The CUR Decomposition of Self-attention

Chong Wu[1], Maolin Che[1,2], and Hong Yan[1]

[1] Department of Electrical Engineering and Centre for Intelligent Multidimensional Data Analysis, City University of Hong Kong, Kowloon, Hong Kong SAR
chong@innocimda.com & h.yan@cityu.edu.hk
[2] School of Mathematics, Southwestern University of Finance and Economics, Chengdu, 611130, P. R. of China
chncml@outlook.com

Abstract. Transformers have achieved great success in natural language processing and computer vision. The core and basic technique of transformers is the self-attention mechanism. The vanilla self-attention mechanism has quadratic complexity, which limits its applications to vision tasks. Most of the existing linear self-attention mechanisms will sacrifice performance to some extent for reducing complexity. In this paper, we propose a novel linear approximation of the vanilla self-attention mechanism named CURSA to achieve both high performance and low complexity at the same time. CURSA is based on the CUR decomposition to decompose the multiplication of large matrices into the multiplication of several small matrices to achieve linear complexity. Experiment results of CURSA in image classification tasks show that it outperforms state-of-the-art self-attention mechanisms with better data efficiency, faster speed, and higher performance.

Keywords: Attention Mechanism · CUR Decomposition · Vision Transformer

1 Introduction

Transformers were first proposed to address natural language processing problems, such as machine translation [39], language inference [33], question answering [47], *etc.* Recently, transformers have also been introduced to solve computer vision tasks and have shown remarkable improvements compared to conventional convolutional neural networks (CNNs) due to their global receptive fields [14]. The core technique for capturing global information of transformers is the self-attention (SA) mechanism which is able to obtain the relationship between any two tokens. However, the quadratic complexity of the vanilla self-attention mechanism becomes a major bottleneck in training some large transformer models [44] especially for vision tasks which usually have a long sequence length generated by the tokenization of an image [24].

Linear approximations based on the kernel mechanism have been proposed to reduce the complexity of softmax normalization in the vanilla self-attention

mechanism [7, 18, 24, 27]. Some works have shown that the attention weight matrix is approximately low-rank [37, 42]. Hence, low-rank matrix decomposition methods, such as the CUR decomposition [14, 24, 44], are introduced to decompose the softmax operation for a large matrix into several softmax operations for small matrices to achieve almost linear complexity. Furthermore, due to the low-rank property of the attention weight matrix, some sparse self-attention methods have been proposed [2, 5, 19, 34, 42, 45, 49].

In this paper, we propose a novel linear self-attention mechanism, named CURSA, that has high performance and low complexity. CURSA is based on the CUR decomposition. Unlike existing linear approximation methods, which take the product of the query and key matrices as a whole for decomposition, CURSA provides a framework to decompose the query and key matrices independently to further reduce the complexity. Therefore, CURSA has lower complexity compared to existing methods using the CUR decomposition. When the sequence length is long, the complexity of CURSA is linear. We evaluated the proposed CURSA on CIFAR10/100 and ImageNet-1K. Experiment results show that CURSA outperforms the vanilla self-attention mechanism and state-of-the-art self-attention approximation methods and is faster than most state-of-the-art self-attention approximation methods. The contributions of this paper can be summarized as follows.

(1) A novel CUR decomposition framework is proposed to design the linear self-attention mechanism.
(2) An upper bound of the segment-mean sampling is given for the CUR decomposition.
(3) A fast and accurate non-iterative method is proposed for the inverse approximation.
(4) Three high-performance ViT architectures are proposed based on the proposed linear self-attention mechanism with good data efficiency and fast speed.

2 Related Work

2.1 Efficient Self-attention

The vanilla self-attention mechanism is based on the softmax operation to achieve normalized nonnegative and nonlinear attention weights, which leads to quadratic complexity with respect to sequence length. To reduce the complexity, numerous works have been proposed and they can be broadly classified into two classes: (1) Softmax based self-attention; (2) Kernel based self-attention.

Softmax Based Self-attention Softmax based efficient self-attention methods focus on reducing the complexity of softmax normalization by performing softmax normalization on small matrices. Since the attention weight matrix is approximately low-rank [37, 42], the low-rank matrix decomposition is introduced to decompose the softmax operation for the product of the query and

key matrices into several softmax operations for small matrices to reduce complexity [14, 31, 44]. [44] introduces the Nyström approximation (a special case of CUR decomposition) to decompose the softmax operation for the product of the query and key matrices into three softmax operations for small matrices, which consists of selecting column landmarks from the query and key matrices. [14] simplifies the inverse computation of [44] by using the permuted diagonal matrix to approximate the intersection matrix to further reduce the complexity. However, the permuted diagonal matrix loses most of the information of the original intersection matrix, introducing more errors and causing performance degradation. In addition, [31] performs softmax normalization independently for the query and key matrices to maintain the nonnegativity and nonlinearity of the vanilla self-attention mechanism.

In addition to the decomposition methods, sparse attention is another way to reduce the complexity. [42] introduces two linear projections to reduce the dimensions of the key and value matrices to achieve linear complexity. [49] only performs softmax normalization on the top-k contributive elements in each row of the product of the query and key matrices to reduce complexity and achieve more focused attention. [2] introduces a sliding window to achieve local focused attention for each token and adopts a task-specific global attention mechanism for some tasks that require global attention. [5] introduces a two-dimensional factorized attention mechanism, which performs local focused attention for each token, and at the same time, performs fixed step attention to obtain a larger receptive field. [34] partitions a long sequence into a set of blocks and adopts a sorting network to learn to rematch the blocks of the key matrix and the blocks of the query matrix to achieve quasi-global local attention in blockwise. [45] randomly selects several tokens from all tokens to enlarge the receptive field and uses a sliding window to select several spatially close neighbors to obtain local focused attention for each token. In addition, [45] introduces several global tokens which have effects on all tokens to maintain global information. [19] uses locality-sensitive-hashing (LSH) to obtain the most similar neighbors for each token and only performs local focused attention on the neighbor set of each token.

Kernel Based Self-attention Another way to reduce the complexity of the vanilla self-attention mechanism is based on the kernel mechanism to remove the softmax operation. [7] introduces a linear approximation of the softmax operation using positive orthogonal random features. [26] introduces random feature mapping to obtain a linear approximation of the softmax function. [43] uses ReLU dividing by the sequence length to replace softmax normalization to achieve comparable performance compared to the vanilla self-attention mechanism. [18] considers the softmax operation as a pairwise similarity between the query matrix and the key matrix and uses a decomposable kernel to replace the softmax operation to achieve linear complexity. [3] introduces a hydra trick to investigate the combination of different decomposable kernels for obtaining the pairwise similarity between the query matrix and key matrix. [24] introduces a

Gaussian kernel similarity function to replace the softmax operation of [44]. [27] performs ReLU activation on the query and key matrices to maintain the non-negative property before the computation of attention weights and introduces a linear operation with a decomposable *cos*-based nonlinear reweighting mechanism to replace the softmax operation. [4] uses ReLU linear attention to replace softmax-based attention. However, ReLU linear attention cannot generate a significant focused similarity map as softmax-based attention due to the lack of nonlinearity [4]. Hence, [4] introduces convolution to obtain a hybrid multi-scale linear attention module to enhance performance. In addition, [13] performs a focused function based on ReLU on the query and key matrices independently to preserve nonlinearity after linearization of self-attention.

Normally, most of the above-mentioned simplified approximation methods for the vanilla self-attention mechanism will sacrifice performance to some extent for reducing complexity. Therefore, a simplified approximation method is necessary, which can achieve both high performance and low complexity at the same time.

2.2 Vision Transformer

Vision transformers (ViTs) take an image as a set of overlapping/non-overlapping patches (tokens), use a linear projection layer to obtain the patch embeddings, and then combine the patch embeddings with the positional embeddings and pass them to a transformer encoder to learn the representations for each patch/token [9, 17]. The self-attention mechanism allows ViTs to capture long-range dependencies (global relations). Hence, ViTs have a larger receptive field than CNNs and have achieved remarkable success in large scale vision tasks [15]. However, the vanilla ViT lacks local inductive biases, which results in poor performance when it is trained from scratch on small and medium scale vision datasets. It shows poor data efficiency of the vanilla ViT [15, 21, 48]. Furthermore, global attention also leads to high complexity [22]. Numerous works have been proposed to improve data efficiency and reduce complexity of ViTs [22, 35, 48]. [35] introduces a CNN as a teacher network to perform knowledge distillation to train a data efficient ViT. [22] introduces non-overlapping sliding windows to divide patches into smaller blocks and performs local attention within these blocks. To capture long-range dependencies, [22] shifts the windows to capture cross-block information and performs block aggregation to reduce token numbers and aggregate information. [48] further simplifies [22] by using simple spatial operations to perform block aggregation. Unlike [22], [15] performs self-attention for each token on its surrounding neighbors and uses overlapping convolutions for the communication of cross-block information. [16] introduces a global query to obtain a global receptive field to improve the performance of [22] on large datasets.

