CS 395T: Foundations of Modern Generative AI Fall 2026

Lecture 4: Linear Attention and State Space Models

Prof. Noah Golowich

1 FLOP accounting for one forward pass

We count one scalar multiplication and one scalar addition as two floating-point operations (FLOPs), and consider one sequence of length T. Assume model width d, H heads of width d/H, MLP hidden width d_{ff}, and vocabulary size $N_{\mathcal{V}}$. Given an attention mask $M \in \{0,1\}^{T\times T}$, let

$$P_M := |\{(j,i) \in [T]^2 : M_{ji} = 1\}|$$

be the number of allowed key–query pairs. Thus $P_M = T^2$ for unmasked attention and $P_M = T(T+1)/2$ for causal attention, provided the implementation actually skips the masked triangle.

Operation in one layer	Leading FLOPs
Query, key, and value projections	$6Td^2$
Attention scores across all heads	$2P_M d$
Weighted sums of values	$2P_M d$
Attention output projection	$2Td^2$
Two-matrix MLP	$4Td\,d_{\text{ff}}$
SwiGLU MLP	$6Td\,d_{\text{ff}}$
Norms, RoPE, residuals, and activations	$O(Td) + O(P_M H)$

Consequently, one unmasked attention layer costs

$$8Td^2 + 4T^2 d + O(T^2 H + Td) \tag{1}$$

FLOPs, while an efficiently triangular causal layer replaces $4T^2 d$ by $2T(T+1)d$. A dense implementation that materializes the full score matrix still pays the unmasked count. With the original choice $d_{\text{ff}} = 4d$, the MLP costs $16Td^2$ FLOPs. A parameter-matched SwiGLU with $d_{\text{ff}} \approx 8d/3$ has the same leading count. Multiplying by L gives the cost of the transformer stack. Computing logits at every position adds $2TdN_{\mathcal{V}}$ FLOPs; in autoregressive generation, only the newest position needs to be unembedded.

The displayed counts describe a full-sequence (or *prefill*) forward pass. During autoregressive decoding, a *key–value cache* stores the earlier keys and values. For one new token, the attention computations in each layer then cost $\Theta(Td)$ rather than recomputing all T^2 pairs, while the projections and MLP cost $\Theta(d^2)$. Nevertheless, generating a length-T continuation sums the growing attention cost over $1, \ldots, T$, giving $\Theta(T^2 d)$ attention work in total.

This accounting exposes two different targets for improvement.

- *Get more capacity from the MLP budget.* A mixture-of-experts (MoE) layer has many expert MLPs but routes each token to only a small subset. It can therefore increase parameter count and specialization without evaluating every expert on every token [6]. We will study MoE later in the course.
- *Avoid all T^2 attention pairs.* Sparse or local patterns, low-rank and kernel approximations, and recurrent or state-space replacements can reduce the dependence on T. These methods trade away some combination of exactness, global interactions, or worst-case expressive power. We will now explore such tradeoffs further.

2 Linear attention

Standard attention forms all T^2 query–key interactions. Linear attention replaces the softmax weight matrix by a kernel feature representation whose contractions can be reordered, removing this quadratic dependence on the sequence length.

2.1 From the exponential kernel to a finite feature map

The exponential dot-product kernel—equivalently, the nonconstant part of an RBF kernel after separating its norm factors—has an infinite-dimensional feature representation. Indeed,

$$\exp(\langle q, k\rangle) = \sum_{a=0}^{\infty} \frac{\langle q, k\rangle^a}{a!} = \langle \phi_\infty(q), \phi_\infty(k)\rangle\,, \qquad \phi_\infty(q) := \left(1, q, \frac{q^{\otimes 2}}{\sqrt{2!}}, \frac{q^{\otimes 3}}{\sqrt{3!}}, \dots\right). \tag{2}$$

While the feature map ϕ_∞ is infinite-dimensional, one can approximate softmax attention (or other forms of attention) by considering more general feature maps; this motivates *linear attention*, which we define next.

Definition 2.1 (Unnormalized linear attention)**.** Let the input be

$$X = [x_1, \dots, x_T] \in \mathbb{R}^{d\times T}.$$

Fix a query/key dimension m, a feature dimension r, learned matrices $W_Q, W_K \in \mathbb{R}^{m\times d}$ and $W_V \in \mathbb{R}^{d\times d}$, and a fixed feature map $\phi : \mathbb{R}^m \to \mathbb{R}^r$. For each position $t \in [T]$, set

$$q_t := W_Q x_t, \qquad k_t := W_K x_t, \qquad v_t := W_V x_t.$$

The unmasked linear-attention output $Y = [y_1, \dots, y_T] \in \mathbb{R}^{d\times T}$ is

$$y_t := \sum_{s=1}^{T} \langle \phi(q_t), \phi(k_s)\rangle\; v_s. \tag{3}$$

Unlike softmax attention, the coefficients in (3) need not be nonnegative or sum to one.

