Learnable Low-Rank Polynomial Sketch for Effective Linear Attention

Anonymous Authors[1]

Abstract

Softmax attention in Transformers suffers from quadratic complexity in sequence length, making it impractical for long-context applications. Linear attention alleviates this issue by replacing the exponential kernel with alternative functions that enable linear-time computation. Among existing linear attention approaches, recent studies have shown that polynomial kernels are particularly effective, as they exhibit sparse and spiky behavior similar to softmax attention, which emphasizes large dot products while suppressing irrelevant interactions. However, the exact computation of high-degree polynomial kernels is infeasible for high-dimensional representations. As a result, prior work relies on approximate polynomial kernels. This introduces a non-negligible approximation error. In this paper, we show that, from the perspective of polynomial kernel approximation, existing linear attention methods are still suboptimal. We propose **L**earnable **Lo**w-Rank **P**olynomial **S**ketch (LLoPS), a principled and flexible framework for approximating polynomial kernels with linear attention. Our method learns a low-rank polynomial sketch that provably achieves a strictly smaller approximation error than existing approaches. Experiment results show that LLoPS achieves the highest performance across extensive benchmarks, comparing with various linear attention baselines.

1. Introduction

Transformers have become the dominant architecture for sequence modeling across natural language processing, vision, and multimodal tasks. Despite their empirical success, the quadratic time and memory complexity of the attention mechanism with respect to sequence length severely limits its scalability to long-context settings. This bottleneck is particularly problematic in emerging applications such as long-document understanding, code modeling, and retrieval-augmented generation, where context lengths can easily reach tens or hundreds of thousands of tokens.

Linear attention methods (Xiong et al., 2021; Team et al., 2025; Schlag et al., 2021) address this challenge by retaining the core attention formulation while reducing its complexity from quadratic to linear in sequence length. By rewriting softmax attention as a kernelized dot-product and exploiting associativity, linear attention enables streaming computation with constant memory. Recent large-scale deployments (Team et al., 2025) demonstrate that linear attention can attain a strong empirical performance while supporting extremely long contexts, thus making it a promising direction for next-generation foundation models.

Existing work on linear attention can be broadly categorized into two complementary directions. The first focuses on improving the state update rule, designing better ways to accumulate and normalize the running key-value statistics over time (Team et al., 2025; Yang et al., 2024). The second direction, which this paper focuses on, aims to improve the feature map of query and key vectors so that the induced kernel more faithfully approximates softmax attention (Arora et al., 2024; Zhang et al., 2024; Han et al., 2023).

Prior studies on feature map design have observed that high-order polynomial kernels can approximate the softmax kernel surprisingly well (Kacham et al., 2023; Arora et al., 2024; Zhang et al., 2024). Intuitively, both exponential and polynomial kernels can exhibit sparse, spiky behavior that emphasizes large dot-products while suppressing irrelevant interactions. However, computing exact polynomial kernels is infeasible for high degree kernel and high-dimensional vectors, as the computation complexity grows exponentially with the polynomial degree.

To address this challenge, PolySketchFormer (Kacham et al., 2023) exploits a data-independent sketching technique to approximate exact polynomial kernels. More recently, BASED (Arora et al., 2024) realizes state-of-the-art performance in language modeling by combining linear attention and sliding window attention. In their linear attention formulation, queries and keys are first compressed into a low-dimensional space and then the exact polynomial kernel

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

is computed in that space.

In this paper, we show that from the perspective of polynomial kernel approximation, the previous methods that approximate polynomial kernel are suboptimal. Our key insight is that, when the output feature dimension is fixed, there exists a theoretically optimal sketch that minimizes the approximation error to the target polynomial kernel. We prove that a linear projection for the p-th order outer products of input vectors can achieve this optimal approximation, whereas existing approaches such as BASED (Arora et al., 2024) and PolySketchFormer (Kacham et al., 2023) can be viewed as constrained or data-independent approximations to this optimal mapping. Building on this observation, we further propose a structured decomposition of the optimal linear mapping using low-rank tensor rank decompositions (also known as CP decompositions) and column-wise Kronecker products, which significantly reduces computational costs. We provide theoretical analysis showing that this structured approximation is strictly tighter than prior feature-map constructions under the same dimensional budget.

Our contributions are summarized as follows: (1) We provide a theoretical characterization of the optimal sketching strategy for approximating polynomial kernels in the context of linear attention. (2) Motivated by this analysis, we propose a novel **L**earnable **Lo**w-Rank **P**olynomial **S**ketch (LLoPS), a linear attention kernel that accurately approximates polynomial attention while remaining computationally efficient. (3) We further derive approximation error analysis for LLoPS and show that several existing linear attention kernels can be interpreted as special cases within our unified framework. (4) Finally, we demonstrate through experiments that LLoPS consistently improves the effectiveness of linear attention transformers across a broad range of benchmarks. Our code will be released upon acceptance.

2. Preliminaries

2.1. Softmax-based Full Attention

Given a sequence of queries $\boldsymbol{Q} \in \mathbb{R}^{n\times d}$, keys $\boldsymbol{K} \in \mathbb{R}^{n\times d}$, and values $\boldsymbol{V} \in \mathbb{R}^{n\times d_v}$, standard Transformers compute attention as

$$\mathrm{Attn}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\boldsymbol{V}, \qquad (1)$$

where the softmax is applied row-wise. Because this formulation has $\mathcal{O}(n^2)$ time and memory complexity due to the explicit computation of the attention matrix $\boldsymbol{Q}\boldsymbol{K}^\top$, it is prohibitively expensive for long input sequences.

2.2. Linear Attention

Previous studies have observed that softmax attention can be interpreted as kernelized dot-product attention with an exponential kernel. By approximating the exponential function with a feature map $\phi : \mathbb{R}^d \to \mathbb{R}^m$ and exploiting associativity, attention can be rewritten as

$$\mathrm{Attn}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \big(\phi(\boldsymbol{Q})\phi(\boldsymbol{K})^\top\big)\,\boldsymbol{V} = \phi(\boldsymbol{Q})\big(\phi(\boldsymbol{K})^\top\boldsymbol{V}\big), \qquad (2)$$

where $\phi(\boldsymbol{Q}), \phi(\boldsymbol{K}) \in \mathbb{R}^{n\times m}$ are mapped features with dimension m and $\phi(\boldsymbol{Q})\phi(\boldsymbol{K})^\top$ is the inner product in the new feature space that approximates $\mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)$.[1] Crucially, the overall complexity is reduced from quadratic to linear in the sequence length n. This enables streaming computation and a constant memory state, which are key advantages of linear attention.

Recent work reveals that polynomial kernels provide a strong approximation to the exponential kernel underlying softmax attention. In particular, a degree-p polynomial kernel, denoted as $\kappa(\cdot,\cdot)$, takes the form

$$\kappa(\boldsymbol{q},\boldsymbol{k}) = (\boldsymbol{q}^\top\boldsymbol{k})^p. \qquad (3)$$

where $\boldsymbol{q},\boldsymbol{k} \in \mathbb{R}^d$ are query and key vectors.

However, computing the exact polynomial feature map required by linear attention involves expanding all degree-p monomials, i.e., $\psi_p(\boldsymbol{x}) = \mathrm{vec}(\boldsymbol{x}^{\otimes p})$, where $\otimes p$ is p-th order outer product, whose dimensionality scales as $\mathcal{O}(d^p)$. This makes exact polynomial kernels impractical for real-world models.

