Nyströmformer: A Nyström-based Algorithm for Approximating Self-Attention

Yunyang Xiong 1  Zhanpeng Zeng 1  Rudrasis Chakraborty 2  Mingxing Tan 3  Glenn Fung 4  Yin Li 1  Vikas Singh 1

1 University of Wisconsin-Madison  2 UC Berkeley  3 Google Brain  4 American Family Insurance
yiong43@wisc.edu, zzeng38@wisc.edu, rudra@berkeley.edu, tanningxing@gmail.com, gfung@amfam.com, yin.li@wisc.edu, vsingh@biostat.wisc.edu

Abstract

Transformers have emerged as a powerful tool for a broad range of natural language processing tasks. A key component that drives the impressive performance of Transformers is the self-attention mechanism that encodes the influence or dependence of other tokens on each specific token. While beneficial, the quadratic complexity of self-attention on the input sequence length has limited its application to longer sequences – a topic being actively studied in the community. To address this limitation, we propose Nyströmformer – a model that exhibits favorable scalability as a function of sequence length. Our idea is based on adapting the Nyström method to approximate standard self-attention with $O(n)$ complexity. The scalability of Nyströmformer enables application to longer sequences with thousands of tokens. We perform evaluations on multiple downstream tasks on the GLUE benchmark and IMDB reviews with standard sequence length, and find that our Nyströmformer performs comparably, or in a few cases, even slightly better, than standard Transformer. Our code is at https://github.com/mlpen/Nystromformer

Introduction

Transformer-based models, such as BERT (Devlin et al. 2019) and GPT-3 (Brown et al. 2020), have been very successful in natural language processing (NLP), achieving state-of-the-art performance in machine translation (Vaswani et al. 2017), natural language inference (Williams, Nangia, and Bowman 2018), paraphrasing (Dolan and Brockett 2005), text classification (Howard and Ruder 2018), question answering (Rajpurkar et al. 2016) and many other NLP tasks (Peters et al. 2018; Radford et al. 2018). A key feature of transformers is what is known as the self-attention mechanism (Vaswani et al. 2017), where each token’s representation is computed from all other tokens. Self-attention enables interactions of token pairs across the full sequence and has been shown quite effective.

Despite the foregoing advantages, self-attention also turns out to be a major efficiency bottleneck since it has a memory and time complexity of $O(n^2)$ where $n$ is the length of an input sequence. This leads to high memory and computational requirements for training large Transformer-based models. For example, training a BERT-large model (Devlin et al. 2019) will need 4 months using a single Tesla V100 GPU (equivalent to 4 days using a 4x4 TPU pod). Further, the $O(n^2)$ complexity makes it prohibitively expensive to train large Transformers with long sequences (e.g., $n = 2048$).

To address this challenge, several recent works have proposed strategies that avoid incurring the quadratic cost when dealing with longer input sequences. For example, (Dai et al. 2019) suggests a trade-off between memory and computational efficiency. The ideas described in (Child et al. 2019; Kitaev, Kaiser, and Levskaya 2019) decrease the self-attention complexity to $O(n\sqrt{n})$ and $O(n \log n)$ respectively. In (Shen et al. 2018b; Katharopoulos et al. 2020; Wang et al. 2020), self-attention complexity can be reduced to $O(n)$ with various approximation ideas, each with its own strengths and limitations.

In this paper, we propose a $O(n)$ approximation, both in the sense of memory and time, for self-attention. Our model, Nyströmformer, scales linearly with the input sequence length $n$. This is achieved by leveraging the celebrated Nyström method, repurposed for approximating self-attention. Specifically, our NyströmFormer algorithm makes use of landmark (or Nyström) points to reconstruct the softmax matrix in self-attention, thereby avoiding computing the $n \times n$ softmax matrix. We show that this yields a good approximation of the true self-attention.