3 Methods

3.1 Definition of Self-attention

For an input sequence $\boldsymbol{H} \in \mathbb{R}^{m\times n}$ with m tokens of embedding dimensions of n, the attention weight matrix $\boldsymbol{A} \in \mathbb{R}^{m\times m}$ of which element is the similarity between any two tokens can be calculated as follows [39],

$$\boldsymbol{A} = \text{softmax}\left(\boldsymbol{Q}\boldsymbol{K}^{\top}\right), \tag{1}$$

where $\boldsymbol{Q} \in \mathbb{R}^{m\times k}$ and $\boldsymbol{K} \in \mathbb{R}^{m\times k}$ are two same sized matrices obtained by multiplying $\boldsymbol{H}$ with two projection matrices $\mathcal{W}_Q \in \mathbb{R}^{n\times k}$ and $\mathcal{W}_K \in \mathbb{R}^{n\times k}$, respectively.

The attention weight matrix $\boldsymbol{A}$ will be used to obtain the aggregation information matrix $\boldsymbol{H}'$ as follows,

$$\boldsymbol{H}' = \boldsymbol{A}\,\boldsymbol{V}, \tag{2}$$

where $\boldsymbol{V} \in \mathbb{R}^{m\times k}$ is also obtained by multiplying $\boldsymbol{H}$ with a projection matrix $\mathcal{W}_V \in \mathbb{R}^{n\times k}$ like $\boldsymbol{Q}$ and $\boldsymbol{K}$.

3.2 The CUR Decomposition: Definition and Error Estimation

The CUR decomposition is often used in place of the low-rank approximation of the singular value decomposition (SVD) in principal component analysis. In particular, for given positive integers $r < m$ and $c < n$, the CUR decomposition of a matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ is three matrices $\boldsymbol{C} \in \mathbb{R}^{m\times c}$, $\boldsymbol{U} \in \mathbb{R}^{c\times r}$, and $\boldsymbol{R} \in \mathbb{R}^{r\times n}$, where $\boldsymbol{C}$ consists of c columns of $\boldsymbol{X}$, $\boldsymbol{R}$ consists of r rows of $\boldsymbol{X}$, and $\boldsymbol{U}$ is the Moore-Penrose inverse of $\boldsymbol{W} \in \mathbb{R}^{r\times c}$ which is the intersection part of $\boldsymbol{C}$ and $\boldsymbol{R}$. If $\|\boldsymbol{X} - \boldsymbol{CUR}\|_\xi$ is the minimum, then the product $\boldsymbol{CUR}$ closely approximates $\boldsymbol{X}$. In general, $\xi = 2$ denotes the spectral norm and $\xi = F$ denotes the Frobenius norm. Note that the CUR decomposition of $\boldsymbol{X}$ takes up $O(m \times c + n \times r)$ space.

Finding the matrices $\boldsymbol{C}$ and $\boldsymbol{R}$ is known as the column subset selection problem. The work in [12] provides the first systematic error estimation of the CUR decomposition. A sub-optimal sampling technique is considered to ensure that $\boldsymbol{U}$ has the maximal volume. One way to build a skeleton is to iteratively select good rows and columns based on the residual matrix. This is known as the cross approximation. As processing the entire residual matrix is not practical, the adaptive cross approximation (see [1]) and the incomplete cross approximation (see [36]) are two faster algorithms that operate on only a small part of the residual matrix. Another way to build a skeleton is to divide the matrix into several small segments uniformly according to the number of rows or columns to be selected and use the segment-mean to represent the skeleton [24, 44]. In this paper, we use the segment-mean to represent the skeleton because of its linear complexity. Although segment-mean is widely used in CUR based linear self-attention mechanisms [24,44], its systematic error estimation is the first time introduced in this paper to the best of our knowledge.

In detail, we assume that $\tilde{r} = p \times r$ and $\tilde{c} = l \times c$ with $\tilde{r} \leq m$ and $\tilde{c} \leq n$, where p and l are two given positive integers. Let $\widetilde{\boldsymbol{C}} = [\boldsymbol{X}(:, j_1), \ldots, \boldsymbol{X}(:, j_{\tilde{c}})]$ and $\widetilde{\boldsymbol{R}} = [\boldsymbol{X}(i_1, :), \ldots, \boldsymbol{X}(i_{\tilde{r}}, :)]$, where $1 \leq i_1 < \cdots < i_{\tilde{r}} \leq m$ and $1 \leq j_1 < \cdots < j_{\tilde{c}} \leq n$. Hence, the ith row of $\boldsymbol{R}$ and the jth column of $\boldsymbol{C}$ are given by

$$\begin{aligned} \boldsymbol{R}(i,:) &= \frac{1}{p}\sum_{s=1}^{p} \widetilde{\boldsymbol{R}}((i-1)p+s,:) = \frac{1}{p}\sum_{s=1}^{p} \boldsymbol{X}(i_{(i-1)p+s},:), \\ \boldsymbol{C}(:,j) &= \frac{1}{l}\sum_{t=1}^{l} \widetilde{\boldsymbol{C}}(:,(j-1)l+t) = \frac{1}{l}\sum_{t=1}^{l} \boldsymbol{X}(:,j_{(j-1)l+t}), \end{aligned} \tag{3}$$

with $i = 1, 2, \ldots, r$ and $j = 1, 2, \ldots, c$.

Define $\boldsymbol{S}_1 \in \mathbb{R}^{n \times \tilde{c}}$ such that its j'th column is the $j_{j'}$th column of the identity matrix in $\mathbb{R}^{n \times n}$ with $j' = 1, 2, \ldots, \tilde{c}$ and $\boldsymbol{S}_2 \in \mathbb{R}^{\tilde{r} \times m}$ such that its i'th row is the $i_{i'}$th row of the identity matrix in $\mathbb{R}^{m \times m}$ with $i' = 1, 2, \ldots, \tilde{r}$. It is easy to see that $\boldsymbol{S}_1$ has full column rank and $\boldsymbol{S}_2$ has full row rank. Hence, we have $\widetilde{\boldsymbol{C}} = \boldsymbol{X}\boldsymbol{S}_1$ and $\widetilde{\boldsymbol{R}} = \boldsymbol{S}_2\boldsymbol{X}$. Define $\boldsymbol{T}_1 \in \mathbb{R}^{\tilde{c} \times c}$ such that for each $j = 1, 2, \ldots, c$ and $i = 1 + (j-1)l, 2 + (j-1)l, \ldots, jl$, $\boldsymbol{T}_1(i,j) = 1/l$, and otherwise, $\boldsymbol{T}_1(i,j) = 0$. We also define $\boldsymbol{T}_2 \in \mathbb{R}^{r \times \tilde{r}}$ such that for each $i = 1, 2, \ldots, r$ and $j = 1 + (i-1)p, 2 + (i-1)p, \ldots, ip$, $\boldsymbol{T}_2(i,j) = 1/p$, and otherwise, $\boldsymbol{T}_2(i,j) = 0$. Hence, we have

$$\begin{aligned} \boldsymbol{C} &= \widetilde{\boldsymbol{C}}\boldsymbol{T}_1 = \boldsymbol{X}\boldsymbol{S}_1\boldsymbol{T}_1, \\ \boldsymbol{R} &= \boldsymbol{T}_2\widetilde{\boldsymbol{R}} = \boldsymbol{T}_2\boldsymbol{S}_2\boldsymbol{X}. \end{aligned} \tag{4}$$

By this way, the core matrix $\boldsymbol{U}$ is obtained by $\boldsymbol{U} = \boldsymbol{C}^{\dagger}\boldsymbol{X}\boldsymbol{R}^{\dagger}$ or $\boldsymbol{U} = (\boldsymbol{T}_2\boldsymbol{S}_2\boldsymbol{X}\boldsymbol{S}_1\boldsymbol{T}_1)^{\dagger}$.