2.2 Linear-time computation

Let

$$\Phi_Q := [\phi(q_1), \dots, \phi(q_T)] \in \mathbb{R}^{r\times T}, \quad \Phi_K := [\phi(k_1), \dots, \phi(k_T)] \in \mathbb{R}^{r\times T}, \quad V := [v_1, \dots, v_T] \in \mathbb{R}^{d\times T}.$$

Writing all columns of (3) at once gives

$$Y = V\Phi_K^\top \Phi_Q. \tag{4}$$

Lemma 2.2 (Linear-time unmasked attention)**.** *Once the feature vectors and values are available, the output in (4) can be computed in $O(Tdr)$ arithmetic operations and $O(dr)$ working memory beyond the input and output. In particular, for fixed d and r, the cost is linear in T.*

Proof. A direct evaluation first forms the $T \times T$ matrix $\Phi_K^\top \Phi_Q$. Instead, associativity of matrix multiplication gives

$$Y = (V\Phi_K^\top)\Phi_Q. \tag{5}$$

The matrix

$$S := V\Phi_K^\top = \sum_{s=1}^{T} v_s \phi(k_s)^\top \in \mathbb{R}^{d\times r}$$

takes $O(Tdr)$ operations to compute: each summand is a $d \times r$ outer product. Multiplying S by each of the T columns of Φ_Q takes another $O(Tdr)$ operations. Only the $d \times r$ matrix S must be retained between these two stages, proving the stated memory and time bounds. □

2.3 Causal linear attention is an RNN

For autoregressive modeling, position t may use only positions $s \le t$. The causal version of (3) is therefore

$$y_t := \sum_{s=1}^{t} \langle \phi(q_t), \phi(k_s) \rangle \; v_s. \tag{6}$$

Lemma 2.3 (Linear-time causal attention)**.** *All outputs in* (6) *can be computed in* $O(Tdr)$ *arithmetic operations using a recurrent state of size* dr*.*

Proof. Initialize $S_0 := 0 \in \mathbb{R}^{d\times r}$. For $t = 1, \dots, T$, update

$$S_t := S_{t-1} + v_t \phi(k_t)^\top, \qquad y_t := S_t \phi(q_t). \tag{7}$$

Inducting on t in the first recurrence gives

$$S_t = \sum_{s=1}^{t} v_s \phi(k_s)^\top.$$

Substitution into the second recurrence yields

$$y_t = \sum_{s=1}^{t} v_s \phi(k_s)^\top \phi(q_t) = \sum_{s=1}^{t} \langle \phi(q_t), \phi(k_s) \rangle \, v_s,$$

which is exactly (6). At each position, the outer product and matrix–vector product cost $O(dr)$ operations, and the state S_t has dr entries. □

Equation (7) is an RNN: it updates a fixed-size hidden state from the current key and value and reads the current output using the query.

3 State space models

The recurrence above has a particularly restricted memory update: the previous state is copied unchanged and a rank-one term is added. State space models (SSMs) generalize this update by allowing a time-dependent linear transformation of the hidden state. This allows different state coordinates to persist, decay, or mix before the current input is incorporated.

Following Dao and Gu [2], let N denote the state dimension. For an input $X = [x_1, \dots, x_T] \in \mathbb{R}^{d\times T}$, define hidden states $H_t \in \mathbb{R}^{N\times d}$ and output columns $y_t \in \mathbb{R}^d$ by

$$H_0 := 0, \qquad H_t := A_t H_{t-1} + B_t x_t^\top, \qquad y_t := H_t^\top C_t, \tag{8}$$

where $A_t \in \mathbb{R}^{N\times N}$ and $B_t, C_t \in \mathbb{R}^N$. The output is $Y = [y_1, \dots, y_T] \in \mathbb{R}^{d\times T}$. Here and throughout, vectors are column vectors: in particular, $B_t x_t^\top$ is the $N \times d$ outer product of B_t and x_t, while $H_t^\top C_t \in \mathbb{R}^d$.