To evaluate our method, we consider a transfer learning setting using Transformers, where models are first pretrained with a language modeling objective on a large corpus, and then finetuned on target tasks using supervised data (Devlin et al. 2019; Liu et al. 2019; Lewis et al. 2020; Wang et al. 2020). Following BERT (Devlin et al. 2019; Liu et al. 2019), we pretrain our proposed model on English Wikipedia and BookCorpus (Zhu et al. 2015) using a masked-language-modeling objective. We observe a similar performance to the baseline BERT model on English Wikipedia and BookCorpus. We then finetune our pretrained models on multiple downstream tasks in the GLUE benchmark (Wang et al. 2018) and IMDB reviews (Maas et al. 2011), and compare our results to BERT in both accuracy and efficiency. Across all tasks, our model compares favorably to the vanilla pretrained BERT with promising speedups. Our model also outperforms several recent efficient transformer models, thus providing a step towards resource efficient Transformers.
Related Work

We briefly review a few results on efficient Transformers, linearized Softmax kernels and Nyström-like methods.

Efficient Transformers. Weight pruning [Michel, Levy, and Neubig 2019], weight factorization [Lan et al. 2020], weight quantization [Zafrir et al. 2019] or knowledge distillation [Sanh et al. 2019] are several strategies that have been proposed to improve memory efficiency in transformers. The use of a new pretraining objective in (Clark et al. 2019), product-key attention in (Lample et al. 2019), and the Transformer-XL model in (Dai et al. 2019) have shown how the overall compute requirements can be reduced. In (Child et al. 2019), a sparse factorization of the attention matrix was used for reducing the overall complexity from quadratic to $O(n\sqrt{n})$ for generative modeling of long sequences. In (Kitaev, Kaiser, and Levskaya 2019), the Reformer model further reduces the complexity to $O(n \log n)$ via locality-sensitive-hashing (LSH). This relies on performing fewer dot product operations overall by assuming that the keys need to be identical to the queries. Recently, in (Wang et al. 2020), the Linformer model suggested the use of random projections based on the JL lemma to reduce the complexity of landmarks selection. Nonetheless, inspired by the key idea of using a subset of columns or rows of a softmax matrix will require the calculation of all elements in the full matrix before the softmax function. Calculating these entries will incur a quadratic complexity in our case. Nonetheless, inspired by the key idea of using a subset of columns to reconstruct the full matrix, we propose a Nyström approximation with $O(n)$ complexity tailored for the softmax matrix, for efficiently computing self-attention.

Nyström-Based Linear Transformers

In this section, we start by briefly reviewing self-attention, then discuss the basic idea of Nyström approximation method for the softmax matrix in self-attention, and finally adapting this idea to achieve our proposed construction.

Self-Attention

What is self-attention? Self-attention calculates a weighted average of feature representations with the weight proportional to a similarity score between pairs of representations. Formally, an input sequence of $n$ tokens of dimensions $d$, $X \in \mathbb{R}^{n \times d}$, is projected using three matrices $W_Q \in \mathbb{R}^{d \times d_Q}$, $W_K \in \mathbb{R}^{d \times d_K}$, and $W_V \in \mathbb{R}^{d \times d_V}$ to extract feature representations $Q$, $K$, and $V$, referred to as query, key, and value respectively with $d_K = d_Q$. The outputs $Q$, $K$, $V$ are computed as

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V. \quad (1)$$

So, self-attention can be written as,

$$S = D(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_q}} \right) V, \quad (2)$$

where softmax denotes a row-wise softmax normalization function. Thus, each element in $S$ depends on all other elements in the same row.

Compute cost of self-attention. The self-attention mechanism requires calculating $n^2$ similarity scores between each pair of tokens, leading to a complexity of $O(n^2)$ for both memory and time. Due to this quadratic dependence on the input length, the application of self-attention is limited to short sequences (e.g., $n < 1000$). This is a key motivation for a resource-efficient self-attention module.
Nyström Method for Matrix Approximation

The starting point of our work is to reduce the computational cost of self-attention in Transformers using the Nyström method, widely adopted for matrix approximation (Williams and Seeger 2001, Drineas and Mahoney 2005, Wang and Zhang 2013). Following (Wang and Zhang 2013), we describe a potential strategy and its challenges for using the Nyström method to approximate the softmax matrix in self-attention by sampling a subset of columns and rows.