BASED addresses this issue by first compressing the query and key vectors into a low-dimensional space using a linear projection,

$$\tilde{\boldsymbol{q}} = \boldsymbol{W}_q\boldsymbol{q}, \quad \tilde{\boldsymbol{k}} = \boldsymbol{W}_k\boldsymbol{k}, \qquad \boldsymbol{W}_q,\boldsymbol{W}_k \in \mathbb{R}^{r\times d},\ r \ll d, \qquad (4)$$

and then computing exact polynomial features in the compressed space. In practice, BASED utilizes a second-order polynomial kernel

$$\kappa(\boldsymbol{q},\boldsymbol{k}) = \psi_2(\boldsymbol{q})^\top\psi_2(\boldsymbol{k}), \qquad (5)$$

which corresponds to the quadratic term in the Taylor expansion of the exponential kernel.[2]

By combining this polynomial linear attention with sliding-window attention to capture local interactions, BASED achieves state-of-the-art language modeling performance while maintaining favorable computational efficiency.

2.3. Related Work

We broadly categorize prior studies on linear attention into two directions: (i) the design of feature maps for query and

[1] We use $\phi(\boldsymbol{Q})$ to denote applying $\phi(\cdot)$ to each row vector in $\boldsymbol{Q}$. Similar for $\phi(\boldsymbol{K})$ and later usages.

[2] BASED proposes to use the Taylor expansion $1 + x + x^2/2$. For simplicity, we omit the constant and linear terms, as they can be handled by concatenation and do not affect the core analysis.

key vectors, and (ii) the design of state update rules.

Feature Map Design. This line of work focuses on constructing efficient and expressive feature maps $\phi(\cdot)$ such that the softmax kernel $\exp(\boldsymbol{q}^\top \boldsymbol{k})$ can be approximated by an inner product $\phi(\boldsymbol{q})^\top \phi(\boldsymbol{k})$ (Katharopoulos et al., 2020; Choromanski et al., 2020; Qin et al., 2022; Keles et al., 2023; Meng et al., 2025). Among the many design choices, several recent studies have identified polynomial kernels as particularly effective approximations, as they share important inductive biases with softmax attention, such as sparsity and sharp selectivity (Han et al., 2023; Zhang et al., 2024; Kacham et al., 2023; Arora et al., 2024). As discussed by (Arora et al., 2024), many existing feature maps are substantially less effective than polynomial-kernel-based approaches when evaluated in realistic settings. Moreover, for current methods that explicitly approximate polynomial kernels (Han et al., 2023; Kacham et al., 2023; Arora et al., 2024), their approximation accuracy is still limited. Our work finds that these approximations can be further improved through a more principled and flexible feature-map construction. Overall, there remains significant room for boosting the effectiveness of linear attention feature maps.

Additionally, despite the extensive exploration of feature map design, many prior works do not assess performance on large-scale, real-world language modeling tasks. In some cases, linear attention variants are reported to outperform softmax attention, which contrasts with empirical observations in large language models where linear attention usually has sub-optimal effectiveness.

State Update Rules. Another line of work focuses on designing improved state update rules for linear attention that aim to stabilize training and better capture long-range dependencies (Schlag et al., 2021; Yang et al., 2024; Team et al., 2025). These methods often resemble state space models (Gu & Dao, 2024; Peng et al., 2023) in that they maintain and update a recurrent state, rather than strictly adhering to the standard attention formulation.

Importantly, these approaches are largely orthogonal to feature map design: their update rules can be combined with different feature maps by replacing $\boldsymbol{q}$ and $\boldsymbol{k}$ with $\phi(\boldsymbol{q})$ and $\phi(\boldsymbol{k})$. Since our work focuses exclusively on improving feature maps for polynomial kernel approximation, we do not directly compare with methods that primarily modify the state update mechanism.

3. Methodology

Polynomial kernel is considered to be a good candidate to replace softmax kernel (Kacham et al., 2023; Zhang et al., 2024; Arora et al., 2024). A key reason for the effectiveness of softmax attention is its sparse and spiky attention distribution: for a given query, only a small subset of keys receives large attention weights, while the majority of keys are assigned values close to zero. This behavior arises from the exponential function, which sharply amplifies differences in inner products, effectively separating keys that are well aligned with the query from those that are not.

Polynomial kernels exhibit a similar spikiness property, as higher-order powers increasingly emphasize large inner products while suppressing smaller ones. Besides, polynomial functions are often used to approximate exponential function (e.g., Taylor expansion) (Zhang et al., 2024; Arora et al., 2024). Thus, a polynomial kernel is a natural and effective alternative to the exponential kernel in the context of linear attention. However, as discussed in Sec 2.2, an exact computation of polynomial kernels becomes impractical for high degrees or high-dimensional vectors due to the exponential growth of dimensionality.

Therefore, our work focuses on designing feature maps for queries and keys separately, denoted by $\phi_Q(\boldsymbol{Q}), \phi_K(\boldsymbol{K}) : \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times m}$, that accurately approximate polynomial kernels while remaining computationally efficient. Specifically, we aim to minimize the following approximation error:

$$\left\|\phi_Q(\boldsymbol{Q})\phi_K(\boldsymbol{K})^\top - (\boldsymbol{Q}\boldsymbol{K}^\top)^{*p}\right\|_F^2, \tag{6}$$

where $*p$ is the element-wise p-th power and $\|\cdot\|_f$ is Frobenius norm.

To achieve this goal, we introduce **L**earnable **Lo**w-Rank **P**olynomial **S**ketch (LLoPS). We first demonstrate that, when the output dimension m is fixed, complicated feature maps are unnecessary to achieve the best compression. Instead, linear projections of $\boldsymbol{q}^{\otimes p}$ and $\boldsymbol{k}^{\otimes p}$ are sufficient to attain the theoretically optimal approximation to the target polynomial kernel. Building on this, we further decompose the optimal linear projection and approximate it using a low-rank CP (CANDECOMP/PARAFAC) decomposition (Papalexakis, 2016), which is also known as tensor rank decomposition, that enables a practical computation with linear attention while preserving the approximation quality. Finally, we provide a detailed error analysis of the proposed feature maps and elucidate their connections to existing linear attention kernels.

3.1. Optimal Sketch of Polynomial Kernel

We first consider the optimal compression of polynomial kernel when the output dimension m is fixed. For clarity, let $\psi_p : \mathbb{R}^d \to \mathbb{R}^{d^p}$, where $\psi_p(\boldsymbol{x}) = \text{vec}(\boldsymbol{x}^{\otimes p})$; let $\boldsymbol{A} = (\boldsymbol{Q}\boldsymbol{K}^\top)^{*p} = \psi_p(\boldsymbol{Q})\psi_p(\boldsymbol{K})^\top$[3]; and let $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ be the SVD decomposition of $\boldsymbol{A}$ where $\boldsymbol{\Sigma} = diag(\sigma_1, \cdots, \sigma_n)$.

Since $\phi_Q(\boldsymbol{Q})$ and $\phi_K(\boldsymbol{K})$ are both an $n \times m$ matrix ($n \gg$

[3]Function ψ_p is applied to each row of the matrices.

m), the rank of $\phi_Q(\boldsymbol{Q})\phi_K(\boldsymbol{K})^\top$ is at most m. By the property of SVD decomposition, the minimum possible loss of Eq. 6 is $\sum_{i=m+1}^{n}\sigma_i^2$ where σ_i is the i-th singular value. It is achieved when

$$\phi_Q(\boldsymbol{Q})\phi_K(\boldsymbol{K})^\top = \boldsymbol{U}\boldsymbol{\Sigma_m}\boldsymbol{V}^\top \tag{7}$$

where $\boldsymbol{\Sigma}_m = diag(\sigma_1, \cdots, \sigma_m, 0, \cdots, 0)$.