In language models, the parameters A_t, B_t, C_t above may be functions of the current token representation; the linear algebra below conditions on the resulting matrices and vectors. Dense A_t would make the recurrence expensive, so practical SSMs impose substantial structure, such as diagonal or diagonal-plus-low-rank matrices. Mamba-2 uses an even more restricted scalar-times-identity transition within each head.

3.1 Matrix transformation and semiseparable structure

For $1 \le s \le t \le T$, define the transition product

$$A_{t:s+1} := A_t A_{t-1} \cdots A_{s+1}, \qquad A_{t:t+1} := I_N.$$

Induction in (8) gives

$$H_t = \sum_{s=1}^{t} A_{t:s+1} B_s x_s^\top, \qquad y_t = \sum_{s=1}^{t} \left(C_t^\top A_{t:s+1} B_s \right) x_s. \tag{9}$$

We may therefore write $Y = XM$, where $M \in \mathbb{R}^{T \times T}$ is defined by

$$M_{st} := \begin{cases} C_t^\top A_{t:s+1} B_s, & s \le t, \\ 0, & s > t. \end{cases} \tag{10}$$

The matrix M is not an arbitrary causal matrix. Its off-diagonal blocks have rank controlled by the state dimension.

Definition 3.1 (N-semiseparable matrix; transpose of Definition 3.1 of [2])**.** An upper-triangular matrix $M \in \mathbb{R}^{T \times T}$ is *N-semiseparable* if every rectangular submatrix lying on or above the diagonal has rank at most N. Equivalently, for all contiguous index intervals $I, J \subseteq [T]$ satisfying $\max I \le \min J$,

$$\operatorname{rank}(M[I, J]) \le N.$$

The integer N is called the order of the semiseparable matrix.

Definition 3.2 (N-sequentially semiseparable representation; transpose of Definition 3.2 of [2])**.** An upper-triangular matrix $M \in \mathbb{R}^{T \times T}$ has an *N-sequentially semiseparable* (SSS) representation if there are vectors and matrices

$$B_1, \ldots, B_T, C_1, \ldots, C_T \in \mathbb{R}^N, \qquad A_1, \ldots, A_T \in \mathbb{R}^{N \times N},$$

such that

$$M_{st} = \begin{cases} C_t^\top A_{t:s+1} B_s, & s \le t, \\ 0, & s > t. \end{cases}$$

We write $M = \mathrm{SSS}(A, B, C)$.

Lemma 3.3 (SSS implies semiseparable; Lemma 3.3 of [2])**.** *Every matrix with an N-SSS representation is N-semiseparable.*

Proof. Fix intervals $I = \{c, \ldots, e\}$ and $J = \{a, \ldots, b\}$ with $e \le a$, so the block $M[I, J]$ lies on or above the diagonal. For $s \in I$ and $t \in J$, split the transition product at the boundary a:

$$M_{st} = \underbrace{(A_{a:s+1} B_s)^\top}_{L_s^\top} \underbrace{A_{t:a+1}^\top C_t}_{R_t}.$$

The empty-product convention covers $t = a$ or $s = a$. Stack the row vectors $L_s^\top$ into a matrix $L \in \mathbb{R}^{|I| \times N}$ and the column vectors R_t into a matrix $R \in \mathbb{R}^{N \times |J|}$. Then

$$M[I, J] = LR.$$

Hence $\operatorname{rank}(M[I, J]) \le N$. Since the choice of I, J was arbitrary, M is N-semiseparable. □

Lemma 3.4 (Semiseparable implies SSS; Proposition 3.4 of [2])**.** *Every N-semiseparable upper-triangular matrix has an N-SSS representation.*

See Appendix A.1 for a proof. Thus the rank condition in Definition 3.1 and the recurrent generators in Definition 3.2 describe the same class of matrices.

3.2 Two computation modes

The setup from the previous section gives two complementary ways to compute the same transformation.

Proposition 3.5 (Compressed SSS multiplication; Proposition 3.6 of [5, 2])**.** *An N-semiseparable matrix of size $T \times T$ has a representation using $O(NT)$ scalars and can be multiplied by a vector, on either side, in $O(NT)$ time and $O(NT)$ space.*

The proof is given in Appendix A.2. Right-multiplying each of the d rows of X by the same matrix multiplies these bounds by d.

1. *Recurrent form.* Equation (8) directly computes the output in one left-to-right pass. With dense transitions, multiplying $A_t H_{t-1}$ costs $O(dN^2)$ per position, hence $O(dN^2T)$ total—or $O(N^2T)$ per scalar input channel. After preprocessing the induced semiseparable matrix into the compressed block-SSS representation of Proposition 3.5, the same transformation takes $O(dNT)$ time, or $O(NT)$ per channel.

