivariate polynomials), and it can be represented as a multivariate polynomial with a maximum degree of $L + 2$. Hence, we can deduce that one element in the output of an S6 layer has a minimal maximum degree of $L + 2$ when processing sequences of length L.

Single Self-Attention Layer as Multivariate Polynomials Given an input matrix X, the self-attention mechanism without softmax operates as follows. The input is projected into query Q, key K, and value V matrices using parameter matrices W_Q, W_K, and W_V, respectively:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V.$$

The attention scores A are then computed as:

$$A = QK^T = (XW_Q)(XW_K)^T.$$

The output matrix Y is calculated by:

$$Y = AV = (XW_Q W_K^T X^T)(XW_V).$$

This formulation leads to the conclusion that each element in Y can be expressed as a multivariate polynomial with a maximum degree of 3 in the elements of X, where the polynomial arises from trilinear interactions. Additionally, it is important to note that in the case of causal attention, the mechanism is more constrained, and the maximum degree does not exceed 3.

Now, as it is clear that each single attention layer can be represented by a multivariate polynomial with a maximum degree of 3, we can generalize our characterization for N-stacked self-attention layers. Recall that the composition of multivariate polynomials is also a multivariate polynomial. Moreover, the maximum degree of the resulting polynomial is the product of the maximum degrees of the composed polynomials. Hence, we can argue that each element in the output of N-stacked self-attention layers can be represented by multivariate polynomials with a maximum degree $p \leq 3^N$. Therefore, it is clear that to represent a polynomial with a maximum degree of $L + 2$ by N-stacked self-attention layers, at least $N \in O(\log L)$ layers are required. □

Lemma 2. *Any function that can be expressed by a single attention head can also be expressed by a single channel of S6.*

Proof of Lemma 2. Let f be a function implemented via a single attention head, which has the parameters W_K, W_V, W_Q.

Recall that we deal with the following polynomial variant of S6, which is defined as follows:

$$C_i = (S_C X_i)^T, \quad \bar{B}_i = S_B X_i, \quad \bar{A}_i = P_A(S_\Delta X_i) \tag{28}$$

$$H_i = \bar{A}_t H_{i-1} + \bar{B}_i X_i, \quad Y_k = C_i X_i \tag{29}$$

For simplicity, assume that we are dealing with a 1-degree polynomial P_A such that $\bar{A}_i = S_\Delta X_i + A$.

By substituting the zero matrix for S_Δ and $A = 1$, and plugging $S_C = W_Q$ and $S_B = W_K$, we get:

$$C_i = (W_Q X_i)^T, \quad \bar{B}_i = W_K X_i, \quad \bar{A}_i = \bar{A}_{i-1} \tag{30}$$

By simply unrolling this equation:

$$H_i = \sum_{j=1}^{i} \left(\Pi_{k=j+1}^{i} \bar{A}_k\right) W_K X_j X_j = \sum_{j=1}^{i} W_K X_j X_j, \tag{31}$$

$$Y_i = {X_i}^T {W_Q}^T \sum_{j=1}^{i} W_K X_i X_j = \sum_{j=1}^{i} {X_i}^T {W_Q}^T W_K X_j X_j \tag{32}$$

By converting Eq. 32 into a matrix form:

$$Y = \tilde{\alpha} X \tag{33}$$

where $\tilde{\alpha}$ is defined by:

$$\begin{bmatrix} {X_1}^T {W_Q}^T W_K X_1 & 0 & \cdots & 0 \\ {X_2}^T {W_Q}^T W_K X_1 & {X_2}^T {W_Q}^T W_K X_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ {X_L}^T {W_Q}^T W_K X_1 & {X_L}^T {W_Q}^T W_K X_2 & \cdots & {X_L}^T {W_Q}^T W_K X_L \end{bmatrix}$$

Which is the exact formulation of causal self-attention (without softmax) using attention matrices denoted by $\tilde{\alpha}$. Furthermore, to incorporate the value matrix W_V an additional linear layer should be applied after step S6. These layers are indeed present in the Mamba block. □

Theorem 2. *Given an input scalar sequence $x \in \mathbb{R}^L$, a model with four stacked Mamba layers, a sufficiently large number of channels, learnable PE, and a linear encoder at the first layer can express any multivariate polynomial of x.*

Proof of Theorem 2. We start with the following definition of our model. We begin by describing the hidden Mamba layers, followed by the input and output layers:

Model Definition We define the model with d channels, ignoring its activations. We denoted the i-th Mamba block by (see Eq. 2):

$$U^{i+1} = \text{Linear}_1{}^i(\text{S6}^i(\text{Conv1D}^i(\text{Linear}_2{}^i(U^i)) \otimes \text{Linear}_3{}^i(U^i)) \tag{34}$$

where sequences and sub-layers associated with the i-th layer are denoted by the super-index i. Thus, the entire computation of single layer can be described by

$$U^{i+1} = \text{Mamba}^i(U_i) \tag{35}$$

The output linear layer projects the last token (which include d channels) into a single output, parameterized by a matrix $W_{\text{out}} \in \mathbb{R}^{d \times 1}$. The input layer includes the learnable positional encoding, represented by a matrix $PE \in \mathbb{R}^{L \times d}$, and an encoding layer parameterized by $\text{Encoding}(x) = W_{\text{in}} x + b + PE$, where $W_{\text{in}}, b \in \mathbb{R}^{1 \times d}$.