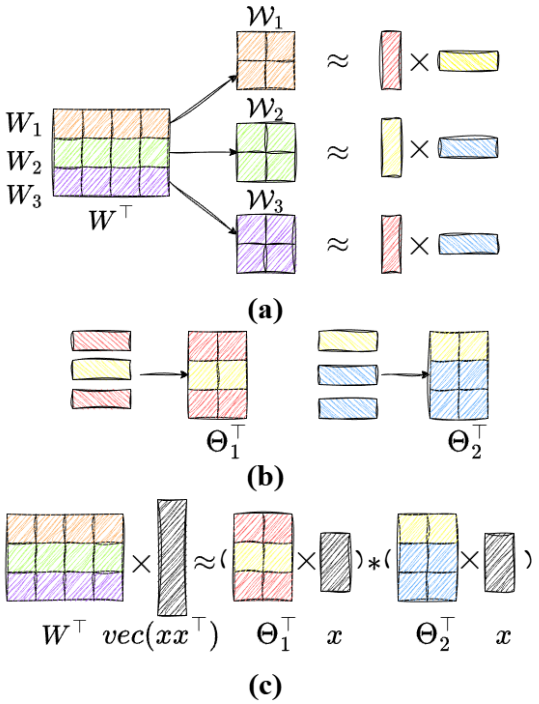


Figure 1. Illustration of LLoPS with an example where $p = 2, d = 2, m = 3$. (a) Given the optimal linear projection matrix $\boldsymbol{W}$, each column of $\boldsymbol{W}$ can be written as a 2 by 2 matrix, and be approximated by its rank-1 CP decomposition; (b) The rank-1 CP decomposition can be stacked into two weight matrices $\boldsymbol{\Theta_1}, \boldsymbol{\Theta_2}$; (c) In this way, the linear projection $\boldsymbol{W}^\top \text{vec}(\boldsymbol{x}\boldsymbol{x}^\top)$ can be approximated as $(\boldsymbol{\Theta_1}^\top\boldsymbol{x}) * (\boldsymbol{\Theta_2}^\top\boldsymbol{x})$ where $*$ is an element-wise product.

The following theorem shows how to achieve the minimum approximation error.

Theorem 3.1 (Optimal Polynomial Sketch)**.** *Let* $\psi_p(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^{d^p} = vec(\boldsymbol{x}^{\otimes p})$. *There exists* $\boldsymbol{W}_Q^*, \boldsymbol{W}_K^* \in \mathbb{R}^{d^p \times m}$ *such that* $\phi_Q^*(\boldsymbol{Q})\phi_K^*(\boldsymbol{K})^\top = \boldsymbol{U}\boldsymbol{\Sigma_m}\boldsymbol{V}^\top$ *where*

$$\begin{aligned}\phi_Q^*(\boldsymbol{Q}) &= \psi_p(\boldsymbol{Q})\boldsymbol{W}_Q^* \\ \phi_K^*(\boldsymbol{K}) &= \psi_p(\boldsymbol{K})\boldsymbol{W}_K^*\end{aligned} \tag{8}$$

The proof is illustrated in Appendix A.1.

Theorem 3.1 suggests that we could compress $\psi_p(\boldsymbol{Q}), \psi_p(\boldsymbol{K})$ into a m-dimensional space via linear mapping, while achieving the minimal approximation error stated in Eq 7.

3.2. Learnable Low-Rank Polynomial Sketch

Although Theorem 3.1 illustrates how to achieve the optimal approximation, since $\boldsymbol{W}_Q^*$ and $\boldsymbol{W}_K^*$ have d^p rows, the linear projection is still too computationally expensive. To address this problem, we propose to decompose and approximate the linear projection matrices utilizing their low-rank CP decompositions (also known as tensor rank decomposition (Papalexakis, 2016)). Figure 1 depicts an example of our decomposed sketch for a toy example where $p = 2, d = 2, m = 3$.

Let $\boldsymbol{W}$ denote $\boldsymbol{W}_Q^*$ or $\boldsymbol{W}_K^*$, and let $\boldsymbol{W}_i \in \mathbb{R}^{d^p}$ $(1 \le i \le m)$ denote the i-th column of $\boldsymbol{W}$. We can write $\boldsymbol{W}_i$ as a (p)-way tensor $\mathcal{W}_i \in \mathbb{R}^{d\times\cdots\times d}$. According to CP decomposition, $\mathcal{W}_i$ can be represented, with a sufficiently large R_i, as a linear combination of R_i rank-1 tensors:

$$\mathcal{W}_i = \sum_{r=1}^{R_i} \lambda_r^{(i)} \boldsymbol{w}_{1,r}^{(i)} \otimes \boldsymbol{w}_{2,r}^{(i)} \otimes \cdots \otimes \boldsymbol{w}_{p,r}^{(i)} \tag{9}$$

where $\lambda_r^{(i)} \in \mathbb{R}$, $\boldsymbol{w}_{j,r}^{(i)} \in \mathbb{R}^d$ $(1 \le j \le p)$, and $\otimes$ denotes outer product.

Inspired by the CP decomposition in Eq. 9, the key idea of LLoPS is as follows. Each row of $\boldsymbol{W}$ is expressed as a sum of rank-one outer products of learnable vectors, as illustrated in Figure 1(a). These vectors are trained implicitly to approximate the low-rank CP decomposition of each row vector $\boldsymbol{W}_i$. We then organize the learned factors into a collection of projection matrices, as shown in Figure 1(b), which enables an efficient computation of the feature map, avoiding explicitly constructing the high-dimensional polynomial features $\psi_p(\boldsymbol{Q})$ and $\psi_p(\boldsymbol{K})$ through a sequence of lightweight matrix multiplications and element-wise operations instead, as depicted in Figure 1(c).

Specifically, $\mathcal{W}_i$ can be approximated with its rank-t CP decomposition, i.e.,

$$\mathcal{W}_i \approx \sum_{r=1}^{t} \lambda_r^{(i)} \boldsymbol{w}_{1,r}^{(i)} \otimes \boldsymbol{w}_{2,r}^{(i)} \otimes \cdots \otimes \boldsymbol{w}_{p,r}^{(i)} \triangleq \hat{\mathcal{W}}_i \tag{10}$$

By reshaping the tensor back to vector, we have an approximate column vector $\hat{\boldsymbol{W}}_i = \text{vec}(\hat{\mathcal{W}}_i)$ and thus the approximate linear projector $\hat{\boldsymbol{W}} = (\hat{\boldsymbol{W}}_1, \cdots, \hat{\boldsymbol{W}}_m) \in \mathbb{R}^{d^p \times m}$. Next, we demonstrate how to efficiently compute $\psi_p(\boldsymbol{Q})\hat{\boldsymbol{W}}$ and $\psi_p(\boldsymbol{K})\hat{\boldsymbol{W}}$ without explicitly computing $\psi_p(\boldsymbol{Q})$, $\psi_p(\boldsymbol{K})$, and $\hat{\boldsymbol{W}}$, which are with extremely high dimensions.