Denote the softmax matrix used in self-attention, \( S = \text{softmax}( \frac{QK^T}{\sqrt{d_q}} ) \), where \( Q \in \mathbb{R}^{n \times m} \) is the query matrix, and \( K \in \mathbb{R}^{m \times n} \) is the key matrix. The softmax operation can be approximated by the Nyström approximation, \( \hat{S} \), which is defined as

\[
\hat{S} = \text{softmax}( \frac{QK^T}{\sqrt{d_q}} ) = \begin{bmatrix} A_S & B_S \\ F_S & C_S \end{bmatrix},
\]

where \( A_S, B_S, C_S, F_S \in \mathbb{R}^{m \times m} \) are matrices, \( A_S \) is designated to be our sample matrix by sampling \( m \) columns and \( m \) rows from \( S \).

**Quadrature technique.** \( S \) can be approximated via the basic quadrature technique of the Nyström method. It begins with the singular value decomposition (SVD) of the sample matrix, \( A_S = U \Sigma V^T \), where \( U, V \in \mathbb{R}^{m \times m} \) are orthogonal matrices, \( \Lambda \in \mathbb{R}^{m \times m} \) is a diagonal matrix. Based on the out-of-sample columns approximation (Wang and Zhang 2013), the explicit Nyström form of \( S \) can be reconstructed with \( m \) columns and \( m \) rows from \( S \),

\[
\hat{S} = \begin{bmatrix} A_S & B_S \\ F_S & C_S \end{bmatrix} = \begin{bmatrix} A_S & B_S \\ F_S & C_S \end{bmatrix} \hat{S},
\]

where \( \hat{S} \) is the Moore-Penrose inverse of \( A_S \). \( C_S \) is approximated by \( F_S A_S^+ B_S \). Here, (4) suggests that the \( n \times n \) matrix \( S \) can be reconstructed by sampling \( m \) rows \( (A_S, B_S) \) and \( m \) columns \( (A_S, F_S) \) from \( S \) and finding the Nyström approximation \( \hat{S} \).

**Nyström approximation for softmax matrix.** We briefly discuss how to construct the out-of-sample approximation for the softmax matrix in self-attention using the standard Nyström method. Given a query \( q_i \), and key \( k_j \), let

\[
K_K(q_i) = \text{softmax}( \frac{q_i K^T}{\sqrt{d_q}} ), \quad K_Q(k_j) = \text{softmax}( \frac{Q k_j^T}{\sqrt{d_q}} )
\]

where \( K_K(q_i) \in \mathbb{R}^{1 \times n} \) and \( K_Q(k_j) \in \mathbb{R}^{n \times 1} \). We can then construct

\[
\phi_K(q_i) = \Lambda^{-\frac{1}{2}} V^T [K_K(q_i)]_{m \times 1}, \\
\phi_Q(k_j) = \Lambda^{-\frac{1}{2}} U^T [K_Q(k_j)]_{m \times 1},
\]

where \([ ]_{m \times 1}\) refers to calculating the full \( n \times 1 \) vector and then taking the first \( m \) entries. With \( \phi_K(q_i) \) and \( \phi_Q(k_j) \) available in hand, the entry of \( \hat{S} \) for standard Nyström approximation is calculated as,

\[
\hat{S}_{ij} = \phi_K(q_i)^T \phi_Q(k_j), \forall i = 1, \ldots, n, j = 1, \ldots, n
\]

**A key challenge of Nyström approximation.** Unfortunately, (4) and (5) require calculating all entries in \( QK^T \) due to the softmax function, even though the approximation only needs to access a subset of the columns of \( S \), i.e., \([A_S \quad F_S]\). The problem arises due to the denominator within the row-wise softmax function. Specifically, computing an element in \( S \) requires a summation of the exponential of all elements in the same row of \( QK^T \). Thus, calculating \([A_S \quad F_S]\) needs accessing the full \( QK^T \), shown in Fig. 1, and directly applying Nyström approximation as in (4) is not attractive.

**Linearized Self-Attention via Nyström Method.**

We now adapt the Nyström method to approximately calculate the full softmax matrix \( S \). The basic idea is to use landmarks \( \tilde{K} \) and \( \tilde{Q} \) from key \( K \) and query \( Q \) to derive an efficient Nyström approximation without accessing the full \( QK^T \). When the number of landmarks, \( m \), is much smaller than the sequence length \( n \), our Nyström approximation scales linearly w.r.t. input sequence length in the sense of both memory and time.