Proof by Construction Let $P(x)$ be a multivariate polynomial with coefficients $c_1, \cdots, c_T$ and variables $x = (x_1, x_2, \ldots, x_n)$. Specifically, $P(x)$ can be expressed as:

$$P(x) = \sum_{i=1}^{T} c_i \cdot P_i(x), \quad \forall i : P_i(x) = \Pi_{j=1}^{L} \alpha_{i,j} {x_j}^{p_{i,j}} \tag{36}$$

We assign the values of $c_1, \cdots, c_T$ to W_{out} such that:

$$W_{\text{out}}[i,1] = \begin{cases} c_i & \text{if } i \leq T, \\ 0 & \text{otherwise.} \end{cases} \tag{37}$$

It remains to show that for any i, $P_i(x)$ can be expressed by the last (4th) Mamba block. From Lemma 3, it is evident that for any j, the univariate polynomial $s_j = \alpha_{i,j} x_j^{p_{i,j}}$ can be represented by the 3rd Mamba block. Thus, in the first linear layer of the 3rd block (Linear$_1^3$), these univariate polynomials can be merged, as follows:

Define the following two sequences with Lemma 3 and Linear$_1^3$:

$$S' = (0, \cdots, 0, 1, s_1, s_2, \cdots, s_L, 1),$$

$$S'' = (0, \cdots, 0, 0, s_1, s_2, \cdots, s_L, 1),$$

which are injected into the last SSM (S6^4).

When applying the same SSM with system matrices equal to 1 ($A_i = B_i = C_i = 1$), and subtracting them, placing the result in the i-th channel (which can be easily implemented by the last linear layer in the final Mamba block), yield:

$$\forall i : U_i^4 = \prod_{j=1}^{L} s_j = P_i(x), \quad \rightarrow W_{out} U^4 = P(x) \tag{38}$$

as required. □

Lemma 3. *Given an input scalar sequence $x \in \mathbb{R}^L$, for any j, a model M with 3-stacked Mamba layers, a sufficiently large number of channels, learnable PE, and a linear encoder in the first layer can express any monomial of a univariate polynomial in x_j. Specifically, for any constants $c \in \mathbb{R}$ and $P \in \mathbb{R}$, there exists a configuration of M such that the output of the k-th channel $M(x)_k = c \cdot x_j^P$ for $k > P + j$.*

Proof of Lemma 3. Given an input sequence $x = (x_1, x_2, \ldots, x_L)$ with d channels, we need to show that a model M composed of three stacked Mamba layers can express any univariate polynomial x_j^P for a particular j, where P are constant.

For simplicity, we assume that $P << L$. However, this assumption can be addressed by extending the sequence length using the ZeroPad component.

Step 1: Position Selection (First Mamba Block)

The first block's role is to isolate the desired position j in the input sequence (x_j) at channel s. We achieve this as follows:

- **Positional Encoding and Gating:** We configure the learnable positional encoding at channel s ($PE_{*,s}$) such that the j-th position is highlighted, and the others positions are masked. Specifically, set the PE vector for the j-th position to act as an indicator function:

  $$PE[i,s] = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

  Please note that $s > \frac{d}{2}$, and the first half of the channels in the PE are set to zeros to ensure a clear separation between the input values X and the positional encoding. Additionally, we set $b = 0$, and configure the first half of the entries in W_{in} to 1, while setting the remaining entries to zero. This configuration effectively duplicates the input sequence, allowing for several manipulations without affecting the original signal.

- **Linear Layers Configuration:** The linear layers Linear$_2^1$ and Linear$_3^1$ are configured to be identity mappings, i.e., they do not alter the input sequence.

- **S6 Module Configuration:** Set the S6 parameters $\bar{A} = 0$ and $\bar{B} = \bar{C} = 1$, effectively making it an identity operation. This setup ensures that the output of this block isolates x_j while setting all other positions to zero:

  $$U^2 = (0, \ldots, 0, x_j, 0, \ldots, 0)$$

Step 2: Duplication of x_j (Second Mamba Block)

The second block is responsible for duplicating the selected element x_j to match the desired power P. Here's how:

- **Identity Mapping:** We set Linear_2^2 and Linear_3^2 to identity mappings. Additionally, the gate branch on the channels operating on the isolated x_j is set to 1 at positions smaller than h and 0 for the rest, also functioning as an identity mapping at positions $i \leq h$, and masking positions for indexes $i > j$.
- **S6 :** The S6 module is configured to allow duplication. Specifically, by setting $\bar{A} = 1$ and $\bar{B}, \bar{C} = 1$, the S6 can output a sequence with multiple copies of x_j:

$$U^3 = (0, \ldots, 0, x_j, x_j, \ldots, x_j, 0, \ldots, 0)$$

Here, x_j appears $P-2$ times, and the last x_j positioned at index $i = h$.

Step 3: Aggregate Multiplications to Powers (Third Mamba Block)

The third Mamba block is designed to aggregate the duplicated elements x_j into the form x_j^P, utilizing the multiplicative capabilities of the S6 module and the subtraction mechanism in the final linear layer.

- **S6 Module Configuration:** In this block, the S6 module is configured to perform the necessary multiplications that aggregate the duplicated values of x_j. This is achieved by setting the system input and output matrices to 1. Hence, the output of the SSM when applied on U_2 at position h will result at:

$$\sum_{j=1}^{P-2} c_j {x_j}^{j+2}$$

Thus, we construct an additional sequence, similar to U_2, denoted by U_2', by introducing an additional zero at the initial occurrence of x_j. We then subtract the outputs from these two identical SSM channels at the final linear layer of the block. This subtraction yields a telescoping series:

$$\text{SSM}(U_2) - \text{SSM}(U_2') = \tag{39}$$

$$\sum_{j=1}^{L} c_j \Big(\Pi_{k=j+1}^{t-1} x_k\Big) {x_t}^2 {x_j}^2 - \sum_{j=2}^{L} c_j \Big(\Pi_{k=j+1}^{t-1} x_k\Big) {x_t}^2 {x_j}^2 = \sum_{j=1}^{P-2} c_j {x_j}^{j+2} - \sum_{j=2}^{P-2} c_j {x_j}^{j+2} = {x_j}^P \tag{40}$$

yields

$$U^4 = \big(0, \ldots, 0, x_j^P, 0, \ldots, 0\big)$$

This construction shows that a model with three stacked Mamba layers and sufficient channels can indeed express any univariate polynomial x_j^P, thereby proving Lemma 3.