Notice that the inner product between the outer product of vectors $\{\boldsymbol{w}_{1,r}, \ldots, \boldsymbol{w}_{p,r}\}$ and $\text{vec}(\boldsymbol{x}^{\otimes p})$ equals the product of $\{\langle \boldsymbol{w}_{j,r}, \boldsymbol{x}\rangle : 1 \le j \le p\}$, i.e.,

$$\langle \text{vec}(\boldsymbol{w}_{1,r} \otimes \cdots \otimes \boldsymbol{w}_{p,r}), \text{vec}(\boldsymbol{x}^{\otimes p})\rangle = \prod_{j=1}^{p} \langle \boldsymbol{w}_{j,r}, \boldsymbol{x}\rangle \tag{11}$$

So we have

$$\langle \hat{\boldsymbol{W}}_i, \mathrm{vec}(\boldsymbol{x}^{\otimes p})\rangle = \sum_{r=1}^{t} \lambda_r^{(i)} \prod_{j=1}^{p} \langle \boldsymbol{w}_{j,r}^{(i)}, \boldsymbol{x}\rangle \tag{12}$$

Thus

$$\begin{aligned}\psi_p(\boldsymbol{Q})\hat{\boldsymbol{W}} &= (\psi_p(\boldsymbol{Q})\hat{\boldsymbol{W}}_1, \cdots, \psi_p(\boldsymbol{Q})\hat{\boldsymbol{W}}_m) \\ &= \Big(\sum_{r=1}^{t} \lambda_r^{(1)} \prod_{j=1}^{p} \boldsymbol{Q}\boldsymbol{w}_{j,r}^{(1)}, \cdots, \sum_{r=1}^{t} \lambda_r^{(m)} \prod_{j=1}^{p} \boldsymbol{Q}\boldsymbol{w}_{j,r}^{(m)}\Big) \\ &= \sum_{r=1}^{t} \Big(\lambda_r^{(1)} \prod_{j=1}^{p} \boldsymbol{Q}\boldsymbol{w}_{j,r}^{(1)}, \cdots, \lambda_r^{(m)} \prod_{j=1}^{p} \boldsymbol{Q}\boldsymbol{w}_{j,r}^{(m)}\Big)\end{aligned} \tag{13}$$

where $\prod$ is the element-wise product.

With Eq 13, we have the following theorem to approximate the optimal projection matrix $\boldsymbol{W}$ with a set of matrices $\boldsymbol{\Theta}$.

Theorem 3.2. *Let $\boldsymbol{\Theta}_{j,r} \in \mathbb{R}^{d\times m}$ be the stack of $\boldsymbol{w}_{j,r}^{(i)}$ for $i = 1, \ldots, m$, i.e.,*

$$\boldsymbol{\Theta}_{j,r} = (\lambda_r^{(1)}\boldsymbol{w}_{j,r}^{(1)}, \cdots, \lambda_r^{(m)}\boldsymbol{w}_{j,r}^{(m)}) \tag{14}$$

where $1 \le j \le p$, $1 \le r \le t$. We have

$$\psi_p(\boldsymbol{Q})\hat{\boldsymbol{W}} = \sum_{r=1}^{t} \prod_{j=1}^{p} \boldsymbol{Q}\boldsymbol{\Theta}_{\boldsymbol{j},\boldsymbol{r}} \tag{15}$$

$$\hat{\boldsymbol{W}} = \sum_{r=1}^{t} \left(\boldsymbol{\Theta}_{\boldsymbol{1},\boldsymbol{r}} \odot_{col} \boldsymbol{\Theta}_{\boldsymbol{2},\boldsymbol{r}} \odot_{col} \cdots \odot_{col} \boldsymbol{\Theta}_{\boldsymbol{p},\boldsymbol{r}}\right) \tag{16}$$

where $\odot_{col}$ is the column-wise Kronecker product.

Similarly, $\psi_p(\boldsymbol{K})\hat{\boldsymbol{W}} = \sum_{r=1}^{t} \prod_{j=1}^{p} \boldsymbol{K}\boldsymbol{\Theta}_{\boldsymbol{j},\boldsymbol{r}}$.

More importantly, this formulation avoids explicitly constructing the exponentially large polynomial feature map $\psi_p(\boldsymbol{Q})$: we first compute the projected matrices $\boldsymbol{Q}\boldsymbol{\Theta}_{j,r}$ for all $j = 1, \ldots, p$ and $r = 1, \ldots, t$ using standard matrix multiplications. Then, for each rank component r, we combine the p projected terms via element-wise multiplication, corresponding to the product over j. Finally, we sum the resulting features across all t components. Overall, the total computation cost is reduced from $O(nmd^p)$ to $O(ndmtp)$.

In practice, we find that the setting $t = 1$ already attains an accurate approximation of the polynomial kernel. Setting $t > 1$ will lead to too many learnable parameters. Therefore, we fix $t = 1$ and our proposed feature maps is formulated as below:

Definition 3.3 (**L**earnable **Lo**w-Rank **P**olynomial **S**ketch)**.**

$$\begin{aligned}\phi_Q(\boldsymbol{Q}) &= \prod_{j=1}^{p} \boldsymbol{Q}\boldsymbol{\Theta}_{\boldsymbol{j}}^{(Q)} \\ \phi_K(\boldsymbol{K}) &= \prod_{j=1}^{p} \boldsymbol{K}\boldsymbol{\Theta}_{\boldsymbol{j}}^{(K)}\end{aligned} \tag{17}$$

where $\boldsymbol{\Theta}_{\boldsymbol{j}}^{(Q)}$ and $\boldsymbol{\Theta}_{\boldsymbol{j}}^{(Q)}$ $(1 \le j \le p)$ are learnable.

The computation complexity is $O(ndmp)$, and the number of trainable parameters are $O(dmp)$. During training, $\boldsymbol{\Theta}$ are trained implicitly to approximate the low-rank CP decomposition of the optimal projecion matrix $\boldsymbol{W}_Q^*$ and $\boldsymbol{W}_K^*$.

The following theorem analyze the error of our proposed learnable polynomial sketch.

Theorem 3.4 (LLoPS Error)**.** *Let $\boldsymbol{W}^* \in \mathbb{R}^{d^p\times m}$ be the optimal matrix suggested in Theorem 3.1. Then*

$$\min_{\boldsymbol{\Theta},\ldots,\boldsymbol{\Theta}_p} \|\boldsymbol{W}^* - \boldsymbol{\Theta}_1 \odot \ldots \odot \boldsymbol{\Theta}_p\|_F^2 = \|\boldsymbol{W}^*\|_F^2 - \sum_{i=1}^{m} \tau_i^2 \tag{18}$$

where

$$\tau_i = \max_{\|\boldsymbol{u}_1\|=\ldots=\|\boldsymbol{u}_p\|=1} \langle \boldsymbol{W}^*[:, i], \boldsymbol{u}_1 \otimes \ldots \otimes \boldsymbol{u}_p\rangle \tag{19}$$

is the tensor spectral norm (i.e., the largest tensor singular value) of $\mathcal{W}_i$, and $\mathcal{W}_i \in \mathbb{R}^{d\times\ldots\times d}$ is the order p tensor by reshaping the ith column of $\boldsymbol{W}^* \in \mathbb{R}^{d^p}$.

The proof is given in Appendix A.2.