2. *Quadratic form.* Materialize the matrix $M = \mathrm{SSS}(A, B, C) \in \mathbb{R}^{T \times T}$ and compute

$$Y = X\,\mathrm{SSS}(A, B, C). \tag{11}$$

   This takes $O(dT^2)$ time and exposes the model as a structured version of masked attention: the zero lower triangle is the causal mask, while each allowed coefficient factors as $C_t^\top A_{t:s+1} B_s$. Thus the attention pattern and its causal mask are not arbitrary; they are determined by the semiseparable matrix $\mathrm{SSS}(A, B, C)$.

The recurrent form is asymptotically preferable for long sequences and autoregressive inference. The quadratic form can nevertheless be useful on short blocks and during model training because it exposes matrix multiplications that are efficient on modern accelerators. Mamba-2 combines these perspectives through blocked algorithms.

3.3 A brief overview of Mamba-2

SSM versus SSD. An *SSM* (state space model) is the general recurrence in (8); in particular, its transition A_t may be a general $N \times N$ matrix, although practical SSMs usually impose additional structure. Mamba-2 considers the special case of scalar-identity transitions

$$A_t = a_t I_N,$$

where $a_t \in \mathbb{R}$. Then $A_{t:s+1} = (\prod_{r=s+1}^{t} a_r) I_N$, so the transition products are encoded by the upper-triangular mask $L \in \mathbb{R}^{T \times T}$ with

$$L_{st} := \begin{cases} \prod_{r=s+1}^{t} a_r, & s \le t, \\ 0, & s > t, \end{cases} \qquad L = \begin{bmatrix} 1 & a_2 & a_2 a_3 & \cdots & a_2 a_3 \cdots a_T \\ 0 & 1 & a_3 & \cdots & a_3 a_4 \cdots a_T \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & a_T \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \tag{12}$$

$$\mathrm{SSS}(A, B, C)_{st} = C_t^\top \cdot B_s \cdot L_{st}. \tag{13}$$

Note that the matrix L in Equation (13) is 1-semiseparable: the coefficient carrying information from position s to position t is the product of the intervening scalar transitions. If $a_t = 1$ for every t, then L is the ordinary causal mask; general, possibly input-dependent a_t give a multiplicatively weighted causal mask. Indeed, if $G_{st} := C_t^\top B_s$, then (10) becomes

$$M = L \odot G, \qquad Y = X(L \odot G),$$

which is masked linear attention with keys B_s, queries C_t, values x_s, and the 1-semiseparable mask L. In the notation of [2], an *SSD layer* is an SSM with this scalar-identity specialization. Thus every SSD layer is an SSM, but a general SSM need not be an SSD layer.

Mamba-2 packages an SSD layer inside a neural-network block designed for efficient language modeling. Within each head, the scalar, input-dependent a_t controls how much earlier information persists, while B_t writes the current input into the state and C_t reads from it.

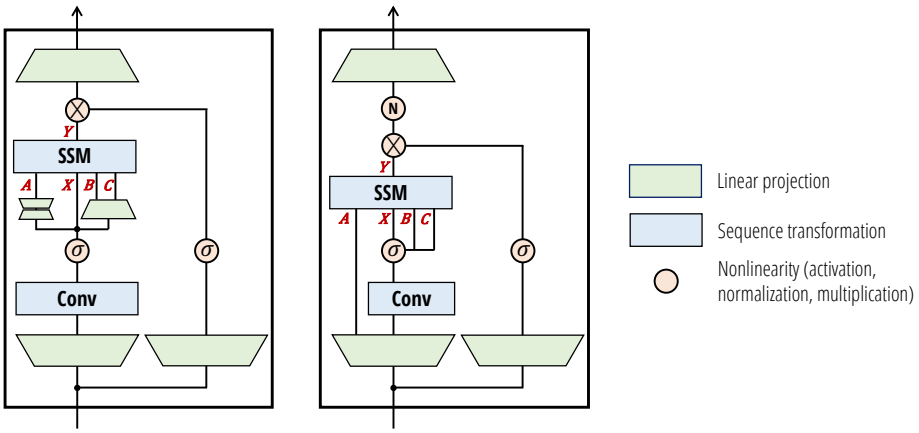

Figure 1: Sequential and parallel Mamba blocks. Mamba-2 uses the parallel block on the right. Copied from Figure 6 of Dao and Gu [2].

The main components in Mamba-2 (Figure 1) are as follows.