□

B.2 Generalization

Let P be a distribution over $\mathbb{R}^{D \times L} \times [C]$. Let $S = \{(X_{(j)}, y_{(j)})\}_{j=1}^m$ be a dataset of i.i.d. samples selected from P. Our generalization bound is based on a uniform-convergence generalization bound provided in (Galanti et al., 2024). The following lemma bounds the gap between the test error and the empirical margin error, represented as $\text{err}_S^\gamma(f_w) = \frac{1}{m}\sum_{j=1}^m \mathbb{I}[\max_{c \neq c'}(f_w^c(X_{(j)})) + \gamma \geq f_w^{c'}(X_{(j)})]$.

Lemma 4. *Let P be a distribution over $\mathbb{R}^{\mathcal{D}} \times [C]$ and $\mathcal{F} \subset \{f' : \mathcal{X} \to \mathbb{R}^C\}$. Let $S = \{(X_j, y_j)\}_{j=1}^m$ be a dataset of i.i.d. samples selected from P and $X = \{X_j\}_{j=1}^m$. Then, with probability at least $1-\delta$ over the selection of S, for any $f_w \in \mathcal{F}$, we have*

$$\text{err}_P(f_w) - \text{err}_S^\gamma(f_w) \leq \frac{2\sqrt{2}}{\gamma} \cdot \mathcal{R}_X(\mathcal{F}) + 3\sqrt{\frac{\log(2/\delta)}{2m}}. \tag{41}$$

Lemma 5. *Let σ_j be a l_j-Lipschitz, positive-homogeneous function and $l = \max_j l_j$. Let $\xi_i \sim U[\{\pm 1\}]$. Then for any class of vector-valued functions $\mathcal{F} \subset \{f \mid f : \mathbb{R}^d \to \mathbb{R}\}$ and any convex and monotonically increasing function $g : \mathbb{R} \to [0, \infty)$,*

$$\mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(\left|\sum_{j=1}^m \xi_j \cdot \sigma_j(f(x_j))\right|\right) \le 2\mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(l\left|\sum_{i=j}^m \xi_j \cdot f(x_j)\right|\right).$$

Proof. We notice that since $g(|z|) \le g(z) + g(-z)$,

$$\begin{aligned}
&\mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(\left|\sum_{j=1}^m \xi_j \cdot \sigma_j(f(x_j))\right|\right) \\
&\le \mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(\sum_{j=1}^m \xi_j \cdot \sigma_j(f(x_j))\right) \\
&+ \mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(-\sum_{j=1}^m \xi_j \cdot \sigma_j(f(x_j))\right) \\
&= 2\mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(\sum_{j=1}^m \xi_j \cdot \sigma_j(f(x_j))\right)
\end{aligned}$$

where the last equality follows from the symmetry in the distribution of the ξ_i random variables. By Equation 4.20 in (Ledoux and Talagrand, 1991) we have the following:

$$\begin{aligned}
&\mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(\sum_{j=1}^m \xi_j \cdot \sigma_j(f(x_j))\right) \\
&= \mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(l\sum_{j=1}^m \xi_j \cdot \frac{1}{l}\sigma_j(f(x_j))\right) \\
&\le \mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(l\sum_{j=1}^m \xi_j \cdot f(x_j)\right) \\
&\le \mathbb{E}_\xi \sup_{f\in\mathcal{F}} g\left(l\left|\sum_{j=1}^m \xi_j \cdot f(x_j)\right|\right)
\end{aligned}$$

where the last equality follows from g being monotonically increasing. □

The model is denoted by:

$$\begin{aligned}
&B_i = S_B x_i, \quad C_i = S_C x_i, \quad \Delta_i = \sigma(S_\Delta x_i), \\
&\bar{A}_i = \exp(\Delta_i A), \quad \bar{B}_i = B_i
\end{aligned} \tag{42}$$

Where σ is a 1-Lipschitz activation function. We consider a classifier $f : \mathbb{R}^{D\times L} \to \mathbb{R}^{\mathcal{C}}$ defined as follows. We have parameters (A, S_B, S_C, S_Δ) associated with the layer. The norms are defined as follows:

$$\begin{aligned}
\rho_A(w) &= \|A\|_{\max} \\
\rho_B(w) &= \|S_B\|_{2,\infty} \\
\rho_C(w) &= \|S_C\|_F \\
\rho_\Delta(w) &= \|S_\Delta\|_2
\end{aligned} \tag{43}$$

We denote the product of these norms as:

$$\Gamma(w) = \rho_A(w) \cdot \rho_B(w) \cdot \rho_C(w) \cdot \rho_\Delta(w) \tag{44}$$

Given these definitions, the classifier f for a specific class $c \in [\mathcal{C}]$ is computed as:

$$f^c(X_{*1}, ..., X_{*L}) = \sum_{d=1}^D W_{c,d} \left(S_C X_{*L}\right)^T \sum_{i=1}^L \left(\prod_{k=i+1}^L \bar{A}_{dk}\right) S_B X_{*i} X_{di}$$

Here, $W \in \mathbb{R}^{\mathcal{C} \times D}$ represents a linear projection from the output to the number of classes, and $\mathcal{C}$ is the number of classes.