Some previous studies found that making the attention score of linear attention to be non-negative is helpful for training stability (Kacham et al., 2023; Han et al., 2023). If non-negativity is required, we could adopt the approach used in (Kacham et al., 2023) to apply another square to the feature map, i.e., $\phi_Q(\boldsymbol{Q}) = \left(\prod_{j=1}^{p} \boldsymbol{Q}\boldsymbol{\Theta}_{\boldsymbol{j}}^{(Q)}\right)^{*2}$, and similarly for $\phi_K(\boldsymbol{K})$.

3.3. Relationship with Existing Feature Maps

In this subsection, we discuss the relationship between our proposed feature map and several existing approaches for approximating polynomial kernels in linear attention.

Element-wise Polynomial Feature Maps Some earlier linear attention methods (Han et al., 2023) adopt element-wise polynomial feature maps of the form $\phi(\boldsymbol{x}) = (\boldsymbol{\Theta}\boldsymbol{x})^{*p}$ where $*p$ denotes the element-wise p-th power. This construction can be viewed as a special case of our formulation in which all projection matrices are shared, i.e., $\boldsymbol{\Theta}_1 = \cdots = \boldsymbol{\Theta}_p$. Under this constraint, our kernel reduces exactly to an element-wise polynomial feature map.

BASED This method (Arora et al., 2024) approximates the second-order polynomial kernel by first projecting queries and keys into a low-dimensional space via a linear mapping, and then applying the quadratic feature map $\psi_2(\cdot)$. Specifically, for $\boldsymbol{W} \in \mathbb{R}^{m'\times d}$ and $\boldsymbol{x} \in \mathbb{R}^d$, BASED computes

$$\psi_2(\boldsymbol{W}\boldsymbol{x}) = \mathrm{vec}(\boldsymbol{W}\boldsymbol{x} \otimes \boldsymbol{W}\boldsymbol{x}) = (\boldsymbol{W} \otimes_{\mathrm{kron}} \boldsymbol{W})\,\mathrm{vec}(\boldsymbol{x}\boldsymbol{x}^\top), \tag{20}$$

where $\otimes_{\text{kron}}$ is the full Kronecker product.

This construction can also be recovered as a special case of our kernel. In particular, let

$$\boldsymbol{\Theta}_1^\top = \boldsymbol{W} \odot_{\text{col}} \mathbf{1}, \qquad \boldsymbol{\Theta}_2^\top = \mathbf{1} \odot_{\text{col}} \boldsymbol{W},$$

where $\mathbf{1} \in \mathbb{R}^{m' \times d}$ is an all-ones matrix. Under this choice, we have $\boldsymbol{\Theta}_1 \odot_{\text{col}} \boldsymbol{\Theta}_2 = (\boldsymbol{W} \otimes_{\text{kron}} \boldsymbol{W})^\top$. Therefore, with these specific parameter settings, our feature map exactly recovers the BASED kernel.

PolySketchFormer This approach (Kacham et al., 2023) approximates polynomial kernels using randomized sketching techniques. Its feature map can be expressed as multiplication by a randomized sketching matrix $\boldsymbol{R} \in \mathbb{R}^{d^p \times m}$ satisfying the (ϵ, p)-AMM property (Woodruff et al., 2014). Following prior constructions (Ahle et al., 2020), $\boldsymbol{R}$ can be formulated as a column-wise Kronecker product of p sparse randomized matrices, permitting efficient approximation of high-order polynomial kernels.

From our perspective, PolySketchFormer corresponds to a special case of our framework in which the projection matrices $\{\boldsymbol{\Theta}_i\}_{i=1}^p$ are fixed, randomized, and sparse, rather than learned from data.

Taken together, these observations reveal that several existing polynomial-kernel-based linear attention methods can be interpreted as special or constrained instances of our proposed feature map. By allowing more general and learnable projections, our formulation strictly generalizes these prior approaches. As a consequence, for a fixed output dimension, our method can achieve an approximation error that is no worse than these existing methods, and in practice often substantially better, as shown in the next section. This is equivalent to enforcing a constraint where half of the learnable matrices share parameters with the other half.

4. Experiments

In this section, we report the main experiment results comparing our method with state-of-the-art linear attention methods. Additional experiments and ablation studies are reported in Appendix B.

4.1. Long Range Arena Benchmark

Following prior work (Zhang et al., 2024; Meng et al., 2025; Han et al., 2023), we first assess our method on the Long Range Arena (LRA) benchmark (Tay et al., 2020). Most existing linear attention papers adopt the original LRA training protocol where models are trained from scratch on each task. However, Amos et al. (2023) pointed out that training from scratch on LRA often produces results inconsistent with pretrained transformers, and may even suggest that linear attention outperforms softmax attention, an observation that contradicts empirical evidence from large-scale language modeling and real-world deployments.

To obtain a more realistic and informative comparison, we follow the evaluation protocol of Amos et al. (2023), which introduces an additional pretraining stage with only the LRA task data. We apply this protocol uniformly to all baselines and our method, thereby ensuring a fair comparison under a setting that better reflects the behavior of pretrained transformer models. For baseline methods, we include the best number recorded in their original paper (under train-from-scratch setting) and our reproduction (under pre-training and fine-tuning setting). For all the methods, we only modify the attention layer to different attention kernels, while the architectures of other layers are identical. All hyper-parameters are selected based on their original papers. We compare our method with: Nystrom (Xiong et al., 2021), Skyformer (Chen et al., 2021), Hedgehog (Zhang et al., 2024), PolaFormer (Meng et al., 2025), Flatten Transformer (Han et al., 2023), and BASED (Arora et al., 2024), and the softmax attention (Vaswani et al., 2017). For our method, the polynomial degree $p = 4$.

The results are summarized in Table 1. First, we observe that under the pretraining protocol, the softmax Transformer consistently outperforms linear attention variants, aligning with common empirical experience and supporting the validity of this evaluation setting. Moreover, our method always produces top-3 performance among all linear attention approaches across tasks, and attains *the highest average accuracy*. On average, our method improves accuracy by 2.28% over the strongest linear attention baseline. These results demonstrate that our approach substantially improves the effectiveness of linear attention while maintaining its efficiency advantages.

4.2. Zero-Shot NLP Benchmarks

Additionally, following prior work on alternative sequence modeling architectures (Gu & Dao, 2024; Arora et al., 2024), we evaluate our method on a suite of zero-shot NLP benchmarks. We compare against the standard softmax Transformer, Mamba, and BASED, using the publicly released 360M and 1B parameter checkpoints from Arora et al. (2024). Given that these architectures require extensive pretraining from scratch, additional linear attention baselines were excluded from this evaluation.

Since the BASED attention kernel can be viewed as a special case of our proposed kernel, we initialize our model from the BASED checkpoint and continue training it to obtain our final model to reduce training time. For LLoPS-360M, the polynomial degree $p = 4$; for LLoPS-1B $p = 2$. Importantly, the original baselines in Arora et al. (2024) were pretrained on the Pile dataset (Gao et al., 2020), which is no longer publicly available. Therefore, we perform con-

Table 1. Performance comparison across Long Range Arena (LRA) benchmarks. **Bold numbers** indicate the best performance in linear attention methods, underlined numbers indicate the second-best, and *italicized numbers* indicate the third-best.