- A single input projection produces the SSM input X, its parameters A, B, C, and the multiplicative gate in parallel. Mamba-1 instead produced some of these quantities after earlier transformations, creating sequential projections.
- A short causal convolution and a nonlinear activation preprocess the X branch before the SSD sequence transformation. Feature nonlinearities, by default SiLU, can also be applied to the B and C branches.

- Across the X heads, the B and C projections are shared. Dao and Gu call this the multi-input SSM pattern; under the attention analogy it is a multi-value-attention pattern.
- The SSD output is multiplied by a separate nonlinear gate. An additional normalization is then applied immediately before the final output projection, improving stability in the authors' experiments.
- The same SSD map admits the recurrent computation in (8), useful for autoregressive generation, and the structured matrix view in (11).

4 Conditional barriers for subquadratic alternatives

We now study results of Alman and Yu [1]. In particular, we consider a synthetic setting where we are given a number T of *documents*, each represented by a Boolean vector in $\{0,1\}^\ell$, for some dimension $\ell \in \mathbb{N}$; for instance, you can think of this Boolean vector as a *bag-of-words* representation of each document (or any other feature representation). For two such vectors, define their cosine similarity by

$$\mathrm{sim}(u,v) := \frac{\langle u, v\rangle}{\|u\|_2 \, \|v\|_2} \in [0,1]. \tag{14}$$

Definition 4.1 (Most and least similar documents)**.** Given $v_1, \dots, v_T \in \{0,1\}^\ell \setminus \{0\}$, the problem $\mathrm{MSD}_{T,\ell}$ asks for distinct $i, j \in [T]$ maximizing $\mathrm{sim}(v_i, v_j)$, and $\mathrm{LSD}_{T,\ell}$ asks for distinct i, j minimizing it.

4.1 SETH and Orthogonal Vectors

Let k-SAT be satisfiability for Boolean formulas in conjunctive normal form with at most k literals per clause. We use N for the number of Boolean variables, so it is not confused with the number T of document vectors.

Conjecture 4.2 (Strong Exponential Time Hypothesis (SETH); [3])**.** For every $\delta > 0$, there exists an integer $k \geq 3$ such that k-SAT on N variables cannot be solved in time $O(2^{(1-\delta)N})$.

Definition 4.3 (Orthogonal Vectors)**.** The problem $\mathrm{OV}_{T,\ell}$ is given $v_1, \dots, v_T \in \{0,1\}^\ell$ and asks whether there are distinct $i, j \in [T]$ such that $\langle v_i, v_j\rangle = 0$.

The Orthogonal Vectors Conjecture (OVC) states that, for every $\epsilon > 0$, there is a constant $c > 0$ for which $\mathrm{OV}_{T, c\log T}$ has no $O(T^{2-\epsilon})$-time algorithm. Williams proved that SETH implies OVC [7]. Thus a truly subquadratic algorithm for OV in all logarithmic dimensions would refute SETH.

4.2 Hardness of document similarity

The following statement combines Theorems 1.1 and 1.2 of Alman and Yu.

Theorem 4.4 (Conditional hardness of MSD and LSD; [1])**.** *Assume SETH. For every $\epsilon > 0$, there are constants $c_{\mathrm{L}}, c_{\mathrm{M}} > 0$ such that the following hold.*

1. *For $\ell = c_{\mathrm{L}} \log T$, the problem $\mathrm{LSD}_{T,\ell}$ cannot be solved in $O(T^{2-\epsilon})$ time.*
2. *For $\ell = T^{c_{\mathrm{M}}/\log\log T}$, the problem $\mathrm{MSD}_{T,\ell}$ cannot be solved in $O(T^{2-\epsilon})$ time.*

We next prove the exact-LSD case of their Theorem 3.1 by a direct reduction from Orthogonal Vectors.

Theorem 4.5 (Hardness of LSD; exact case of Theorem 3.1)**.** *Assume SETH (or, more directly, OVC). For every $\epsilon > 0$, there is a constant $c > 0$ such that, when $\ell = c \log T$, there is no algorithm that solves $\mathrm{LSD}_{T,\ell}$ in $O(T^{2-\epsilon})$ time.*

Proof. Fix $\epsilon > 0$, set $\delta := \min\{\epsilon/2, 1/2\}$, and apply OVC with exponent δ. Let $c > 0$ be the resulting constant. Suppose for contradiction that an algorithm $\mathcal{A}$ solves $\mathrm{LSD}_{T,\ell}$ in time $O(T^{2-\epsilon})$ when $\ell = c \log T$.