Following the Nyström method, we also start with the SVD of a smaller matrix, \( A_S \), and apply the basic quadrature technique. But instead of subsampling the matrix after the softmax operation, we select landmarks \( \tilde{Q} \) from queries \( Q \) and \( \tilde{K} \) from keys \( K \) before softmax and then form a \( m \times m \) matrix \( \tilde{A}_S \) by applying the softmax operation on the landmarks. We also form the matrices corresponding to the left
and right matrices in (4) using landmarks \( \tilde{Q} \) and \( \tilde{K} \). This provides a \( n \times m \) matrix and \( m \times n \) matrix respectively. With these three \( n \times m, m \times n, m \times n \) matrices we constructed, our Nyström approximation of the \( n \times n \) matrix \( S \) involves the multiplication of three matrices as in (4).

In the description that follows, we first define the matrix form of landmarks. Then, based on the landmarks matrix, we form the three matrices needed for our approximation.

**Definition 1.** Let us assume that the selected landmarks for inputs \( Q = [q_1; \ldots; q_n] \) and \( K = [k_1; \ldots; k_n] \) are \( \{q_j\}_{j=1}^m \) and \( \{k_j\}_{j=1}^m \) respectively. We denote the matrix form of the corresponding landmarks as

- For \( \{q_j\}_{j=1}^m \), \( \tilde{Q} = [q_1; \ldots; q_m] \in \mathbb{R}^{n \times d_q} \)
- For \( \{k_j\}_{j=1}^m \), \( \tilde{K} = [k_1; \ldots; k_m] \in \mathbb{R}^{m \times d_q} \)

The corresponding \( m \times m \) matrix is generated by

\[
A_S = \text{softmax} \left( \frac{\tilde{Q} \tilde{K}^T}{\sqrt{d_q}} \right) \quad \text{where} \quad A_S = U_{m \times m} A_{m \times m} V_{m \times m}^T
\]

Note that in the SVD decomposition of \( A_S \), \( U_{m \times m} \) and \( V_{m \times m} \) are orthogonal matrices. Similar to the out-of-sample approximation procedure for the standard Nyström scheme described above, given a query \( q_i \) and key \( k_j \), let

\[
K_{\tilde{K}}(q_i) = \text{softmax} \left( \frac{q_i \tilde{K}^T}{\sqrt{d_q}} \right) \quad \text{and} \quad K_{\tilde{Q}}(k_j) = \text{softmax} \left( \frac{\tilde{Q} k_j^T}{\sqrt{d_q}} \right),
\]

where \( K_{\tilde{K}}(q_i) \in \mathbb{R}^{1 \times m} \) and \( K_{\tilde{Q}}(k_j) \in \mathbb{R}^{m \times 1} \). We can then construct

\[
\phi_{\tilde{K}}(q_i) = \Lambda_{m \times m}^{\frac{1}{2}} V_{m \times m}^T K_{\tilde{K}}(q_i) \quad \text{and} \quad \phi_{\tilde{Q}}(k_j) = \Lambda_{m \times m}^{-\frac{1}{2}} U_{m \times m}^T K_{\tilde{Q}}(k_j)
\]

So, the entry for \( \tilde{S} \) depends on landmark matrices \( \tilde{K} \) and \( \tilde{Q} \) and is calculated as,

\[
\tilde{S}_{ij} = \phi_{\tilde{K}}(q_i)^T \phi_{\tilde{Q}}(k_j), \forall i = 1, \ldots, n, j = 1, \ldots, n,
\]

To derive the explicit Nyström form, \( \hat{S} \), of the softmax matrix with the three \( n \times m, m \times m, m \times n \) matrices, we assume that \( A_S \) is non-singular first to guarantee that the above expression to define \( \phi_{\tilde{K}} \) and \( \phi_{\tilde{Q}} \) is meaningful. We will shortly relax this assumption to achieve the general form as (4).