We denote the parameters of the classifier by

$$w = (A, S_B, S_C, S_\Delta, W)$$

and the function induced by a specific instance of w is denoted by f_w. The class of functions taking on different parameter instances w is denoted by $\mathcal{F}$. Denote

$$\rho = \{\rho_W, \rho_A, \rho_B, \rho_C, \rho_\Delta\}$$

and let:

$$\mathcal{F}_\rho = \{f_w \in \mathcal{F} : \Gamma(w) \leq \rho_A \rho_B \rho_C \rho_\Delta \rho_W =: \Gamma\} \tag{45}$$

The following theorem provides a bound on the Rademacher complexity of the class $\mathcal{F}_\rho$.

Theorem 4. *Let $\rho = \{\rho_W, \rho_A, \rho_B, \rho_C, \rho_\Delta\}$. Suppose we have m sample sequences $X = \{X_{(j)}\}_{j=1}^m$, where each $X_{(j)} = (X_{(j)_{*1}}, \ldots, X_{(j)_{*L}}) \in \mathbb{R}^{D \times L}$. Assume that $\forall j \in [m] : ||X_{(j)}||_{\max} \leq 1$. Additionally, suppose $\forall k \in [L], d \in [D] : ||A_{dk}||_{\max} < K < 1$. Then,*

$$\mathcal{R}_X(\mathcal{F}_\rho) \leq \frac{1}{m} D^2 \Gamma(1 + \sqrt{2\log(2L\mathcal{C}D^4N)}) \cdot \sqrt{\max_{t,k} \sum_{j=1}^m (X_{(j)_{tk}})^2} \frac{K}{(K-1)^2}$$

where the maximum is taken over $t \in [D]$, $k \in [L]$.

Proof. For aesthetic purposes, we define B as S_B and C as S_C. The Rademacher complexity of $\mathcal{F}_\rho$ is given by:

$$\begin{aligned}
m\mathcal{R}(\mathcal{F}_\rho) &= \mathbb{E}_\xi \left[\sup_w \sum_{j=1}^m \sum_{c=1}^{\mathcal{C}} \xi_{jc} f_w^c(X_{(j)}) \right] \\
&= \mathbb{E}_\xi \left[\sup_w \sum_{j=1}^m \sum_{c=1}^{\mathcal{C}} \xi_{jc} \sum_{d=1}^D W_{c,d} \left(S_C X_{(j)_{*L}}\right)^T \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot A_{d*}\right) \right) S_B X_{(j)_{*i}} X_{(j)_{di}} \right] \\
&= \mathbb{E}_\xi \left[\sup_w \sum_{j=1}^m \sum_{c=1}^{\mathcal{C}} \xi_{jc} \sum_{d=1}^D W_{c,d} \left(C X_{(j)_{*L}}\right)^T \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot A_{d*}\right) \right) B X_{(j)_{*i}} X_{(j)_{di}} \right]
\end{aligned}$$

Here, A_{d*} represents the d-th row of A, which has a size of N. We think of A_{d*} as a diagonal matrix of size $N \times N$, where its diagonal elements are the values in the dth row of A. Thus, $A_{d*} = \text{diag}(a_{d1}, ..., a_{dN})$. Hence,

$$\begin{aligned}
&\mathbb{E}_\xi \left[\sup_w \sum_{j=1}^m \sum_{c=1}^{\mathcal{C}} \xi_{jc} \sum_{d=1}^D W_{c,d} \left(C X_{(j)_{*L}}\right)^T \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot A_{d*}\right) \right) B X_{(j)_{*i}} X_{(j)_{di}} \right] \\
&= \mathbb{E}_\xi \left[\sup_w \sum_{j=1}^m \sum_{c=1}^{\mathcal{C}} \xi_{jc} \sum_{d=1}^D W_{c,d} \sum_{l=1}^N \left(C X_{(j)_{*L}}\right)_l^T \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) \right) B_{l*} X_{(j)_{*i}} X_{(j)_{di}} \right] \\
&= \mathbb{E}_\xi \left[\sup_w \sum_{j=1}^m \sum_{c=1}^{\mathcal{C}} \xi_{jc} \sum_{d=1}^D W_{c,d} \sum_{l=1}^N \left(C_{l*} X_{(j)_{*L}}\right) \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) \right) B_{l*} X_{(j)_{*i}} X_{(j)_{di}} \right] \\
&= \mathbb{E}_\xi \left[\sup_w \sum_{c=1}^{\mathcal{C}} \sum_{d=1}^D W_{c,d} \sum_{j=1}^m \xi_{jc} \sum_{l=1}^N \left(C_{l*} X_{(j)_{*L}}\right) \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) \right) B_{l*} X_{(j)_{*i}} X_{(j)_{di}} \right]
\end{aligned}$$

This follows from expressing the model in a more explicit form. Next,

$$\begin{aligned}
&\mathbb{E}_\xi \left[\sup_w \sum_{c=1}^{\mathcal{C}} \sum_{d=1}^D W_{c,d} \sum_{j=1}^m \xi_{jc} \sum_{l=1}^N \left(C_{l*} X_{(j)_{*L}}\right) \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) \right) B_{l*} X_{(j)_{*i}} X_{(j)_{di}} \right] \\
&= \sqrt{D} \rho_W \mathbb{E}_\xi \left[\sup_{w,c,d} \left| \sum_{j=1}^m \xi_{jc} \sum_{l=1}^N \left(C_{l*} X_{(j)_{*L}}\right) \sum_{i=1}^L \left(\prod_{k=i+1}^L \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) \right) B_{l*} X_{(j)_{*i}} X_{(j)_{di}} \right| \right]
\end{aligned}$$