Consider an instance $v_1, \ldots, v_T \in \{0,1\}^\ell$ of $\mathrm{OV}_{T,\ell}$. If one of the vectors is zero, then for $T \geq 2$ it forms an orthogonal pair with any other vector, and we may return YES. We can check this case in $O(T\ell)$ time. Hence suppose all vectors are nonzero, as required by LSD.

Run $\mathcal{A}$ and let $i \neq j$ be the pair it returns. Because all vectors are nonzero and $\mathcal{A}$ returns a pair of minimum cosine similarity,

$$\langle v_i, v_j \rangle = 0 \quad \Longleftrightarrow \quad \min_{p \neq q} \mathrm{sim}(v_p, v_q) = 0 \quad \Longleftrightarrow \quad \text{the OV instance contains an orthogonal pair.}$$

Thus, after one call to $\mathcal{A}$, we solve $\mathrm{OV}_{T,\ell}$ by computing $\langle v_i, v_j \rangle$ and returning YES exactly when it is zero. The total running time is

$$O(T\ell) + O(T^{2-\epsilon}) = O(T^{2-\delta})$$

for all sufficiently large T, because $\ell = c \log T$. This contradicts the choice of c from OVC, and hence also contradicts SETH. □

4.3 A single attention head can solve OV

The lower bound above is informative only if standard attention can represent the hard interaction, which we now proceed to show.

Definition 4.6 (Single-unit transformer of Alman and Yu; adapted from [1])**.** Fix a sequence length T, embedding dimension d, and query/key width m. Let $\phi_{\mathrm{in}} : \mathbb{R}^{d \times T} \to \mathbb{R}^{d \times T}$ be an input MLP whose output is the sequence $X \in \mathbb{R}^{d \times T}$. Given $W_Q, W_K \in \mathbb{R}^{m \times d}$ and $W_V \in \mathbb{R}^{d \times d}$, the transformer sequence transformation has the form

$$Y = \mathsf{TF}(E) := W_V X \, \mathrm{softmax}\left(\frac{X^\top W_K^\top W_Q X}{\sqrt{m}} \right) \in \mathbb{R}^{d \times T}, \qquad X := \phi_{\mathrm{in}}(E). \tag{15}$$

In words, the model consists of an input MLP followed by one unmasked attention unit.

Theorem 4.7 (Representing OV with one attention head; Theorem 4.1 of [1])**.** *For every $T \geq 2$ and $\ell \geq 1$, there is a transformer of the form (15) that solves $\mathrm{OV}_{T,\ell}$ on the augmented input $\widehat{E} = [v_1, \ldots, v_T, 0] \in \mathbb{R}^{\ell \times (T+1)}$. Its sequence length is $T+1$, $d = \ell$, and query/key width $m \geq \ell + 1$. The decision rule reads the first coordinate of the first T output columns.*

Proof. Let $v_1, \ldots, v_T \in \{0,1\}^\ell$ be the input. A zero vector can be detected by the input map; when $T \geq 2$ it immediately gives a YES instance. We therefore assume every v_i is nonzero. Append one additional vector $v_{T+1} := 0 \in \mathbb{R}^\ell$ and take the input map to be the identity on the augmented input, so $X = \phi_{\mathrm{in}}(\widehat{E}) = [v_1, \ldots, v_T, v_{T+1}]$.

Choose a constant $C > 0$ momentarily. Because $m \geq \ell + 1$, we can choose query and key matrices whose scaled product satisfies

$$\frac{W_K^\top W_Q}{\sqrt{m}} = -C \log T \, I_\ell. \tag{16}$$

Let $e_1 \in \mathbb{R}^\ell$ be the first standard basis vector and set $W_V = e_1 \mathbf{1}_\ell^\top$. Thus the first value coordinate of v_j is $\|v_j\|_1$, every other value coordinate is zero, and the value of v_{T+1} is zero. If $r_i := e_1^\top Y[:, i]$ is the first output coordinate at query position $i \in [T]$, then unmasked attention returns

$$r_i = \frac{\sum_{j=1}^{T} T^{-C\langle v_i, v_j\rangle} \|v_j\|_1}{1 + \sum_{k=1}^{T} T^{-C\langle v_i, v_k\rangle}}. \tag{17}$$

The 1 in the denominator is the exponential score of $v_{T+1} = 0$; this additional vector contributes zero to the numerator.