We now consider the upper bound for $\|\boldsymbol{X} - \boldsymbol{C}\boldsymbol{U}\boldsymbol{R}\|_{\xi}$ under the case of $\boldsymbol{U} = \boldsymbol{C}^{\dagger}\boldsymbol{X}\boldsymbol{R}^{\dagger}$. By tedious multiplications, we have

$$\begin{aligned} \|\boldsymbol{X} - \boldsymbol{C}\boldsymbol{C}^{\dagger}\boldsymbol{X}\boldsymbol{R}^{\dagger}\boldsymbol{R}\|_{\xi} &= \|\boldsymbol{X} - \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\boldsymbol{X}\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}} + \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\boldsymbol{X}\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}} - \boldsymbol{C}\boldsymbol{C}^{\dagger}\boldsymbol{X}\boldsymbol{R}^{\dagger}\boldsymbol{R}\|_{\xi} \\ &\leq \|\boldsymbol{X} - \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\boldsymbol{X}\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}}\|_{\xi} + \|\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\boldsymbol{X}\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}} - \boldsymbol{C}\boldsymbol{C}^{\dagger}\boldsymbol{X}\boldsymbol{R}^{\dagger}\boldsymbol{R}\|_{\xi} \\ &\leq \|\boldsymbol{X} - \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\boldsymbol{X}\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}}\|_{\xi} + \|\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\|_2\|\boldsymbol{X}(\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}} - \boldsymbol{R}^{\dagger}\boldsymbol{R})\|_{\xi} + \\ &\quad \|(\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger} - \boldsymbol{C}\boldsymbol{C}^{\dagger})\boldsymbol{X}\|_{\xi}\|\boldsymbol{R}^{\dagger}\boldsymbol{R}\|_2 \\ &\leq \|\boldsymbol{X} - \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\boldsymbol{X}\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}}\|_{\xi} + \|\boldsymbol{X}\|_{\xi}(\|\widetilde{\boldsymbol{R}}^{\dagger}\widetilde{\boldsymbol{R}} - \boldsymbol{R}^{\dagger}\boldsymbol{R}\|_2 + \\ &\quad \|\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger} - \boldsymbol{C}\boldsymbol{C}^{\dagger}\|_2), \end{aligned} \tag{5}$$

where the first and the second inequalities hold based on the triangular inequality of matrix norms and the last inequality holds because of $\|\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger}\|_2 = 1$ and $\|\boldsymbol{R}^{\dagger}\boldsymbol{R}\|_2 = 1$. We now consider the upper bound for $\|\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger} - \boldsymbol{C}\boldsymbol{C}^{\dagger}\|_2$. By using $\boldsymbol{C} = \widetilde{\boldsymbol{C}}\boldsymbol{T}_1$ and $\widetilde{\boldsymbol{C}} = \boldsymbol{X}\boldsymbol{S}_1$, we have

$$\begin{aligned} \|\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger} - \boldsymbol{C}\boldsymbol{C}^{\dagger}\|_2 = \|\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{\dagger} - \widetilde{\boldsymbol{C}}\boldsymbol{T}_1\boldsymbol{T}_1^{\dagger}\widetilde{\boldsymbol{C}}^{\dagger}\|_2 &\leq \|\widetilde{\boldsymbol{C}}\|_2\|\widetilde{\boldsymbol{C}}^{\dagger}\|_2\|\boldsymbol{I} - \boldsymbol{T}_1\boldsymbol{T}_1^{\dagger}\|_2, \\ &\leq \|\boldsymbol{X}\|_2\|\boldsymbol{X}^{\dagger}\|_2\|\boldsymbol{S}_1\|_2\|\boldsymbol{S}_1^{\dagger}\|_2 = \|\boldsymbol{X}\|_2\|\boldsymbol{X}^{\dagger}\|_2, \end{aligned} \tag{6}$$

where the last equality holds for the fact that $\|\boldsymbol{S}_1\|_2 = \|\boldsymbol{S}_1^\dagger\|_2 = 1$. Similarly, we have

$$\|\widetilde{\boldsymbol{R}}^\dagger \widetilde{\boldsymbol{R}} - \boldsymbol{R}^\dagger \boldsymbol{R}\|_2 \leq \|\boldsymbol{X}\|_2 \|\boldsymbol{X}^\dagger\|_2. \tag{7}$$

By substituting Eq. (6) and Eq. (7) into Eq. (5), we have

$$\|\boldsymbol{X} - \boldsymbol{C}\boldsymbol{C}^\dagger \boldsymbol{X}\boldsymbol{R}^\dagger \boldsymbol{R}\|_\xi \leq \|\boldsymbol{X} - \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^\dagger \boldsymbol{X}\widetilde{\boldsymbol{R}}^\dagger \widetilde{\boldsymbol{R}}\|_\xi + 2 \cdot \mathrm{cond}(\boldsymbol{X})\|\boldsymbol{X}\|_\xi, \tag{8}$$

where $\mathrm{cond}(\boldsymbol{X}) = \|\boldsymbol{X}\|_2\|\boldsymbol{X}^\dagger\|_2 \geq 1$ is the condition number of $\boldsymbol{X}$.

Note that the term $\|\boldsymbol{X} - \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^\dagger \boldsymbol{X}\widetilde{\boldsymbol{R}}^\dagger \widetilde{\boldsymbol{R}}\|_\xi$ depends on the way of choosing two indices $\{i_1, i_2, \ldots, i_{\tilde{r}}\}$ and $\{j_1, j_2, \ldots, j_{\tilde{c}}\}$. The interested readers can refer to [6, 10,29,32] and their references for more details about the term $\|\boldsymbol{X} - \widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^\dagger \boldsymbol{X}\widetilde{\boldsymbol{R}}^\dagger \widetilde{\boldsymbol{R}}\|_\xi$. To simplify the sampling process and reduce the computation, we choose $\tilde{r} = m$ and $\tilde{c} = n$, then the upper bound of error (Eq. (8)) becomes

$$\|\boldsymbol{X} - \boldsymbol{C}\boldsymbol{C}^\dagger \boldsymbol{X}\boldsymbol{R}^\dagger \boldsymbol{R}\|_\xi \leq 2 \cdot \mathrm{cond}(\boldsymbol{X})\|\boldsymbol{X}\|_\xi. \tag{9}$$

More complex and adaptive strategies can be used to generate $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$ to further reduce the coefficient of the above upper bound (see Eq. (6)), but they will increase computational complexity. Fig. 1 shows an example of segment-mean sampling.

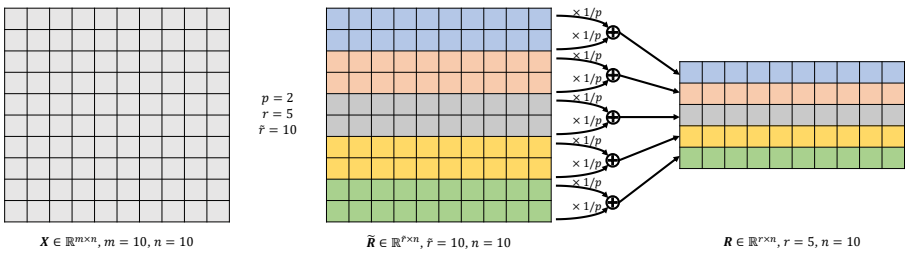

Fig. 1: A toy example of segment-mean sampling.