where the inequality follows from moving the norm of W to W_c for maximizing the inner term and applying the Cauchy-Schwartz inequality. Next,

$$
\begin{aligned}
&\sqrt{D}\rho_W \mathbb{E}_\xi \left[\sup_{w,c,d} |\sum_{j=1}^{m} \xi_{jc} \sum_{l=1}^{N} \left(C_{l*} X_{(j)_{*L}}\right) \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right)\right) B_{l*} X_{(j)_{*i}} X_{(j)_{di}} | \right] \\
&= \sqrt{D}\rho_W \mathbb{E}_\xi \left[\sup_{w,c,d} |\sum_{j=1}^{m} \xi_{jc} \sum_{l=1}^{N} \left(\sum_{s=1}^{D} C_{ls} X_{(j)_{sL}}\right) \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right)\right) \left(\sum_{s'=1}^{D} B_{ls'} X_{(j)_{s'i}}\right) X_{(j)_{di}} | \right] \\
&= \sqrt{D}\rho_W \mathbb{E}_\xi \left[\sup_{w,c,d} |\sum_{l=1}^{N} \sum_{s=1}^{D} C_{ls} \sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right)\right) \left(\sum_{s'=1}^{D} B_{ls'} X_{(j)_{s'i}}\right) X_{(j)_{di}} | \right] \\
&\leq D\rho_W \rho_C \mathbb{E}_\xi \left[\sup_{w,c,d,l,s} |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right)\right) \left(\sum_{s'=1}^{D} B_{ls'} X_{(j)_{s'i}}\right) X_{(j)_{di}} | \right] \\
&= D\rho_W \rho_C \mathbb{E}_\xi \left[\sup_{w,c,d,l,s} |\sum_{s'=1}^{D} B_{ls'} \sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right)\right) X_{(j)_{s'i}} X_{(j)_{di}} | \right]
\end{aligned}
$$

where the first inequality follows from moving the norm of C to C_l for maximizing the inner term and applying the Cauchy-Schwartz inequality.

$$
\begin{aligned}
&= D\rho_W \rho_C \mathbb{E}_\xi \left[\sup_{w,c,d,l,s} |\sum_{s'=1}^{D} B_{ls'} \sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right)\right) X_{(j)_{s'i}} X_{(j)_{di}} | \right] \\
&\leq D^{1.5}\rho_W \rho_C \rho_B \mathbb{E}_\xi \left[\sup_{w,c,d,l,s,s'} |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \sum_{i=1}^{L} \left(\prod_{k=i+1}^{L} \exp\left(\sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right)\right) X_{(j)_{s'i}} X_{(j)_{di}} | \right] \\
&= D^{1.5}\rho_W \rho_C \rho_B \mathbb{E}_\xi \left[\sup_{w,c,d,l,s,s'} |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \sum_{i=1}^{L} \exp\left(\sum_{k=i+1}^{L} \sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) X_{(j)_{s'i}} X_{(j)_{di}} | \right] \\
&\leq D^{1.5}\rho_W \rho_C \rho_B \sum_{i=1}^{L} \mathbb{E}_\xi \left[\sup_{w,c,d,l,s,s'} |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \cdot \exp\left(\sum_{k=i+1}^{L} \sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) X_{(j)_{s'i}} X_{(j)_{di}} | \right]
\end{aligned}
$$

where the second to last inequality follows from $\|x\|_2 \leq \sqrt{n}\|x\|_\infty$ for any $x \in \mathbb{R}^n$. Jensen's inequality gives the following inequality:

$$
\begin{aligned}
&\mathbb{E}_\xi \left[\sup_{w,c,d,l,s,s'} |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \cdot \exp\left(\sum_{k=i+1}^{L} \sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) X_{(j)_{s'i}} X_{(j)_{di}} | \right] \\
&\leq \frac{1}{\lambda_i} \log(\mathbb{E}_\xi \left[\sup_{w,c,d,l,s,s'} \exp(\lambda_i |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \cdot \exp\left(\sum_{k=i+1}^{L} \sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) X_{(j)_{s'i}} X_{(j)_{di}} |) \right]) \\
&\leq \frac{1}{\lambda_i} \log(\sum_{c,d,l,s,s'} \mathbb{E}_\xi \left[\sup_{w} \exp(\lambda_i |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \cdot \exp\left(\sum_{k=i+1}^{L} \sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) X_{(j)_{s'i}} X_{(j)_{di}} |) \right]) \\
&\leq \frac{1}{\lambda_i} \log(\mathcal{C} D^3 N \max_{c,d,l,s,s'} \mathbb{E}_\xi \left[\sup_{w} \exp(\lambda_i |\sum_{j=1}^{m} \xi_{jc} X_{(j)_{sL}} \cdot \exp\left(\sum_{k=i+1}^{L} \sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}\right) X_{(j)_{s'i}} X_{(j)_{di}} |) \right]) := \Theta
\end{aligned}
\tag{46}
$$

For fixed $\lambda_i > 0$. The second inequality follows from the fact that $\sup_x \sup_y f(x,y) = \sup_{x,y} f(x,y)$. We observe that the inner expectation sup depends only on w. Next we use Lemma 5 with $\sigma_{ij}(z) = \exp(z) X_{(j)_{sL}} X_{(j)_{s'i}} X_{(j)_{di}}$ on its domain and $g_i(X_{(j)}) = \sum_{k=i+1}^{L} \sigma(S_\Delta X_{(j)_{*k}}) \cdot a_{dl}$. The corresponding Lipschitz constants are $l_{ij} =$