Suppose first that $\langle v_{i_\star}, v_{j_\star}\rangle = 0$ for some distinct $i_\star, j_\star \in [T]$. The $j_\star$ term in the numerator of (17) is $\|v_{j_\star}\|_1 \geq 1$, while every exponential in the denominator is at most one. Hence

$$r_{i_\star} \geq \frac{1}{T+1}. \tag{18}$$

Now suppose there is no orthogonal pair. Since all vectors are nonzero, $\langle v_i, v_j\rangle \geq 1$ for every $i, j \in [T]$, including $i = j$. The denominator of (17) is at least one, and $\|v_j\|_1 \leq \ell$, so

$$r_i \leq T\ell\, T^{-C} = \ell T^{1-C} \qquad \text{for every } i \in [T]. \tag{19}$$

Alman and Yu take $C = 3$; in their logarithmic and subpolynomial dimension regimes this is less than $(T+1)^{-3/2}$ for all sufficiently large T. For arbitrary ℓ, the same construction is made uniform by taking

$$C := 4 + \lceil \log_T \ell \rceil,$$

which makes the right-hand side of (19) at most T^{-3} and hence strictly less than $(T+1)^{-3/2}$ for $T \geq 2$.

There is now a nonempty gap between the no-instance bound $a := (T+1)^{-3/2}$ and the yes-instance bound $b := (T+1)^{-1}$.

□

The construction exposes why pairwise attention is useful here: exponentiating a negative inner product gives an observable constant-scale contribution precisely when a query has an orthogonal partner. The appended all-zero vector stabilizes the denominator and, because its value is zero, does not itself create a false positive.

A Proofs for semiseparable matrices

A.1 The converse SSS representation

Let $M \in \mathbb{R}^{T\times T}$ be N-semiseparable. For each $k \in [T]$, consider the upper-right block

$$G_k := M[\{1, \ldots, k\}, \{k, \ldots, T\}].$$

This entire block lies on or above the diagonal, so $\operatorname{rank}(G_k) \leq N$. Choose a rank factorization, padded with zero rows or columns if necessary,

$$G_k = U_k V_k, \qquad U_k \in \mathbb{R}^{k\times N}, \quad V_k \in \mathbb{R}^{N\times(T-k+1)}, \tag{20}$$

such that the rows of V_k span the row space of G_k.

Delete the first column of V_k. The rows of the resulting matrix span the row space of

$$M[\{1, \ldots, k\}, \{k+1, \ldots, T\}],$$

which is formed by the first k rows of G_{k+1}. This row space is therefore contained in the row space of V_{k+1}. Consequently, for each $k < T$, there is a matrix $A_{k+1} \in \mathbb{R}^{N\times N}$ such that

$$V_k[:, 2 : T-k+1] = A_{k+1}^\top V_{k+1}. \tag{21}$$

Let $B_k^\top$ be the last row of U_k, and let C_k be the first column of V_k.

Fix $1 \le s \le t \le T$. From (20), the last row of G_s is $B_s^\top V_s$, so M_{st} is entry $t-s+1$ of this row. Repeatedly applying (21) gives

$$V_s[:, t-s+1] = A_{s+1}^\top A_{s+2}^\top \cdots A_t^\top V_t[:, 1] = A_{t:s+1}^\top C_t.$$

Therefore

$$M_{st} = B_s^\top A_{t:s+1}^\top C_t = C_t^\top A_{t:s+1} B_s.$$

For $s > t$, both the SSS formula and M_{st} are zero because M is upper triangular. Hence $M = \mathrm{SSS}(A, B, C)$, proving Lemma 3.4.

A.2 Compressed SSS multiplication

In this section we prove Proposition 3.5.

If $T \le N$, storing M densely and using ordinary matrix–vector multiplication costs $O(T^2) \le O(NT)$, so suppose $T > N$. Partition $[T]$ into $m := \lceil T/N \rceil$ consecutive blocks $I_1, \ldots, I_m$, each of size $b_i := |I_i| \le N$. For each block boundary $k \in [m-1]$, define

$$\mathcal{I}_{\le k} := I_1 \cup \cdots \cup I_k, \qquad \mathcal{I}_{>k} := I_{k+1} \cup \cdots \cup I_m,$$

and consider the matrix crossing that boundary,

$$H_k := M[\mathcal{I}_{\le k}, \mathcal{I}_{>k}].$$

This entire matrix lies strictly above the diagonal, so $\mathrm{rank}(H_k) \le N$. From Lemma 3.4, we may choose a rank factorization

$$H_k = U_k V_k, \qquad U_k \in \mathbb{R}^{|\mathcal{I}_{\le k}|\times N}, \quad V_k \in \mathbb{R}^{N\times|\mathcal{I}_{>k}|}, \tag{22}$$

padded with zeros if necessary, such that the rows of V_k span the row space of H_k.