3.3 The CUR Decomposition of Self-attention

In this paper, we introduce the CUR decomposition to design a linear approximation of self-attention named CURSA as follows,

$$\boldsymbol{H}' \approx \mathrm{softmax}\left(\boldsymbol{C}_Q\right) \boldsymbol{U}_{\mathrm{softmax}(\boldsymbol{W}_Q)} \mathrm{softmax}\left(\boldsymbol{R}_Q \boldsymbol{K}^\top\right) \boldsymbol{V}. \tag{10}$$

According to [31], for two matrices $\boldsymbol{E}$ and $\boldsymbol{F}$, the softmax operation of their product can be approximated as follows,

$$\mathrm{softmax}\left(\boldsymbol{E}\boldsymbol{F}\right) \approx \mathrm{softmax}\left(\boldsymbol{E}\right) \mathrm{softmax}\left(\boldsymbol{F}\right). \tag{11}$$

Hence, Eq. (10) can be obtained as follows,

$$\begin{aligned}\boldsymbol{H}' &= \text{softmax}\left(\boldsymbol{Q}\boldsymbol{K}^\top\right)\boldsymbol{V}\\ &\approx \text{softmax}\left(\boldsymbol{Q}\right)\text{softmax}\left(\boldsymbol{K}^\top\right)\boldsymbol{V}\\ &\approx \text{softmax}\left(\boldsymbol{C}_Q\right)\boldsymbol{U}_{\text{softmax}(\boldsymbol{W}_Q)}\text{softmax}\left(\boldsymbol{R}_Q\right)\text{softmax}\left(\boldsymbol{K}^\top\right)\boldsymbol{V}\\ &\approx \text{softmax}\left(\boldsymbol{C}_Q\right)\boldsymbol{U}_{\text{softmax}(\boldsymbol{W}_Q)}\text{softmax}\left(\boldsymbol{R}_Q\boldsymbol{K}^\top\right)\boldsymbol{V}.\end{aligned} \tag{12}$$

With an increasing number of rows and columns sampled, $\text{softmax}\left(\boldsymbol{C}_Q\right)\boldsymbol{U}_{\text{softmax}(\boldsymbol{W}_Q)}$ will finally equal to an identity matrix and $\text{softmax}\left(\boldsymbol{R}_Q\boldsymbol{K}^\top\right)$ will equal to $\text{softmax}\left(\boldsymbol{Q}\boldsymbol{K}^\top\right)$ exactly. Then Eq. (10) justly equals to Eq. (2) which promises the approximation. $\boldsymbol{H}' \approx \text{softmax}\left(\boldsymbol{Q}\right)\text{softmax}\left(\boldsymbol{K}^\top\right)\boldsymbol{V}$ is the design of efficient attention (EFFATT) [31]. It has a problem of concentration reduction of attention maps introduced by performing softmax normalization on the query matrix and key matrix independently. Our design reduces this phenomenon by preserving higher nonlinearity. Most of existing CUR decomposition based methods are based on the following formula [14, 24, 44],

$$\boldsymbol{H}' \approx \text{sim}\left(\boldsymbol{Q}\left(\boldsymbol{R}_K\right)^\top\right)\boldsymbol{U}_{\text{sim}\left(\boldsymbol{R}_Q\left(\boldsymbol{R}_K\right)^\top\right)}\text{sim}\left(\boldsymbol{R}_Q\boldsymbol{K}^\top\right)\boldsymbol{V}, \tag{13}$$

where $\text{sim}(\cdot)$ can be $\text{softmax}(\cdot)$ or other element-wise functions. Compared to the proposed CURSA in Eq. (10), Eq. (13) has higher complexity because it takes $\text{sim}\left(\boldsymbol{Q}\boldsymbol{K}^\top\right)$ as a whole for decomposition.

3.4 Approximation of Moore-Penrose Inverse of the Intersection Matrix

The Moore-Penrose inverse matrix $\boldsymbol{U}$ of the intersection matrix $\boldsymbol{W}$ can be obtained by performing the SVD on $\boldsymbol{W}$. However, the cubic complexity of the SVD is not suitable for CURSA, which is required to compute a large number of Moore-Penrose inverse matrices. We introduce a fast iterative Moore-Penrose inverse approximation method [28, 44] to obtain $\boldsymbol{U}$ as follows,

$$\begin{aligned}\boldsymbol{U}_{i+1} &= \frac{13}{4}\boldsymbol{U}_i - \frac{15}{4}\boldsymbol{U}_i\boldsymbol{W}\boldsymbol{U}_i + \frac{7}{4}\boldsymbol{U}_i\boldsymbol{W}\boldsymbol{U}_i\boldsymbol{W}\boldsymbol{U}_i - \frac{1}{4}\boldsymbol{U}_i\boldsymbol{W}\boldsymbol{U}_i\boldsymbol{W}\boldsymbol{U}_i\boldsymbol{W}\boldsymbol{U}_i\\ &= \frac{13}{4}\boldsymbol{U}_i - \left(\frac{15}{4}\boldsymbol{U}_i - \left(\frac{7}{4}\boldsymbol{U}_i - \frac{1}{4}\boldsymbol{U}_i\boldsymbol{W}\boldsymbol{U}_i\right)\boldsymbol{W}\boldsymbol{U}_i\right)\boldsymbol{W}\boldsymbol{U}_i\\ &= \frac{1}{4}\boldsymbol{U}_i\left(13\boldsymbol{I} - \left(15\boldsymbol{I} - \left(7\boldsymbol{I} - \boldsymbol{W}\boldsymbol{U}_i\right)\boldsymbol{W}\boldsymbol{U}_i\right)\boldsymbol{W}\boldsymbol{U}_i\right),\ \text{if } c \geq r,\\ \boldsymbol{U}_{i+1} &= \frac{1}{4}\left(13\boldsymbol{I} - \boldsymbol{U}_i\boldsymbol{W}\left(15\boldsymbol{I} - \boldsymbol{U}_i\boldsymbol{W}\left(7\boldsymbol{I} - \boldsymbol{U}_i\boldsymbol{W}\right)\right)\right)\boldsymbol{U}_i,\ \text{if } c < r.\end{aligned} \tag{14}$$

If $\boldsymbol{U}_0$ is initialized according to $\|\boldsymbol{W}\boldsymbol{W}^\dagger - \boldsymbol{W}\boldsymbol{U}_0\| < 1$, then Eq. (14) converges to $\boldsymbol{W}^\dagger$ in the third-order. As suggested by [44], Eq. (14) usually requires at least 6 iterations to converge.

An alternative iterative inverse approximation method [28] exists which has a faster convergence rate, as shown below,

$$\begin{aligned}
\Phi &= -11\boldsymbol{I} + \boldsymbol{W}\boldsymbol{U}_i\left(25\boldsymbol{I} + \boldsymbol{W}\boldsymbol{U}_i\left(-30\boldsymbol{I} + \boldsymbol{W}\boldsymbol{U}_i\left(20\boldsymbol{I} + \boldsymbol{W}\boldsymbol{U}_i\left(-7\boldsymbol{I} + \boldsymbol{W}\boldsymbol{U}_i\right)\right)\right)\right)\\
\boldsymbol{U}_{i+1} &= -\frac{1}{4}\boldsymbol{U}_i\Phi\left(4\boldsymbol{I} + \boldsymbol{W}\boldsymbol{U}_i\Phi\right), \text{ if } c \geq r,\\
\Phi &= -11\boldsymbol{I} + \boldsymbol{U}_i\boldsymbol{W}(25\boldsymbol{I} + \boldsymbol{U}_i\boldsymbol{W}(-30\boldsymbol{I} + \boldsymbol{U}_i\boldsymbol{W}(20\boldsymbol{I} + \boldsymbol{U}_i\boldsymbol{W}(-7\boldsymbol{I} + \boldsymbol{U}_i\boldsymbol{W}))))\\
\boldsymbol{U}_{i+1} &= -\frac{1}{4}\Phi\left(4\boldsymbol{I} + \boldsymbol{U}_i\boldsymbol{W}\Phi\right)\boldsymbol{U}_i, \text{ if } c < r.
\end{aligned} \tag{15}$$

Eq. (15) can converge to $\boldsymbol{W}^\dagger$ in the tenth-order. To obtain the Moore-Penrose inverse rapidly, we combine Eq. (14) with Eq. (15) together to propose a non-iterative method. In detail, we first execute Eq. (15) for 1 iteration to converge quickly and then execute Eq. (14) for 1 more iteration to obtain the accurate inverse. We will show that the non-iterative inverse approximation method based on Eq. (14) and Eq. (15) together has non-inferior performance compared to Eq. (14) using 6 iterations in Sec. 4.2.