$\max_{z\in dom(\sigma_{ij})}(\exp(z)X_{(j)_{sL}}X_{(j)_{s'i}}X_{(j)_{di}})$ and $l_i = \max(l_{ij})$. Therefore:

$$\begin{aligned}
&\mathbb{E}_\xi\left[\sup_w \exp(\lambda_i|\sum_{j=1}^{m}\xi_{jc}X_{(j)_{sL}}\cdot\ \exp\left(\sum_{k=i+1}^{L}\sigma(S_\Delta X_{(j)_{*k}})\cdot a_{dl}\right)X_{(j)_{s'i}}X_{(j)_{di}}|)\right]\\
&=\mathbb{E}_\xi\left[\sup_w \exp\left(\lambda_i\left|\sum_{j=1}^{m}\xi_{jc}(\sigma_{ij}(g_i(X^{(j)})))\right|\right)\right]\\
&\le 2\mathbb{E}_\xi\left[\sup_w \exp\left(\lambda_i l_i\left|\sum_{j=1}^{m}\xi_{jc}\sum_{k=i+1}^{L}\sigma(S_\Delta X_{(j)_{*k}})\cdot a_{dl}\right|\right)\right]\\
&\le 2\mathbb{E}_\xi\left[\sup_w \exp\left(\lambda_i\rho_A l_i\left|\sum_{j=1}^{m}\xi_{jc}\sum_{k=i+1}^{L}\sigma(S_\Delta X_{(j)_{*k}})\right|\right)\right]\\
&\le 2\mathbb{E}_\xi\left[\sup_w \exp\left(\lambda_i\rho_A(L-i) l_i\left|\sum_{j=1}^{m}\xi_{jc}\sigma(S_\Delta X_{(j)_{*k}})\right|\right)\right]\\
&\le 4\mathbb{E}_\xi\left[\sup_w \exp\left(\lambda_i\rho_A(L-i) l_i\left|\sum_{j=1}^{m}\xi_{jc}S_\Delta X_{(j)_{*k}}\right|\right)\right]
\end{aligned} \tag{47}$$

This follows from applying Lemma 5 with σ which has a Lipschitz constant of 1, as assumed. Hence:

$$\begin{aligned}
&4\mathbb{E}_\xi\left[\sup_w \exp\left(\lambda_i\rho_A(L-i) l_i\left|\sum_{j=1}^{m}\xi_{jc}S_\Delta X_{(j)_{*k}}\right|\right)\right]\\
&\le 4\mathbb{E}_\xi\left[\sup_k \exp\left(\lambda_i\rho_A\rho_\Delta(L-i) l_i||\sum_{j=1}^{m}\xi_{jc}X_{(j)_{*k}}||\right)\right]\\
&\le 4\mathbb{E}_\xi\left[\sup_{t,k} \exp\left(\lambda_i\sqrt{D}\rho_A\rho_\Delta(L-i) l_i\left|\sum_{j=1}^{m}\xi_{jc}X_{(j)_{tk}}\right|\right)\right]\\
&\le 4DL\max_{t,k}\mathbb{E}_\xi\left[\exp\left(\lambda_i\sqrt{D}\rho_A\rho_\Delta(L-i) l_i\left|\sum_{j=1}^{m}\xi_{jc}X_{(j)_{tk}}\right|\right)\right]
\end{aligned} \tag{48}$$

Denote:

$$M_i := \sqrt{D}\rho_A\rho_\Delta(L-i)l_i \tag{49}$$

As a next step, we would like to bound the above term using a function of the data that is not dependent on an expected value of noise labels ξ. For this purpose we apply a technique that was introduced in the proof of Theorem 1 in (Golowich et al., 2018). We apply this process separately for each $i \in [L]$. Let $i \in [L]$: We define a random variable Z:

$$\begin{aligned}
Z &= M_i\left|\sum_{j=1}^{m}\xi_{jc}X_{(j)_{tk}}\right|\\
z_j &= X_{(j)_{tk}} \rightarrow Z = M_i|\sum_{j=1}^{m}\xi_{jc}z_j|
\end{aligned}$$

The random variable Z depends on the random variables ξ_{jc}. Then, we have:

$$\begin{aligned}
&= \frac{1}{\lambda_i}\log\mathbb{E}_\xi\left[\exp(\lambda_i Z)\right]\\
&= \frac{1}{\lambda_i}\log\mathbb{E}_\xi\left[\exp(\lambda_i Z+\lambda_i\mathbb{E}(Z)-\lambda_i\mathbb{E}(Z))\right]\\
&= \frac{1}{\lambda_i}\log\mathbb{E}_\xi\left[\exp(\lambda_i Z-\lambda_i\mathbb{E}(Z))\right]+\mathbb{E}_\xi(Z)
\end{aligned}$$

By Jensen's inequality, we obtain a bound for $\mathbb{E}(|\sum_{j=1}^{m}\xi_{jc}z_j|)$:

$$\mathbb{E}_\xi(|\sum_{j=1}^{m}\xi_{jc}z_j|)=\mathbb{E}_\xi(\sqrt{|\sum_{j=1}^{m}\xi_{jc}z_j|^2})\leq\sqrt{\mathbb{E}_\xi(|\sum_{j=1}^{m}\xi_{jc}z_j|^2)}=$$
$$\sqrt{\mathbb{E}_\xi(|\sum_{j=1}^{m}\xi_{jc}z_j|^2)}=\sqrt{\mathbb{E}_\xi(|\sum_{j,j'=1}^{m}\xi_{jc}\xi_{j'c}z_jz_{j'}|)}=\sqrt{\sum_{j=1}^{m}|z_j|^2}$$