Method	Image	Text	Retrieval	ListOps	Pathfinder	Avg
Softmax Attn	64.36	82.96	83.23	48.49	70.45	69.90
Nyström	**58.66**	66.19	**83.90**	*45.46*	69.71	64.78
Skyformer	58.54	65.25	79.55	46.77	72.73	*64.57*
Hedgehog	52.68	*66.59*	82.24	41.03	**74.16**	63.34
PolaFormer	47.74	73.06	80.50	38.86	70.53	62.14
Flatten	51.48	66.13	78.72	38.51	70.27	61.02
PolySketchFormer	51.40	64.60	81.10	43.00	66.20	61.26
BASED	51.13	65.73	80.76	43.55	*72.98*	62.83
LLoPS	*57.54*	**73.94**	*82.17*	**48.29**	73.84	**67.16**

tinued training utilizing the SlimPajama-6B dataset (Soboleva et al., 2023)[4], which is 50 times smaller in size compared to Pile. For a fair comparison under this updated data setting, the BASED models are also post-trained with SlimPajama-6B for the same number of iterations. This allows us to isolate the benefits of our proposed kernel from differences arising purely from additional training data or compute. The evaluated benchmarks are implemented in `lm_eval` (Gao et al., 2023), including: WinoGrande (Sakaguchi et al., 2021), PIQA (Bisk et al., 2020), LAMBADA (Paperno et al., 2016), HellaSwag (Zellers et al., 2019), ARC-challenge (Clark et al., 2018), SQUAD (Rajpurkar et al., 2016), SWDE (Lockard et al., 2019; Arora et al., 2023), FDA (Arora et al., 2023). The tasks include information extraction and commonsense reasoning QA.

The results are reported in Table 2. Notably, for 360M models, we observe that after continued training on SlimPajama-6B, both BASED and our method exhibit a substantial accuracy drop on the FDA benchmark. We hypothesize that this degradation arises from a mismatch between the SlimPajama-6B and the FDA task distribution, which may induce catastrophic forgetting of task-relevant knowledge. Since this phenomenon occurs for both BASED and our method, it is more likely due to the choice of training data rather than a limitation of our proposed kernel. Despite this issue, our method produces a higher average accuracy than BASED. These results further support the effectiveness of our approach in improving the performance of linear attention models across diverse zero-shot NLP benchmarks.

4.3. Polynomial Kernel Approximation Error

Furthermore, we directly assess the error of our proposed sketch in approximating polynomial kernels. To that end, we obtain a set of query and key vectors with the following steps: (1) we train a transformer with the exact polynomial kernel ($Attn(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}) = (\boldsymbol{q}^\top \boldsymbol{k})^p \boldsymbol{v}$, $p = 2, 3$) on the synthetic associative recall (AR) task introduced in Arora et al. (2024); (2) we select the first (layer 0), middle (layer 8), and last attention layers (layer 16) of the trained model and sample the corresponding query and key vectors with synthetic AR data. After obtaining the query and key vectors, for each layer, we apply different sketching methods to approximate the polynomial kernel on these sampled vectors.

We compare our sketch against the data-independent sketching approach used in PolySketchFormer (Kacham et al., 2023) (PSF), as well as neural network-based sketches (NN). For the neural sketch baseline, the feature maps ϕ_q and ϕ_k are parameterized as two-layer MLPs and then trained to approximate the exact polynomial kernel. To quantify approximation quality, we report the relative mean absolute error (RMAE), defined as

$$\mathrm{RMAE} = \mathbf{E}_{\boldsymbol{q},\boldsymbol{k}} \left| \frac{(\boldsymbol{q}^\top \boldsymbol{k})^p - \phi_q(\boldsymbol{q})^\top \phi_k(\boldsymbol{k})}{(\boldsymbol{q}^\top \boldsymbol{k})^p} \right|. \tag{21}$$

The results are summarized in Table 3. We observe that our sketch consistently achieves substantially smaller approximation error than PSF. This aligns with the intuition that learnable methods are usually better than data-independent methods. Furthermore, our learnable sketch also outperforms the neural network-based feature maps. We attribute this to two main reasons. First, Theorem 3.1 suggests that a linear projection is sufficient to achieve optimal compression for polynomial kernels, making more complex nonlinear parameterizations unnecessary. Second, our structured and decomposed linear formulation is easier to optimize than MLP-based sketches, which may suffer from additional training instability. Overall, these results demonstrate that our proposed sketch provides an effective and accurate approximation of polynomial kernels in practical transformer query and key distributions.

4.4. Efficiency Comparison

Finally, we conduct a simple benchmark to evaluate the computational efficiency of our proposed linear attention kernel. We compare our method against standard softmax attention and the linear attention kernel used in BASED.

[4] https://huggingface.co/datasets/DKYoon/SlimPajama-6B

Table 2. Zero-shot accuracy (%) across diverse NLP benchmarks for pretrained models at two scales. Bold numbers indicate the best performance among linear-complexity methods within each model size.

360M Models										
Method	Wino	PIQA	LAM	Hella	ARC-E	ARC-C	FDA	SWDE	SQuAD	Avg
Softmax Attn	51.78	63.71	29.32	33.60	42.85	24.32	57.89	56.08	27.88	43.05
Mamba	50.36	63.76	27.89	33.88	42.42	24.66	5.90	17.37	24.83	32.34
BASED	50.91	65.40	27.73	36.85	43.10	**25.51**	**8.35**	30.87	26.27	35.00
LLoPS	**53.04**	**66.70**	**29.23**	**37.07**	**43.94**	**25.51**	7.53	**31.41**	**31.94**	**36.26**
1B Models										
Method	Wino	PIQA	LAM	Hella	ARC-E	ARC-C	FDA	SWDE	SQuAD	Avg
Softmax Attn	52.49	67.30	38.07	39.12	51.05	22.10	68.78	68.68	35.79	49.26
Mamba	51.30	66.81	38.87	39.34	46.80	26.11	11.07	28.17	29.46	37.55
BASED	51.22	68.44	36.43	41.59	47.01	**27.30**	24.41	46.80	36.23	42.16
LLoPS	**51.62**	**68.72**	**38.11**	**42.69**	**47.31**	27.13	**26.32**	**47.97**	**36.36**	**42.91**

Table 3. Relative mean absolute error of different sketches approximating polynomial kernels with degree $p = 2, 3$.

p	Method	Layer 0	Layer 8	Layer 15	Avg
2	PSF	0.186	0.247	0.445	0.293
	NN	0.035	0.070	0.058	0.054
	LLoPS	0.008	0.032	0.025	0.022
3	PSF	0.740	0.738	0.745	0.741
	NN	0.059	0.114	0.143	0.105
	LLoPS	0.018	0.040	0.058	0.038

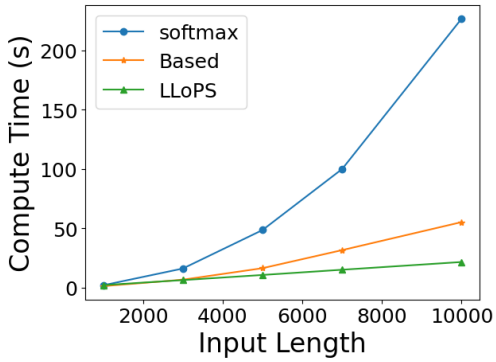


Figure 2. Time cost of different attention kernels at different input length with $p = 2$.