3.5 Complexity Analysis

The computational complexity of $\boldsymbol{Q}\boldsymbol{K}^\top$ in Eq. (1) is $\mathrm{O}(m^2 \times k)$ and the computational complexity of the softmax operation in vanilla self-attention is $\mathrm{O}(m^2)$. Hence, the computational complexity of vanilla self-attention (Eq. (2)) is $\mathrm{O}(m^2 \times k)$. The computational complexity of softmax $\left(\boldsymbol{R}_Q\boldsymbol{K}^\top\right)\boldsymbol{V}$ in Eq. (10) is $\mathrm{O}(m \times k \times r_Q)$. Assuming that $c_Q \geq r_Q$, for Eq. (14), its complexity is $\mathrm{O}(2 \times r_Q^2 \times c_Q + 2 \times r_Q^3)$ for one iteration. As suggested by [44], Eq. (14) requires y ($y = 6 \ll m$ or k) iterations to converge, taking $\mathrm{O}(2 \times y \times (r_Q^2 \times c_Q + r_Q^3))$. Furthermore, for the non-iterative inverse approximation method based on the combination of Eq. (14) and Eq. (15), its complexity is $\mathrm{O}(4 \times r_Q^2 \times c_Q + 8 \times r_Q^3)$ which is much smaller than the iterative method based on Eq. (14) only. Then the total computational complexity of Eq. (10) is $\mathrm{O}(m \times k \times r_Q)$. For the case where $c_Q < r_Q$, the complexity of Eq. (14) is $\mathrm{O}(2 \times c_Q^2 \times r_Q + 2 \times c_Q^3)$ for one iteration. Hence, the complexity of Eq. (14) using y iterations is $\mathrm{O}(2 \times y \times (c_Q^2 \times r_Q + c_Q^3))$ and the complexity of the non-iterative inverse approximation method based on the combination of Eq. (14) and Eq. (15) is $\mathrm{O}(4 \times c_Q^2 \times r_Q + 8 \times c_Q^3)$. Both $\mathrm{O}(2 \times y \times (c_Q^2 \times r_Q + c_Q^3))$ and $\mathrm{O}(4 \times c_Q^2 \times r_Q + 8 \times c_Q^3)$ are smaller than $\mathrm{O}(m \times k \times r_Q)$. Hence, the total computational complexity of CURSA is still $\mathrm{O}(m \times k \times r_Q)$. In general, m or k is much larger than r_Q. Hence, the total computational complexity of Eq. (10) is $\mathrm{O}(m \times k)$. If $k \ll m$, the above computational complexity will be reduced to linear complexity $\mathrm{O}(m)$.

4 Experiments and Results

4.1 Experiment Settings

CIFAR10 [20] and CIFAR100 [20] were selected to evaluate the data efficiency of the proposed method. ImageNet-1K [30] was selected to evaluate the performance of the proposed method on the large scale dataset. We adopted CSWin [8, 13], FLatten [13], and NesT [48] as the backbone network architectures. The optimizer for CURSA is AdamW [23] with an initial learning rate of $\frac{B}{128} \times 1.25 \times 10^{-4}$ and weight decay of 0.05, where B is the batch size. The learning rate schedule of CURSA is the cosine decay scheduler that uses 5 epochs (CIFAR10/100) and 20 epochs (ImageNet-1K) for warm-up. The stochastic depth drop rate for CIFAR10/100 was set to 0.1 and for ImageNet-1K to 0.2 (CSWin and FLatten backbones) and 0.3 (NesT backbone). We compared our method with state-of-the-art ViTs: EfficientViT [4], CSWin [8], ViT [9], GC-ViT [16], Swin [22], FastViT [38], and NesT [48]. We adopted the commonly used data augmentation methods [11, 35] without random repeated augmentation to preprocess CIFAR10/100 and ImageNet-1K for all methods. We re-trained these methods from the scratch with using the same data augmentation methods to maintain the fairness. Furthermore, to investigate the effectiveness of the proposed method, we compared it with the vanilla self-attention mechanism (ViT [9]), decomposition based self-attention mechanism (EFFATT [31] and Nyströmformer [44]), kernel based self-attention mechanism (FLatten [13], SOFT [24], cosFormer [27], and SOFT-Norm [46]), and sparse attention mechanism (CSWin [8, 13] and NesT [48]). The experiments of ImageNet-1K were carried out on 3 NVIDIA Tesla A100 (40GB) GPUs and the experiments of CIFAR10/100 were carried out on 1 NVIDIA Tesla A100 (40GB) GPU. The latency and inference throughput comparison experiments were carried out on 1 NVIDIA GeForce RTX 3080 Laptop (16GB) GPU.

4.2 Experiment Results

Comparison of Image Classification Performance on Small Datasets Tab. 1 shows the comparison of the proposed method and state-of-the-art ViTs on CIFAR10/100. CURSA outperforms three backbones and other ViTs significantly. Some advanced ViTs such as EfficientViT, GC-ViT, and FastViT perform much worse than CURSA on these small datasets, further showing the good data efficiency of CURSA. Nyströmformer improves the performance of NesT but is still worse than CURSA. The performance of EFFATT is slightly inferior to that of NesT, which shows that the reduction in concentration of attention maps introduced by performing softmax normalization on the query matrix and the key matrix independently will cause performance degradation. CURSA preserves softmax normalization for some parts of the product of query and key matrices to reduce this phenomenon. cosFormer performs much worse than NesT on CIFAR10 and its *cos*-based re-weighting mechanism shows high sensitive to the normalization threshold to avoid gradient explosion in training.

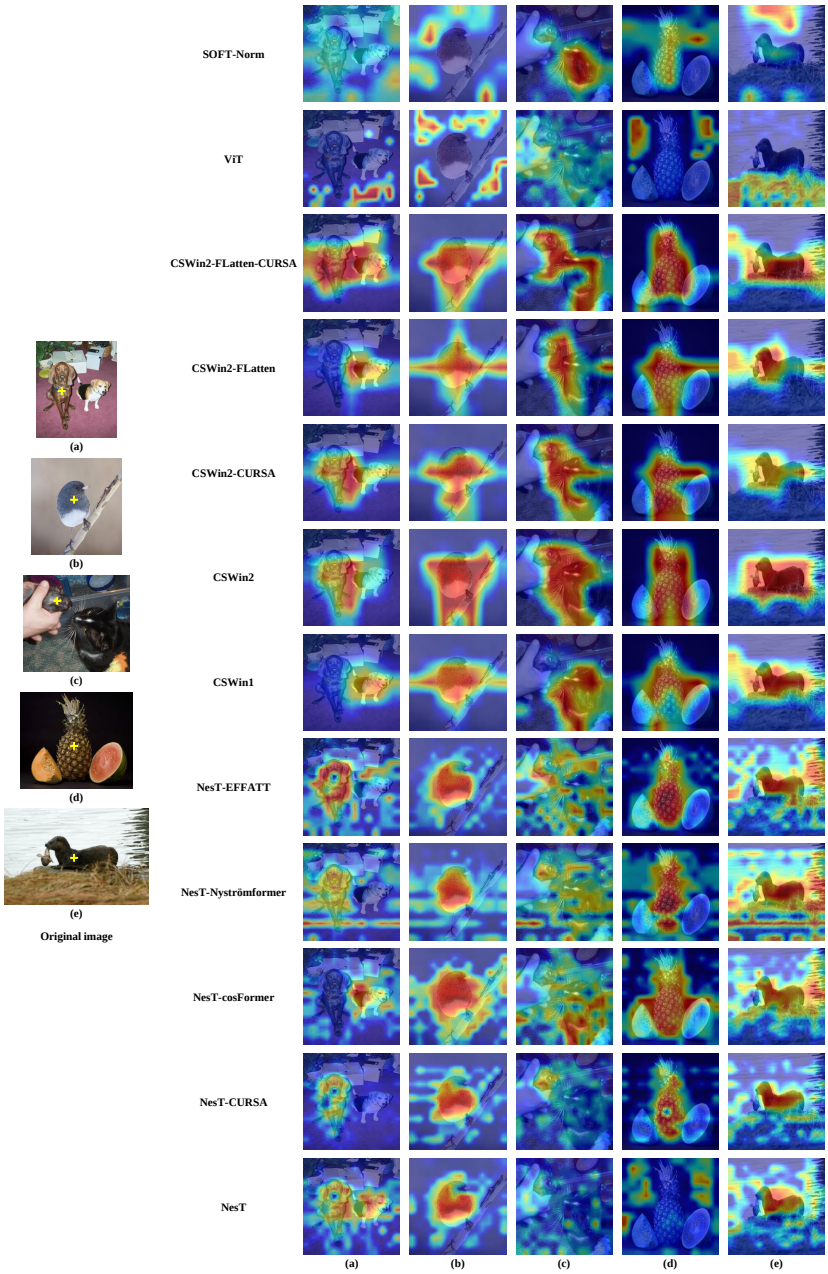

Fig. 2: Visualization of class activation map (CAM) based attention results of different attention mechanisms. All CAM based attention results were generated based on Score-CAM [41]. Note: CSWin1 denotes CSWin-T12211 and CSWin2 denotes CSWin-T24181.