**Algorithm 1:** Pipeline for Nyström approximation of softmax matrix in self-attention

**Input:** Query \( Q \) and Key \( K \).

**Output:** Nyström approximation of softmax matrix. Compute landmarks from input \( Q \) and landmarks from input \( K, \tilde{Q} \) and \( \tilde{K} \) as the matrix form ;

1. Compute \( \tilde{F} = \text{softmax}(\frac{QK^T}{\sqrt{d_q}}) \), \( \tilde{B} = \text{softmax}(\frac{QK^T}{\sqrt{d_q}})^+ \);
2. Compute \( \hat{A} = \text{softmax}(\frac{\tilde{Q} \tilde{K}^T}{\sqrt{d_q}}) \);
3. return \( \hat{F} \times \hat{A} \times \tilde{B} \);

When \( A_S \) is non-singular,

\[
\hat{S}_{ij} = \phi_{\tilde{K}}(q_i)^T \phi_{\tilde{Q}}(k_j)
\]

(8)

\[
= \frac{K_{\tilde{K}}(q_i) V_{m \times m} A_{m \times m}^{-1} U_{m \times m}^T K_{\tilde{Q}}(k_j)}{\Lambda_{m \times m}^{\frac{1}{2}} V_{m \times m}^T K_{\tilde{K}}(q_i) A_{m \times m}^{-1} K_{\tilde{Q}}(k_j)}.
\]

(9)

Let \( W_m = V_{m \times m} A_{m \times m}^{-1} U_{m \times m}^T \). Recall that a SVD of \( A_S \) is \( U_{m \times m} A_{m \times m} V_{m \times m}^T \), and so, \( W_m A_S = I_{m \times m} \). Therefore,

\[
\hat{S}_{ij} = \frac{K_{\tilde{K}}(q_i) A_S^{-1} K_{\tilde{Q}}(k_j)}{\Lambda_{m \times m}^{\frac{1}{2}} V_{m \times m}^T K_{\tilde{K}}(q_i) A_S^{-1} K_{\tilde{Q}}(k_j)}.
\]

(10)

Based on (10), we can rewrite it to have a similar form as (4) (i.e., not requiring that \( A_S \) is non-singular) as

\[
\hat{S}_{ij} = K_{\tilde{K}}(q_i)^T A_S^{-1} K_{\tilde{Q}}(k_j),
\]

(11)

where \( A_S^{-1} \) is a Moore-Penrose pseudoinverse of \( A_S \). So,

\[
\hat{S}_{ij} = \text{softmax} \left( \frac{q_i \tilde{K}^T}{\sqrt{d_q}} \right) A_S^{-1} \text{softmax} \left( \frac{\tilde{Q} k_j^T}{\sqrt{d_q}} \right),
\]

(12)

for \( i, j = \{1, \ldots, n\} \). The Nyström form of the softmax matrix, \( S = \text{softmax} \left( \frac{QK^T}{\sqrt{d_q}} \right) \) is thus approximated as

\[
\hat{S} = \text{softmax} \left( \frac{QK^T}{\sqrt{d_q}} \right) \left( \text{softmax} \left( \frac{QK^T}{\sqrt{d_q}} \right)^+ \right)^{-1} \text{softmax} \left( \frac{QK^T}{\sqrt{d_q}} \right) \]

(13)

Note that we arrive at (13) via an out-of-sample approximation similar to (4). The key difference is that that in (13), the landmarks are selected before the softmax operation to generate the out-of-sample approximation. This avoids the need to compute the full softmax matrix \( S \) for a Nyström approximation. Fig. 2 illustrates the proposed Nyström approximation and Alg. 1 summarizes our method.

We now describe (a) the calculation of the Moore-Penrose inverse and (b) the selection of landmarks.

**Moore-Penrose inverse computation.** Moore-Penrose pseudoinverse can be calculated by using singular value decomposition. However, SVD is not very efficient on GPUs. To accelerate the computation, we use an iterative method from (Razavi et al. 2014) to approximate the Moore-Penrose inverse efficient matrix multiplications.

**Lemma 1.** For \( A_S \in \mathbb{R}^{m \times n} \), the sequence \( \{Z_j\}_{j=0}^{\infty} \) generated by (Razavi et al. 2014),

\[
Z_{j+1} = \frac{1}{4} Z_j (13I - A_S Z_j (15I - A_S Z_j) (7I - A_S Z_j))
\]

(14)
converges to the Moore-Penrose inverse $A_S^+$ in the third-order with initial approximation $Z_0$ satisfying $||A_S A_S^+ - A_S Z_0|| < 1$.