Namely $\mathbb{E}_\xi(Z)\leq M_i\sqrt{\sum_{j=1}^{m}|z_j|^2}$. Z is a deterministic function of the i.i.d. random variables ξ_{jc} and satisfies the following:

$$Z(\xi_{1c},...,\xi_{jc},...,\xi_{mc})-Z(\xi_{1c},...,-\xi_{jc},...,\xi_{mc})\leq 2|z_j|$$

This follows from the triangle inequality. This means that Z satisfies a bounded-difference condition, which, by the proof of Theorem 6.2 in (Stéphane Boucheron, 2010), implies that Z is sub-Gaussian, with variance factor:

$$v=\frac{1}{4}\sum_{j=1}^{m}(2M_i|z_j|)^2=M_i^2\sum_{j=1}^{m}|z_j|^2$$

It follows that:

$$\frac{1}{\lambda_i}\log\mathbb{E}_\xi\left[\exp(\lambda_iZ-\lambda_i\mathbb{E}_\xi(Z))\right]\leq$$
$$\frac{1}{\lambda_i}\frac{\lambda_i^2M_i^2\sum_{j=1}^{m}|z_j|^2}{2}=\frac{\lambda_iM_i^2\sum_{j=1}^{m}|z_j|^2}{2}$$

Therefore:

$$\begin{aligned}&\frac{1}{\lambda_i}\log\mathbb{E}_\xi\left[\exp(\lambda_iZ-\lambda_i\mathbb{E}_\xi(Z))\right]+\mathbb{E}_\xi(Z)\\&\leq\frac{\lambda_iM_i^2\sum_{j=1}^{m}|z_j|^2}{2}+M_i\sqrt{\sum_{j=1}^{m}|z_j|^2}\end{aligned}\tag{50}$$

Analyzing Lipschitz constants l_{ij}. Next, we analyze the Lipschitz constants l_{ij}. For $l_{ij}=\max_{z\in dom(\sigma_{ij})}(\exp(z)X_{(j)_{sL}}X_{(j)_{s'i}}X_{(j)_{di}})$ and $l_i=\max(l_{ij})$ is of the form $g_i(X_{(j)})=\sum_{k=i+1}^{L}\sigma(S_\Delta X_{(j)_{*k}})\cdot a_{dl}$ (see equation 48). Since

$$l_i=\max_j l_{ij}=\max_j\max_{z\in dom(\sigma_{ij})}(\exp(z)X_{sL}^{(j)}X_{s'i}^{(j)}X_{di}^{(j)})$$
$$\leq\max_j\max_{z\in dom(\sigma_{ij})}\exp\left(\sum_{k=i+1}^{L}\sigma(S_\Delta X_{(j)_{*k}})\cdot a_{dl}\right)\cdot 1<K^{L-i}$$

which is followed from our assumptions.

We get:

$$\sum_{i=1}^{L}l_i(L-i)\leq\sum_{i=1}^{L}(L-i)(K)^{L-i}=\frac{(L-1)K^{L+1}-LK^L+K}{(K-1)^2}$$

We conclude that:

$$\lim_{L\to\infty}\frac{(L-1)K^{L+1}-LK^L+K}{(K-1)^2}=\frac{K}{(K-1)^2}$$

Concluding the proof. By combining equation 46, equation 48 and equation 50, we have:

$$\begin{aligned}&\Theta\leq\frac{1}{\lambda_i}\log\left(4L\mathcal{C}D^4N\right)+\max_{c,d,l,s,s',t,k}\frac{1}{\lambda_i}\cdot\\&\log\left(\mathbb{E}_\xi\left[\exp\left(\lambda_i\sqrt{D}\rho_A\rho_\Delta(L-i)l_i\left|\sum_{j=1}^{m}\xi_{jc}X_{(j)_{tk}}\right|\right)\right]\right)\\&\leq\frac{1}{\lambda_i}\log\left(4L\mathcal{C}D^4N\right)+\max_{c,d,l,s,s',t,k}\frac{\lambda_i(\sqrt{D}\rho_A\rho_\Delta(L-i)l_i)^2\sum_{j=1}^{m}(X_{(j)_{tk}})^2}{2}+(\sqrt{D}\rho_A\rho_\Delta(L-i)l_i)\sqrt{\sum_{j=1}^{m}(X_{(j)_{tk}})^2}\end{aligned}$$

We choose $\lambda_i = \sqrt{\frac{2\log(4L\mathcal{C}D^4N)}{M_i^2 \max_{t,k}\sum_{j=1}^m (X_{(j)_{tk}})^2}}$ which minimizes the above term and obtain the following inequality:

$$\sum_{i=1}^{L} \frac{1}{\lambda_i}\log\left(4L\mathcal{C}D^4N\right) + \max_{t,k} \frac{\lambda_i(\sqrt{D}\rho_A\rho_\Delta(L-i)l_i)^2\sum_{j=1}^m (X_{(j)_{tk}})^2}{2} + (\sqrt{D}\rho_A\rho_\Delta(L-i)l_i)\sqrt{\sum_{j=1}^m (X_{(j)_{tk}})^2}$$
$$\leq \sum_{i=1}^{L}(1+\sqrt{2\log(4L\mathcal{C}D^4N)})M_i\sqrt{\max_{t,k}\sum_{j=1}^m (X_{(j)_{tk}})^2}$$
$$= \sum_{i=1}^{L}(1+\sqrt{2\log(4L\mathcal{C}D^4N)})\sqrt{D}\rho_A\rho_\Delta(L-i)l_i \cdot \sqrt{\max_{t,k}\sum_{j=1}^m (X_{(j)_{tk}})^2}$$
$$\leq (1+\sqrt{2\log(4L\mathcal{C}D^4N)})\sqrt{D}\rho_A\rho_\Delta\sqrt{\max_{t,k}\sum_{j=1}^m (X_{(j)_{tk}})^2}\frac{K}{(K-1)^2}$$