SOFT performs much worse than NesT on CIFAR10/100, which shows that $\boldsymbol{Q} = \boldsymbol{K}$ pays great attention to each token itself and reduces the effect of other tokens. Some papers have shown that most of the attention map patterns are usually not diagonal [25, 40]. $\boldsymbol{Q} = \boldsymbol{K}$ reduces the effect of these non-diagonal patterns that can result in performance degradation. CSWin does not have an architecture for small resolution images, hence we referred to Swin to design CSWin-T2262, which uses a depth of 2-2-6-2 and a stripe width of 1-2-4-4. In addition, we further designed an improved version called CSWin-T1281, which uses a depth of 1-2-8-1 and a stripe width of 1-2-4-4. CURSA improves NesT, CSWin, and FLatten by 0.5%, 0.6%, and 0.1% on CIFAR10 and 1.0%, 1.5%, and 1.2% on CIFAR100, respectively. Furthermore, as Tab. 1 shows, CURSA does not affect the number of parameters of the backbones.

Table 1: Comparison of top-1 test accuracy and parameter numbers (#) of all methods on CIFAR10/100. Note: The best values are in bold and the second best values are underlined.

Methods	Res.	CIFAR10 Acc. (%)	CIFAR100 Acc. (%)	#
NesT_4-T-CURSA ($r_Q = \frac{1}{4}m$)	32	<u>96.3</u>	80.5	6M
NesT_4-T	32	95.8	79.5	6M
NesT_4-T-FLatten	32	95.9	79.9	6M
NesT_4-T-Nyströmformer ($r_Q = m$)	32	95.9	79.8	6M
NesT_4-T-Nyströmformer ($r_Q = \frac{5}{8}m$)	32	96.0	79.5	6M
NesT_4-T-Nyströmformer ($r_Q = \frac{3}{8}m$)	32	95.7	79.6	6M
NesT_4-T-Nyströmformer ($r_Q = \frac{1}{4}m$)	32	95.8	79.1	6M
NesT_4-T-Nyströmformer ($r_Q = \frac{1}{8}m$)	32	95.7	79.2	6M
NesT_4-T-EFFATT	32	95.8	79.2	6M
NesT_4-T-cosFormer	32	91.2	79.4	6M
CSWin-T1281-CURSA ($r_Q = \frac{1}{4}m, c_Q = \frac{1}{6}k$)	32	**96.4**	<u>81.4</u>	11M
CSWin-T1281-FLatten-CURSA ($r_Q = \frac{1}{4}m$)	32	**96.4**	**81.7**	11M
CSWin-T1281-FLatten	32	<u>96.3</u>	80.5	11M
CSWin-T2262	32	95.7	78.7	13M
CSWin-T1281	32	95.8	79.9	11M
SOFT-Tiny	32	89.1	64.0	12M
Swin-T	32	95.2	77.1	27M
GC-ViT-XXT	32	89.7	67.7	11M
FastViT-SA12	32	84.3	59.0	10M
EfficientViT-B0	32	78.6	46.6	2M

Comparison of Image Classification Performance of Different Attention Mechanisms Tab. 2 shows the comparison of top-1 test accuracy, latency of self-attention (SA), and inference throughput (IT) of different attention mechanisms on ImageNet-1K. FLatten uses a modified CSWin-T architecture named CSWin-T24181, which uses a depth of 2-4-18-1, as the backbone. To maintain fairness, we also implemented our CURSA using CSWin-T24181 as the backbone. We also provided the results obtained by the official CSWin-T architecture named CSWin-T12211, which uses a depth of 1-2-21-1. CURSA achieves the highest accuracy and significantly improves the speed of the backbones. It should

Table 2: Comparison of top-1 test accuracy, multiply–accumulate operations (MACs), parameter numbers (#), latency of self-attention (SA), and inference throughput (IT) of different attention mechanisms on ImageNet-1K. Note: SA and IT of all methods except CSWin2-FLatten were obtained in mixed precision because CSWin-T24181-FLatten is incompatible with FP16. CSWin1 denotes CSWin-T12211 and CSWin2 denotes CSWin-T24181. × denotes the rate of improvement. The best values are in bold and the second best values are underlined.

Methods	Res.	Acc. (%)	SA (ms/img)	×	IT (imgs/s)	×	#	MACs
$NesT_4$-S-CURSA ($r_Q = \frac{1}{7}m$)	224	**83.4**	4.77e-2	3.6	438	1.5	38M	9G
$NesT_4$-S-Nyströmformer ($r_Q = \frac{1}{7}m$)	224	81.9	5.46e-2	3.1	389	1.3	38M	9G
$NesT_4$-S-EFFATT	224	81.4	3.66e-2	4.7	525	1.7	38M	9G
$NesT_4$-S-cosFormer	224	81.0	4.09e-2	4.2	470	1.6	38M	9G
$NesT_4$-S	224	<u>83.3</u>	5.56e-2	3.1	381	1.3	38M	9G
CSWin2-CURSA ($r_Q = \frac{1}{4}m$)	224	82.9	<u>3.35e-2</u>	<u>5.1</u>	<u>663</u>	<u>2.2</u>	20M	4G
CSWin2-FLatten-CURSA ($r_Q = \frac{1}{4}m$)	224	83.1	4.27e-2	4.0	565	1.9	20M	4G
CSWin2-FLatten	224	83.1	7.01e-2	2.4	316	1.0	21M	4G
CSWin2	224	82.4	7.39e-2	2.3	363	1.2	20M	4G
CSWin1	224	82.7	**3.03e-2**	**5.6**	**671**	**2.2**	22M	4G
SOFT-Norm-Small	224	82.4	17.06e-2	1.0	301	1.0	24M	3G
SOFT-Small	224	82.2	13.25e-2	1.3	347	1.2	24M	3G
ViT-B/16	224	77.3	7.64e-2	2.2	483	1.6	86M	16G

be noted that CURSA offers 2.2 times speed-up of SA and 1.8 times speed-up of IT over CSWin-T24181. Furthermore, CURSA offers 2.3 times speed-up of SA and 1.4 times speed-up of IT over ViT-B/16. Although cosFormer is slightly faster than CURSA when using the NesT backbone, CURSA significantly outperforms cosFormer in terms of accuracy. Another CUR decomposition based method, Nyströmformer, only improves the speed of the backbone slightly, and its performance is much inferior to ours and that of the backbone. The improved version of Nyströmformer, SOFT, is still inferior to CURSA and is significantly slower than CURSA. EFFATT improves the speed of the backbone (NesT) much more than other self-attention mechanisms. However, its performance is significantly inferior to ours and that of NesT. CURSA offers 2.1 times speed-up of SA and 2.2 times speed-up of IT over FLatten when using CSWin-T24181 as the backbone. Although FLatten introduces a large number of normalization operations to avoid gradient explosion, it still meets the gradient explosion problem in mixed precision. FLatten using FP32 precision has even faster SA speed than CSWin-T24181 using mixed precision. However, the slow FP32 operations make its overall speed (IT) slower than that of CSWin-T24181. To make FLatten compatible with FP16, we replaced focused linear attention with CURSA, and the proposed FLatten-CURSA offers 1.7 times speed-up of SA and 1.9 times speed-up of IT over FLatten. Also, as Tab. 2 shows, CURSA does not affect the number of multiply-accumulate operations (MACs) and parameters of the backbones. Furthermore, as Tab. 1 shows, CURSA also outperforms other attention mechanisms on small datasets. Fig. 2 shows the visualization comparison of the class activation map (CAM) based attention results of different attention mecha-

nisms. CURSA helps the backbones generate a more focused attention map and reduce misclassification.

Table 3: Comparison of top-1 test accuracy and parameter numbers (#) of CURSA using different r_Q and c_Q on CIFAR10/100. Note: The best values are in bold face.