We next compare the factorizations at two consecutive block boundaries. Delete from V_k the columns indexed by I_{k+1}. The rows of the remaining matrix span the row space of

$$M[\mathcal{I}_{\le k}, I_{k+2} \cup \cdots \cup I_m].$$

This matrix consists of the first $|\mathcal{I}_{\le k}|$ rows of H_{k+1}, so its row space is contained in the row space of V_{k+1}. Consequently, for every $k \in [m-2]$ there is a matrix $R_{k+1} \in \mathbb{R}^{N\times N}$ such that

$$V_k[:, I_{k+2} \cup \cdots \cup I_m] = R_{k+1}^\top V_{k+1}. \tag{23}$$

Thus, moving the cut by one block plays the same role as moving it by one position in (21): instead of deleting one column, we delete all the columns in the block passed by the cut.

Define matrices

$$Q_i^\top := U_i[I_i, :] \in \mathbb{R}^{b_i\times N} \quad (i < m), \qquad P_j^\top := V_{j-1}[:, I_j] \in \mathbb{R}^{N\times b_j} \quad (j > 1),$$

and set P_1 and Q_m to zero. Also set $D_i := M[I_i, I_i]$. For $i < j$, restricting (22) to rows I_i and columns I_j gives

$$M[I_i, I_j] = Q_i^\top V_i[:, I_j].$$

Repeatedly applying (23), once at each intervening boundary, gives

$$V_i[:, I_j] = R_{i+1}^\top R_{i+2}^\top \cdots R_{j-1}^\top P_j^\top.$$

We have therefore constructed matrices

$$D_i \in \mathbb{R}^{b_i \times b_i}, \quad P_i \in \mathbb{R}^{b_i \times N}, \quad Q_i \in \mathbb{R}^{N \times b_i}, \quad R_i \in \mathbb{R}^{N \times N}$$

such that $D_i = M[I_i, I_i]$ and, whenever $i < j$,

$$M[I_i, I_j] = Q_i^\top R_{i+1}^\top R_{i+2}^\top \cdots R_{j-1}^\top P_j^\top, \tag{24}$$

where the product of transition matrices is empty when $j = i + 1$.

The diagonal blocks use

$$\sum_{i=1}^{m} b_i^2 \le N \sum_{i=1}^{m} b_i = NT$$

scalars. The P_i and Q_i together use $O(NT)$ scalars, and the $m - 2$ transition matrices use $O(mN^2) = O(NT)$ scalars. Thus the representation has size $O(NT)$.

It remains to show that right multiplication uses the compressed representation directly. Partition $z \in \mathbb{R}^T$ into blocks $z_i \in \mathbb{R}^{b_i}$. Initialize $s_1^\top := 0 \in \mathbb{R}^{1 \times N}$ and compute

$$w_1^\top := z_1^\top D_1, \qquad s_2^\top := z_1^\top Q_1^\top.$$

For $i = 2, \ldots, m$, compute

$$w_i^\top := z_i^\top D_i + s_i^\top P_i^\top, \qquad s_{i+1}^\top := s_i^\top R_i^\top + z_i^\top Q_i^\top \quad \text{when } i < m. \tag{25}$$

Induction on i shows that

$$s_i^\top = \sum_{j<i} z_j^\top Q_j^\top R_{j+1}^\top \cdots R_{i-1}^\top,$$

so (24) implies that the concatenation of $w_1^\top, \ldots, w_m^\top$ is $z^\top M$. Each block step costs $O(N^2)$ operations, and there are $O(T/N)$ blocks, giving $O(NT)$ time. Storing the representation takes $O(NT)$ space, while the recurrence itself needs only $O(N)$ additional working space. This proves Proposition 3.5.

References

[1] Josh Alman and Hantao Yu. Fundamental limitations on subquadratic alternatives to transformers. In *International Conference on Learning Representations (ICLR)*, 2025. OpenReview.

[2] Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. arXiv:2405.21060, 2024. arXiv.

[3] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

[4] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5156–5165, 2020. PMLR.

[5] Clément Pernet, Hippolyte Signargout, and Gilles Villard. Exact computations with quasiseparable matrices. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, 2023. doi:10.1145/3597066.3597102.

[6] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017. OpenReview.

[7] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2–3):357–365, 2005. doi:10.1016/j.tcs.2005.09.023.

AI Use. I prepared these lecture notes with the aid of AI assistance; I take full responsibility for all content in the notes.