We conclude that:

$$m\mathcal{R}(\mathcal{F}_\rho)$$
$$\leq D^2\Gamma(1+\sqrt{2\log(4L\mathcal{C}D^4N)})\sqrt{\max_{t,k}\sum_{j=1}^m (X_{(j)_{tk}})^2}\frac{K}{(K-1)^2}$$

□

Theorem 5. *Let P be a distribution over $\mathbb{R}^{D\times L}\times[C]$ and $\delta > 0$. Let $S = \{(X_{(j)}, y_{(j)})\}_{j=1}^m$ be a dataset of i.i.d. samples selected from P. Assume that $\forall j \in [m] : ||X_{(j)}||_{\max} \leq 1$. Additionally, suppose $\forall k \in [L], d \in [D] : ||\bar{A}_{dk}||_{\max} < K < 1$. Then, with probability at least $1-\delta$ over the selection of S, for any $f_w \in \mathcal{F}$,*

$$\text{err}_P(f_w) - \frac{1}{m}\sum_{j=1}^m \mathbb{I}[\max_{c\neq c'}(f_w^c(X_{(j)})) + \gamma \geq f_w^{c'}(X_{(j)})]$$
$$= \text{err}_P(f_w) - \text{err}_S^\gamma(f_w) \leq \frac{2\sqrt{2}}{\gamma m}(\Gamma(w) + \frac{1}{D^2N^2})D^2\cdot$$
$$(1+\sqrt{2\log(4L\mathcal{C}D^4N)})\sqrt{\max_{t,k}\sum_{j=1}^m (X_{(j)_{tk}})^2}\frac{K}{(K-1)^2}$$
$$+ 3\sqrt{\frac{\log(2/\delta) + 2\log(D^2N^2\Gamma(w)+2)}{2m}},$$

where the maximum is taken over $t \in [D]$, $k \in [L]$.

Proof. For aesthetic purposes, we define B as S_B and C as S_C. We want to prove the bound for all $f_w \in \mathcal{F}$ where:

$$\mathcal{F} :=$$
$$\{f_w : w = (A, B, C, S_\Delta, W), \forall k \in [L], d \in [D] : ||\bar{A}_{dk}||_{\max} < K < 1\}$$

Let $t \in \mathbb{N}$. Denote:

$$\mathcal{S}(t) := \{f_w \in \mathcal{F}, \Gamma(w) < \frac{t}{D^2N^2}\}$$

Correspondingly subdivide δ as:

$$\delta(t) := \frac{\delta}{t(t+1)}$$

By Lemma 4 and Theorem 4, with probability at least $1-\delta(t)$: for any function $f_w \in \mathcal{S}(t)$, we have the following inequality:

$$\text{err}_P(f_w) - \text{err}_S^\gamma(f_w) \leq \frac{2\sqrt{2}}{\gamma}\cdot\mathcal{R}(\mathcal{S}(t)) + 3\sqrt{\frac{\log(2/\delta(t))}{2m}}.$$

Using the union bound over all possible set $\mathcal{S}(t)$, we establish that the above probabilistic bound holds uniformly for all functions $f_w \in \mathcal{S}(t)$ with probability at least $1-\delta$. Hence, let $f_w \in \mathcal{F}$ with weight vector $w = (A, B, C, S_\Delta, W)$. We

choose the smallest (t) such that, $f_w \in \mathcal{S}(t)$. We have:

$$\begin{aligned}
&\text{err}_P(f_w) - \text{err}_S^{\gamma}(f_w) \leq \frac{2\sqrt{2}}{\gamma} \cdot \mathcal{R}(S(t)) + 3\sqrt{\frac{\log(2/\delta(t))}{2m}} \\
&= \frac{2\sqrt{2}}{\gamma m} \frac{t}{D^2N^2} D^2(1 + \sqrt{2\log(4L\mathcal{C}D^4N)}) \cdot \sqrt{\max_{t,k} \sum_{j=1}^{m} (X_{(j)_{tk}})^2} \frac{K}{(K-1)^2} + 3\sqrt{\frac{\log(2/\delta) + 2\log(t+1)}{2m}} \\
&\leq \frac{2\sqrt{2}}{\gamma m}(\Gamma(w) + \frac{1}{D^2N^2}) D^2(1 + \sqrt{2\log(4L\mathcal{C}D^4N)}) \cdot \sqrt{\max_{t,k} \sum_{j=1}^{m} (X_{(j)_{tk}})^2} \frac{K}{(K-1)^2} 3\sqrt{\frac{\log(2/\delta) + 2\log(D^2N^2\Gamma(w) + 2)}{2m}}
\end{aligned}$$

□

C Additional Related Work on Generalization

Explaining the performance of overparameterized deep neural networks (DNNs) on test data remains a major challenge in deep learning theory. Traditional tools like the PAC-learning framework and VC-dimension often provide vacuous bounds when the number of parameters greatly exceeds the number of data points. To address this, many studies conduct architecture-specific analyses. For instance, Allen et al. analyze the dynamics of stochastic gradient descent on RNNs with ReLU activations, offering optimization and generalization guarantees (Allen-Zhu and Li, 2019). Other works have explored the generalization of RNNs for unseen data and longer sequences under various assumptions (Cohen-Karlik et al., 2022a; Emami et al., 2021; Cohen-Karlik et al., 2022b; Hardt et al., 2018).

There are relatively few results that address modern architectures such as S6 layers. A recent contribution proposes a bound for standard SSMs (Liu and Li, 2024), but it does not extend to Selective SSMs. To address this gap, we propose a new bound specifically for Selective SSMs.