Methods	Res.	CIFAR10 Acc. (%)	CIFAR100 Acc. (%)	#
NesT_4-T-CURSA ($r_Q = m$)	32	**96.3**	80.0	6M
NesT_4-T-CURSA ($r_Q = \frac{5}{8}m$)	32	**96.3**	80.1	6M
NesT_4-T-CURSA ($r_Q = \frac{1}{2}m$)	32	**96.3**	80.0	6M
NesT_4-T-CURSA ($r_Q = \frac{3}{8}m$)	32	**96.3**	80.4	6M
NesT_4-T-CURSA ($r_Q = \frac{1}{4}m$)	32	**96.3**	**80.5**	6M
NesT_4-T-CURSA ($r_Q = \frac{1}{8}m$)	32	96.2	80.1	6M
CSWin-T1281-CURSA ($r_Q = \frac{1}{4}m$)	32	**96.4**	81.1	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{4}m, c_Q = \frac{1}{6}k$)	32	**96.4**	**81.4**	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{6}m$)	32	**96.4**	81.0	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{6}m, c_Q = \frac{1}{4}k$)	32	**96.4**	81.0	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{6}m, c_Q = \frac{1}{6}k$)	32	**96.4**	80.9	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{6}m, c_Q = \frac{1}{8}k$)	32	96.2	80.9	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{6}m, c_Q = \frac{1}{10}k$)	32	96.2	80.8	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{8}m$)	32	**96.4**	81.0	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{8}m, c_Q = \frac{1}{6}k$)	32	**96.4**	80.9	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{10}m$)	32	96.3	81.0	11M
CSWin-T1281-CURSA ($r_Q = \frac{1}{10}m, c_Q = \frac{1}{6}k$)	32	96.3	81.3	11M
CSWin-T1281-FLatten-CURSA ($r_Q = \frac{1}{4}m$)	32	**96.4**	**81.7**	11M
CSWin-T1281-FLatten-CURSA ($r_Q = \frac{1}{6}m$)	32	**96.4**	81.6	11M
CSWin-T1281-FLatten-CURSA ($r_Q = \frac{1}{8}m$)	32	**96.4**	81.5	11M
CSWin-T1281-FLatten-CURSA ($r_Q = \frac{1}{10}m$)	32	**96.4**	81.5	11M

Table 4: Comparison of top-1 test accuracy and parameter numbers (#) of CURSA using different r_Q and c_Q on ImageNet-1K. Note: The best values are in bold face.

Methods	Res.	ImageNet-1K Acc. (%)	#
CSWin-T24181-CURSA ($r_Q = \frac{1}{4}m$)	224	**82.9**	20M
CSWin-T24181-CURSA ($r_Q = \frac{1}{8}m$)	224	82.6	20M
CSWin-T24181-CURSA ($r_Q = \frac{1}{8}m, c_Q = \frac{1}{8}k$)	224	82.4	20M

Effect of r_Q and c_Q Tab. 3 and Tab. 4 show the comparison of CURSA using different r_Q and c_Q on CIFAR10/100 and ImageNet-1K. With a decreasing number of r_Q, the performance of CURSA using NesT as the backbone decreases slightly on CIFAR10, while its performance on CIFAR100 improves first and then decreases. Compared to the backbone (NesT) and all other simplified versions of NesT in Tab. 1, all NesT_4-T-CURSA variants in Tab. 3 significantly outperform them, which shows the good data efficiency of CURSA. Furthermore, as Tab. 3 shows, with a decreasing number of r_Q or c_Q, the performance of CURSA using CSWin as the backbone on CIFAR10/100 is in a downward trend except

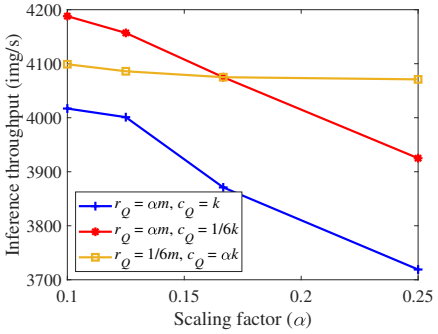

Fig. 3: Comparison of inference throughput of CSWin-T1281-CURSA using different r_Q and c_Q on CIFAR10/100.

when $r_Q = \frac{1}{4}m, c_Q = \frac{1}{6}k$ and $r_Q = \frac{1}{10}m, c_Q = \frac{1}{6}k$. Furthermore, its performance decreases slightly on ImageNet-1K with a decreasing number of r_Q or c_Q as Tab. 4 shows. The performance of FLatten-CURSA also decreases slightly on CIFAR10/100 with a decreasing number of r_Q. The general trend of performance of CURSA is decreasing with a decreasing number of r_Q or c_Q. Fig. 3 shows the comparison of inference throughput of CSWin-T1281-CURSA using different r_Q and c_Q on CIFAR10/100. With a decreasing number of r_Q, the inference throughput increases significantly. When c_Q is reduced, the improvement of inference throughput is not significant. According to the analysis in Sec. 3.5, the total computational complexity of CURSA depends on r_Q. Hence, r_Q has a larger effect on complexity than c_Q. Hence, r_Q and c_Q can be determined according to the requirements of performance and efficiency.

Effect of Different Inverse Approximation Methods Tab. 5 shows the comparison of the top-1 test accuracy of CURSA using different inverse approximation methods. We compared the original Eq. (14) using 6 iterations as suggested by [44] with two non-iterative inverse approximation methods that we propose in this paper, which are based on the combination of Eq. (14) and Eq. (15), each using a single iteration. The difference between two non-iterative inverse approximation methods lies in the execution order of Eq. (14) and Eq. (15). Both non-iterative inverse approximation methods demonstrate comparable or superior performance to Eq. (14) with 6 iterations. There is no significant difference between the performance of two non-iterative inverse approximation methods. Two non-iterative inverse approximation methods have faster speed than Eq. (14) using 6 iterations.

5 Conclusion

In this paper, we proposed a novel linear self-attention mechanism, named CURSA, for ViTs. The proposed self-attention mechanism can significantly reduce the

Table 5: Comparison of top-1 test accuracy, parameter numbers (#), and inference throughput (IT) of CURSA using different inverse approximation methods. Note: The best values are in bold face.

Methods	Res.	CIFAR10 Acc. (%)	CIFAR100 Acc. (%)	#	IT (imgs/s)
NesT$_4$-T-CURSA ($r_Q = \frac{1}{4}m$)					
Eq. (14) 6 iterations	32	**96.3**	**80.5**	6M	1567
Eq. (15) & Eq. (14)	32	**96.3**	**80.5**	6M	1688
Eq. (14) & Eq. (15)	32	**96.3**	80.2	6M	1688
CSWin-T1281-CURSA ($r_Q = \frac{1}{4}m, c_Q = \frac{1}{6}k$)					
Eq. (14) 6 iterations	32	**96.4**	81.4	11M	3925
Eq. (15) & Eq. (14)	32	**96.4**	81.7	11M	3983
Eq. (14) & Eq. (15)	32	96.3	**81.8**	11M	3983
CSWin-T1281-FLatten-CURSA ($r_Q = \frac{1}{4}m$)					
Eq. (14) 6 iterations	32	96.4	81.7	11M	3384
Eq. (15) & Eq. (14)	32	**96.6**	81.7	11M	3515
Eq. (14) & Eq. (15)	32	**96.6**	**81.9**	11M	3515

complexity of self-attention in ViTs. It achieves significantly better performance compared to state-of-the-art attention mechanisms. In the future, we will investigate the combination of CURSA with more ViT architectures for more vision tasks.