For each attention mechanism, we measure the runtime of the forward pass by repeating the computation 1,000 times, with sequence lengths ranging from 1,000 to 10,000 tokens. All kernels are implemented in PyTorch under the same hardware and software settings. The input vectors have a dimension of 128. We set the output feature dimension of our kernel to 256 and $p = 2$. To ensure a fair comparison, queries and keys of BASED kernel are compressed to 16 dimensions so that the final dimension of query and key vectors is $16^2 + 16 + 1 = 273$.

The total forward-pass runtime is seen in Figure 2. As expected, linear attention methods achieve substantially lower latency than softmax attention due to their reduced complexity. Moreover, our kernel is consistently faster than the BASED kernel across all sequence lengths. We attribute this improvement to the fact that our formulation relies primarily on matrix multiplications and element-wise products, which are highly optimized and parallelizable on modern accelerators (e.g. tensor cores in GPUs), whereas BASED involves additional outer-product style operations that introduce less efficient computation patterns. Overall, these results highlight that our proposed kernel not only improves approximation quality, but also offers strong practical efficiency benefits for long-sequence inference.

5. Conclusion

We studied polynomial-kernel-based linear attention through the lens of sketching. We provided a theoretical characterization of the optimal sketch for approximating polynomial kernels in the linear attention setting, revealing that structured and learnable projections can significantly improve approximation quality over previous data-independent feature constructions (Kacham et al., 2023). Building on this insight, we proposed LLoPS, a Learnable Decomposed Polynomial Sketch that offers an accurate and efficient approximation to polynomial attention while retaining the favorable computational properties of linear attention. We further analyzed approximation error for LLoPS and showed that several existing linear attention kernels can be interpreted as special cases within our framework. Empirically, LLoPS consistently improves the effectiveness of linear attention Transformers across a diverse set of benchmarks, narrowing the performance gap between linear and softmax attention while maintaining strong efficiency advantages.

References

Ahle, T. D., Kapralov, M., Knudsen, J. B., Pagh, R., Velingker, A., Woodruff, D. P., and Zandieh, A. Oblivious sketching of high-degree polynomial kernels. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 141–160. SIAM, 2020.

Amos, I., Berant, J., and Gupta, A. Never train from scratch: Fair comparison of long-sequence models requires data-driven priors. *arXiv preprint arXiv:2310.02980*, 2023.

Arora, S., Yang, B., Eyuboglu, S., Narayan, A., Hojel, A., Trummer, I., and Ré, C. Language models enable simple systems for generating structured views of heterogeneous data lakes. *arXiv preprint arXiv:2304.09433*, 2023.

Arora, S., Eyuboglu, S., Zhang, M., Timalsina, A., Alberti, S., Zinsley, D., Zou, J., Rudra, A., and Ré, C. Simple linear attention language models balance the recall-throughput tradeoff. *arXiv preprint arXiv:2402.18668*, 2024.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Chen, Y., Zeng, Q., Ji, H., and Yang, Y. Skyformer: Remodel self-attention with gaussian kernel and nystr\" om method. *Advances in Neural Information Processing Systems*, 34:2122–2135, 2021.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., Li, H., McDonell, K., Muennighoff, N., Ociepa, C., Phang, J., Reynolds, L., Schoelkopf, H., Skowron, A., Sutawika, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. A framework for few-shot language model evaluation, 12 2023. URL https://zenodo.org/records/10256836.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*, 2024.

Han, D., Pan, X., Han, Y., Song, S., and Huang, G. Flatten transformer: Vision transformer using focused linear attention. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5961–5971, 2023.

Kacham, P., Mirrokni, V., and Zhong, P. Polysketchformer: Fast transformers via sketching polynomial kernels. *arXiv preprint arXiv:2310.01655*, 2023.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Keles, F. D., Wijewardena, P. M., and Hegde, C. On the computational complexity of self-attention. In *International conference on algorithmic learning theory*, pp. 597–619. PMLR, 2023.

Lockard, C., Shiralkar, P., and Dong, X. L. Openceres: When open information extraction meets the semi-structured web. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3047–3056, 2019.

Meng, W., Luo, Y., Li, X., Jiang, D., and Zhang, Z. Polaformer: Polarity-aware linear attention for vision transformers. *arXiv preprint arXiv:2501.15061*, 2025.

Papalexakis, E. E. Automatic unsupervised tensor mining with quality assessment. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pp. 711–719. SIAM, 2016.

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, N.-Q., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 1525–1534, 2016.

Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Biderman, S., Cao, H., Cheng, X., Chung, M., Grella, M., et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

Qin, Z., Sun, W., Deng, H., Li, D., Wei, Y., Lv, B., Yan, J., Kong, L., and Zhong, Y. cosformer: Rethinking softmax in attention. *arXiv preprint arXiv:2202.08791*, 2022.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Schlag, I., Irie, K., and Schmidhuber, J. Linear transformers are secretly fast weight programmers. In *International conference on machine learning*, pp. 9355–9366. PMLR, 2021.

Soboleva, D., Al-Khateeb, F., Myers, R., Steeves, J. R., Hestness, J., and Dey, N. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama, June 2023. URL https://huggingface.co/datasets/cerebras/SlimPajama-627B.

Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., and Metzler, D. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.

Team, K., Zhang, Y., Lin, Z., Yao, X., Hu, J., Meng, F., Liu, C., Men, X., Yang, S., Li, Z., et al. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.

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

Woodruff, D. P. et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y., and Singh, V. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 14138–14148, 2021.

Yang, S., Kautz, J., and Hatamizadeh, A. Gated delta networks: Improving mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Zhang, M., Bhatia, K., Kumbong, H., and Ré, C. The hedgehog & the porcupine: Expressive linear attentions with softmax mimicry. *arXiv preprint arXiv:2402.04347*, 2024.

A. Proofs

A.1. Proof of Theorem 3.1

Proof. For simplicity, let $\tilde{\boldsymbol{Q}} = \psi_p(\boldsymbol{Q})$ and $\tilde{\boldsymbol{K}} = \psi_p(\boldsymbol{K})$. Notice that

$$\tilde{\boldsymbol{Q}}\tilde{\boldsymbol{K}}^\top = (\boldsymbol{Q}\boldsymbol{K}^\top)^{\circ p} \tag{22}$$

Let $\phi_Q(\boldsymbol{Q}) = \tilde{\boldsymbol{Q}}\boldsymbol{W}_1$, $\phi_K(\boldsymbol{K}) = \tilde{\boldsymbol{K}}\boldsymbol{W}_2$, where $\boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathbb{R}^{d^p \times m}$. We need to prove that there exist $\boldsymbol{W}_1, \boldsymbol{W}_2$ such that

$$\tilde{\boldsymbol{Q}}\boldsymbol{W}_1\boldsymbol{W}_2^\top\tilde{\boldsymbol{K}}^\top = \boldsymbol{U_m}\boldsymbol{\Sigma_m}\boldsymbol{V_m}^\top \tag{23}$$

where $\boldsymbol{S}_m = \boldsymbol{U_m}\boldsymbol{\Sigma_m}\boldsymbol{V_m}^\top$ is the truncated SVD decomposition of $\boldsymbol{S} = \tilde{\boldsymbol{Q}}\tilde{\boldsymbol{K}}^\top$.

Notice that the column and row spaces of $\boldsymbol{S}_m$ are subspaces of the column and row spaces of $\boldsymbol{S}$.