References

1. Bebendorf, M.: Approximation of boundary element matrices. Numerische Mathematik **86**, 565–589 (2000)
2. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The long-document transformer. CoRR **abs/2004.05150** (2020), https://arxiv.org/abs/2004.05150
3. Bolya, D., Fu, C.Y., Dai, X., Zhang, P., Hoffman, J.: Hydra attention: Efficient attention with many heads. In: Karlinsky, L., Michaeli, T., Nishino, K. (eds.) Computer Vision – ECCV 2022 Workshops. pp. 35–49. Springer Nature Switzerland, Cham (2023)
4. Cai, H., Li, J., Hu, M., Gan, C., Han, S.: EfficientViT: Lightweight multi-scale attention for high-resolution dense prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 17302–17313 (2023)
5. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. CoRR **abs/1904.10509** (2019), http://arxiv.org/abs/1904.10509
6. Chiu, J., Demanet, L.: Sublinear randomized algorithms for skeleton decompositions. SIAM Journal on Matrix Analysis and Applications **34**(3), 1361–1383 (2013)
7. Choromanski, K.M., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J.Q., Mohiuddin, A., Kaiser, L., Belanger, D.B., Colwell, L.J., Weller, A.: Rethinking attention with performers. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=Ua6zuk0WRH
8. Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B.: CSWin transformer: A general vision transformer backbone with cross-shaped win-

dows. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12124–12134 (2022)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=YicbFdNTTy
10. Drineas, P., Mahoney, M.W., Muthukrishnan, S.: Relative-error CUR matrix decompositions. SIAM Journal on Matrix Analysis and Applications **30**(2), 844–881 (2008)
11. Gani, H., Naseer, M., Yaqub, M.: How to train vision transformer on small-scale datasets? In: 33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022. BMVA Press (2022), https://bmvc2022.mpi-inf.mpg.de/0731.pdf
12. Goreinov, S.A., Tyrtyshnikov, E.E., Zamarashkin, N.L.: A theory of pseudoskeleton approximations. Linear Algebra and its Applications **261**(1-3), 1–21 (1997)
13. Han, D., Pan, X., Han, Y., Song, S., Huang, G.: FLatten transformer: Vision transformer using focused linear attention. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5961–5971 (2023)
14. Han, J., Zeng, L., Du, L., Ye, X., Ding, W., Feng, J.: Modify self-attention via skeleton decomposition for effective point cloud transformer. Proceedings of the AAAI Conference on Artificial Intelligence **36**(1), 808–816 (2022). https://doi.org/10.1609/aaai.v36i1.19962
15. Hassani, A., Walton, S., Li, J., Li, S., Shi, H.: Neighborhood attention transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6185–6194 (2023)
16. Hatamizadeh, A., Yin, H., Heinrich, G., Kautz, J., Molchanov, P.: Global context vision transformers. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (eds.) Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 12633–12646. PMLR (2023), https://proceedings.mlr.press/v202/hatamizadeh23a.html
17. Islam, K.: Recent advances in vision transformer: A survey and outlook of recent work. arXiv preprint arXiv:2203.01536 (2022)
18. Katharopoulos, A., Vyas, A., Pappas, N., Fleuret, F.: Transformers are RNNs: Fast autoregressive transformers with linear attention. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 5156–5165. PMLR (2020), https://proceedings.mlr.press/v119/katharopoulos20a.html
19. Kitaev, N., Kaiser, L., Levskaya, A.: Reformer: The efficient transformer. In: International Conference on Learning Representations (2020), https://openreview.net/forum?id=rkgNKkHtvB
20. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto (2009), https://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf
21. Liu, Y., Sangineto, E., Bi, W., Sebe, N., Lepri, B., Nadai, M.: Efficient training of visual transformers with small datasets. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems. vol. 34, pp. 23818–23830. Curran Associates, Inc. (2021), https://proceedings.neurips.cc/paper_files/paper/2021/file/c81e155d85dae5430a8cee6f2242e82c-Paper.pdf

22. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10012–10022 (2021)
23. Loshchilov, I., Hutter, F.: Fixing weight decay regularization in Adam (2018), https://openreview.net/forum?id=rk6qdGgCZ
24. Lu, J., Yao, J., Zhang, J., Zhu, X., Xu, H., Gao, W., Xu, C., Xiang, T., Zhang, L.: SOFT: Softmax-free transformer with linear complexity. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems. vol. 34, pp. 21297–21309. Curran Associates, Inc. (2021), https://proceedings.neurips.cc/paper_files/paper/2021/file/b1d10e7bafa4421218a51b1e1f1b0ba2-Paper.pdf
25. Mareček, D., Rosa, R.: From Balustrades to Pierre Vinken: Looking for syntax in transformer self-attentions. In: Linzen, T., Chrupała, G., Belinkov, Y., Hupkes, D. (eds.) Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP. pp. 263–275. Association for Computational Linguistics, Florence, Italy (2019). https://doi.org/10.18653/v1/W19-4827
26. Peng, H., Pappas, N., Yogatama, D., Schwartz, R., Smith, N., Kong, L.: Random feature attention. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=QtTKTdVrFBB
27. Qin, Z., Sun, W., Deng, H., Li, D., Wei, Y., Lv, B., Yan, J., Kong, L., Zhong, Y.: cosFormer: Rethinking softmax in attention. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=Bl8CQrx2Up4
28. Razavi, M.K., Kerayechian, A., Gachpazan, M., Shateyi, S.: A new iterative method for finding approximate inverses of complex matrices. Abstract and Applied Analysis **2014**, Article ID 563787, 7 pages (2014)
29. Rudelson, M.: Random vectors in the isotropic position. Journal of Functional Analysis **164**(1), 60–72 (1999)
30. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
31. Shen, Z., Zhang, M., Zhao, H., Yi, S., Li, H.: Efficient attention: Attention with linear complexities. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 3531–3539 (2021)
32. Sorensen, D.C., Embree, M.: A DEIM induced CUR factorization. SIAM Journal on Scientific Computing **38**(3), A1454–A1482 (2016)
33. Stacey, J., Belinkov, Y., Rei, M.: Supervising model attention with human explanations for robust natural language inference. Proceedings of the AAAI Conference on Artificial Intelligence **36**(10), 11349–11357 (2022). https://doi.org/10.1609/aaai.v36i10.21386
34. Tay, Y., Bahri, D., Yang, L., Metzler, D., Juan, D.C.: Sparse Sinkhorn attention. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 9438–9447. PMLR (2020), https://proceedings.mlr.press/v119/tay20a.html
35. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers & distillation through attention. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 10347–10357. PMLR (2021), https://proceedings.mlr.press/v139/touvron21a.html

36. Tyrtyshnikov, E.: Incomplete cross approximation in the mosaic-skeleton method. Computing **64**, 367–380 (2000)
37. Udell, M., Townsend, A.: Why are big data matrices approximately low rank? SIAM Journal on Mathematics of Data Science **1**(1), 144–160 (2019)
38. Vasu, P.K.A., Gabriel, J., Zhu, J., Tuzel, O., Ranjan, A.: FastViT: A fast hybrid vision transformer using structural reparameterization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5785–5795 (2023)
39. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
40. Voita, E., Talbot, D., Moiseev, F., Sennrich, R., Titov, I.: Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In: Korhonen, A., Traum, D., Màrquez, L. (eds.) Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 5797–5808. Association for Computational Linguistics, Florence, Italy (2019). https://doi.org/10.18653/v1/P19-1580
41. Wang, H., Wang, Z., Du, M., Yang, F., Zhang, Z., Ding, S., Mardziel, P., Hu, X.: Score-CAM: Score-weighted visual explanations for convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (2020)
42. Wang, S., Li, B.Z., Khabsa, M., Fang, H., Ma, H.: Linformer: Self-attention with linear complexity. arXiv preprint arXiv:2006.04768 (2020)
43. Wortsman, M., Lee, J., Gilmer, J., Kornblith, S.: Replacing softmax with ReLU in vision transformers. arXiv preprint arXiv:2309.08586 (2023)
44. Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y., Singh, V.: Nyströmformer: A Nyström-based algorithm for approximating self-attention. Proceedings of the AAAI Conference on Artificial Intelligence **35**(16), 14138–14148 (2021). https://doi.org/10.1609/aaai.v35i16.17664
45. Zaheer, M., Guruganesh, G., Dubey, K.A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., Ahmed, A.: Big Bird: Transformers for longer sequences. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 17283–17297. Curran Associates, Inc. (2020), https://proceedings.neurips.cc/paper_files/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf
46. Zhang, L., Lu, J., Zhang, J., Zhu, X., Feng, J., Xiang, T.: Softmax-free linear transformers. arXiv preprint arXiv:2207.03341 (2022)
47. Zhang, X., Bosselut, A., Yasunaga, M., Ren, H., Liang, P., Manning, C.D., Leskovec, J.: GreaseLM: Graph REASoning enhanced language models. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=41e9o6cQPj
48. Zhang, Z., Zhang, H., Zhao, L., Chen, T., Arik, S.Ö., Pfister, T.: Nested hierarchical transformer: Towards accurate, data-efficient and interpretable visual understanding. Proceedings of the AAAI Conference on Artificial Intelligence **36**(3), 3417–3425 (2022). https://doi.org/10.1609/aaai.v36i3.20252
49. Zhao, G., Lin, J., Zhang, Z., Ren, X., Su, Q., Sun, X.: Explicit sparse transformer: Concentrated attention through explicit selection. CoRR **abs/1912.11637** (2019), http://arxiv.org/abs/1912.11637