Since $\boldsymbol{S} = \tilde{\boldsymbol{Q}}\tilde{\boldsymbol{K}}^\top$, the columns of $\boldsymbol{S}$ are linear combinations of the columns of $\tilde{\boldsymbol{Q}}$. Therefore, the range (column space) of $\boldsymbol{S}$ is a subspace of the range of $\tilde{\boldsymbol{Q}}$:

$$\text{range}(\boldsymbol{S}) \subseteq \text{range}(\tilde{\boldsymbol{Q}}) \tag{24}$$

Similarly, the rows of $\boldsymbol{S}$ are linear combinations of the rows of $\tilde{\boldsymbol{K}}^\top$:

$$\text{range}(\boldsymbol{S}^\top) \subseteq \text{range}(\tilde{\boldsymbol{K}}) \tag{25}$$

As a result, $\text{range}(\boldsymbol{U_m}) \subseteq \text{range}(\tilde{\boldsymbol{Q}})$ and $\text{range}(\boldsymbol{V_m}) \subseteq \text{range}(\tilde{\boldsymbol{K}})$

Therefore, there must exist $\boldsymbol{W}_1, \boldsymbol{W}_2$ such that $\tilde{\boldsymbol{Q}}\boldsymbol{W}_1 = \boldsymbol{U_m}\boldsymbol{\Sigma_m}^{1/2}$ and $\tilde{\boldsymbol{K}}\boldsymbol{W}_2 = \boldsymbol{V_m}\boldsymbol{\Sigma_m}^{1/2}$.

So $\tilde{\boldsymbol{Q}}\boldsymbol{W}_1(\tilde{\boldsymbol{K}}\boldsymbol{W}_2)^\top = \boldsymbol{S}_m$ □

A.2. Proof of Theorem 3.4

Proof.

$$\min_{\boldsymbol{\Theta}_1,\ldots,\boldsymbol{\Theta}_p} \|\boldsymbol{W}^* - \boldsymbol{\Theta}_1 \odot \ldots \odot \boldsymbol{\Theta}_p\|_F^2 = \sum_{i=1}^{m} \min_{\boldsymbol{\Theta}_{1,i},\ldots,\boldsymbol{\Theta}_{p,i}} \|\boldsymbol{\Theta}_{1,i} \odot \ldots \odot \boldsymbol{\Theta}_{p,i} - \boldsymbol{W}^*[:,i]\|_F^2 \tag{26}$$

where $\boldsymbol{\Theta}_{j,i}$ is the i-th row of $\boldsymbol{\Theta}_j$, $1 \le j \le p$, $1 \le i \le m$.

By the definition of tensor spectral norm and the best Rank-1 CP approximation, for each i,

$$\min_{\boldsymbol{w}_{1,i},\ldots,\boldsymbol{w}_{p,i}} \|\boldsymbol{w}_{1,i} \odot \ldots \odot \boldsymbol{w}_{p,i} - \boldsymbol{W}^*[:,i]\|_F^2 = \|\boldsymbol{W}^*[:,i]\|_2^2 - \tau_i^2 \tag{27}$$

□

B. Additional Experiment Results

B.1. Synthetic Associative Recall Task

We compare our method with Based on the synthetic associative reacall task introduced in Arora et al. (2024). For both methods, the hidden dimension of the model is set to $d \in \{64, 128\}$. The query and key dimensions of BASED is 16. The sketch dimension of LLoPS $m = 256$ and $p = 4$. Table 4 shows the result. We can see that LLoPS achieves higher accuracy than BASED, further proving its effectiveness.

B.2. Polynomial Degree Ablation

Here, we study how the polynomial degree p affects the effectiveness of LLoPS. We consider the synthetic associative recall task Arora et al. (2024) and train three 2-layer transformers from scratch using polynomial kernels with degrees $p \in \{2, 4, 6\}$. All models use a hidden dimension of 128, and the output sketch dimension is fixed to 256 across settings. The results are summarized in Table 5.

Table 4. Comparion of Based and LLoPS on synthetic associative recall task. For LLoPS, $m = 256$, $p = 4$. For both methods $d = 128$.

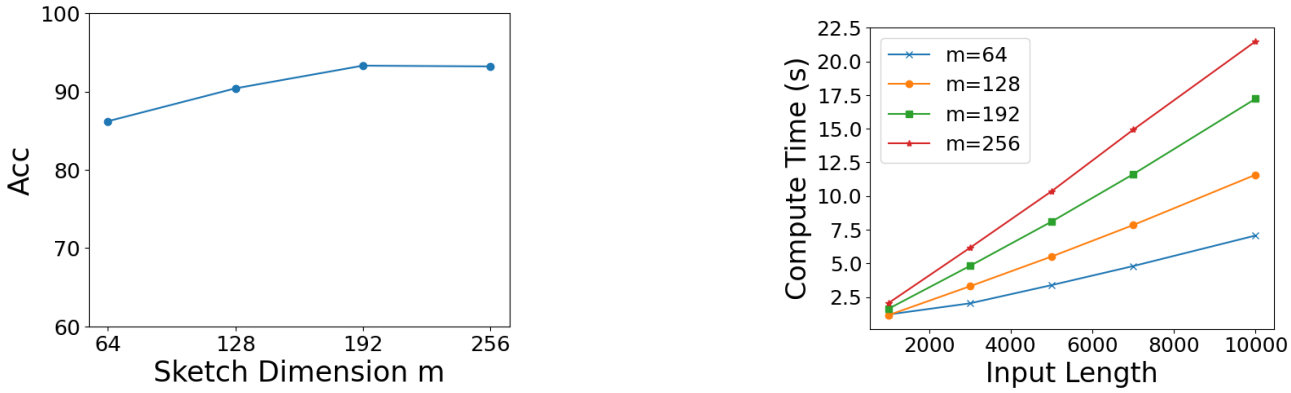
d	128	192
BASED	91.9	98.1
LLoPS	93.2	99.0

Table 5. Accuracy of synthetic association recall task under different polynomial degree p with $d = 128$.

p	2	4	6
Acc	88.7	93.2	86.6

We observe that increasing the degree from $p = 2$ to $p = 4$ leads to improved accuracy. This suggests that moderate higher-order interactions can better distinguish key vectors that align well with the queries from those that do not, thereby enhancing retrieval performance. However, when further increasing the degree to $p = 6$, accuracy declines. We hypothesize that this drop arises because the sketch dimension m remains fixed: as p grows, the polynomial kernel becomes more difficult to approximate accurately, and the resulting increase in approximation error can negatively impact model effectiveness. Another possible reason is that, since the number of parameters increase with p, a higher value of p make the model easier to overfit the training data, which would also cause the test accuracy to drop.

Overall, these results highlight an important trade-off: choosing p requires balancing representational power and approximation accuracy, and higher degree is not always better in practice.



(a) m vs Acc

(b) m vs efficency

Figure 3. Results of how accuracy and forward time changes with sketch dimension $m \in \{64, 128, 192, 256\}$ and $d = 128$, $p = 4$.

B.3. Sketch Dimension Ablation

Here we demonstrate the effect of sketch dimension m to effectiveness and effciency. We fix the model dimension $d = 128$ and vary $m \in \{64, 128, 192, 256\}$. We measure the accuracy for synthetic association recall task (Arora et al., 2024) and the forward computation time. The results show that when m increases, the effective improves but the efficiency drops, which fits the intuition. Nevertheless, we note that when the sketch dimension drops from 256 to 192, the efficiency improves while the accuracy does not drop. This suggests that our method has great potential in approximating polynomial kernels with limited budget of state space size.