We select $Z_0$ by $Z_0 = A_S / (||A_S||_2 ||A_S||_\infty)$ where

$$||A_S||_1 = \max_j \sum_i |(A_S)_{ij}|; \quad ||A_S||_\infty = \max_i \sum_j |(A_S)_{ij}|,$$

based on (Pan and Schreiber [1991]). This choice ensures that $||I - A_S Z_0||_2 < 1$. When $A_S$ is non-singular,

$$||A_S A_S^+ - A_S Z_0||_2 = ||I - A_S Z_0||_2 < 1.$$ Without the non-singular constraint, the choice of initializing $Z_0$ provides a good approximation in our experiments. For all our experiments, we need to run about 6 iterations in order to achieve a good approximation of the pseudoinverse.

Let $A_S$ be approximated by $Z^*$ with (14). Our Nyström approximation of $S$ can be calculated as

$$\tilde{S} = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) Z^* \text{softmax} \left( \frac{\tilde{Q}K^T}{\sqrt{d_q}} \right).$$

Here, (15) only needs matrix-matrix multiplication, thus the gradient computation is straightforward.

**Landmarks selection.** Landmark points (inducing points (Lee et al. 2019)) can be selected by using K-means clustering (Zhang, Tsang, and Kwok 2008; Vyas, Katharopoulos, and Fleuret 2020). However, the EM style of updates in K-means is less desirable during mini-batch training. We propose to simply use Segment-means similar to the local average pooling previously used in the NLP literature (Shen et al. 2018a). Specifically, for input queries $Q = [q_1; \ldots; q_n]$, we separate the $n$ queries into $m$ segments. As we can pad inputs to a length divisible to $m$, we assume $n$ is divisible by $m$ for simplicity. Let $l = n/m$, landmark points for $Q$ are computed in (16). Similarly, for input keys $K = [k_1; \ldots; k_n]$, landmarks are computed as shown in (16).

$$\tilde{q}_j = \frac{\sum_{i=(j-1)x+l}^{(j-1)x+l+m} q_i}{m}, \quad \tilde{K}_j = \frac{\sum_{i=(j-1)x+l}^{(j-1)x+l+m} k_i}{m},$$

where $j = 1, \ldots, m$. Segment-means requires a single scan of the sequence to compute the landmarks leading to a complexity of $O(n)$. We find that using 64 landmarks is often sufficient to ensure a good approximation, although this depends on the application. More details regarding the landmark selection is in the supplement.

**Approximate self-attention.** With landmark points and pseudoinverse computed, the Nyström approximation of the softmax matrix can be calculated. By plugging in the Nyström approximation, we obtain a linearized version $\tilde{S}V$, to approximate the true self-attention $SV$,

$$\tilde{S}V = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) Z^* \text{softmax} \left( \frac{\tilde{Q}K^T}{\sqrt{d_q}} \right) V.$$  (17)

Fig. 3 presents an example of the fidelity between Nyström approximate self-attention versus true self-attention.

![Figure 3: An example of Nyström approximation vs. ground-truth self-attention. Top: standard self-attention computed by $QK$. Bottom: self-attention from our proposed Nyström approximation in (17). We see that the attention patterns are quite similar.](image)

**Complexity analysis.** We now provide a complexity analysis of the Nyström approximation which needs to account for landmark selection, pseudoinverse calculation, and the matrix multiplications. Landmark selection using Segment-means takes $O(n)$. Iterative approximation of the pseudoinverse takes $O(m^3)$ in the worst case. The matrix multiplication first calculates $\text{softmax}(QK^T/\sqrt{d_k}) \times Z^*$ and $\text{softmax}(\tilde{Q}K^T/\sqrt{d_q}) \times V$, and then calculates the product $(\text{softmax}(QK^T/\sqrt{d_k}) \times Z^*) \times (\text{softmax}(\tilde{Q}K^T/\sqrt{d_q}) \times V)$. This costs $O(nm^2 + mn^2 + m^3 + nm^2)$. The overall time complexity is thus $O(n + m^3 + mn + mn + nd_v)$. For memory cost, storing landmarks matrix $\tilde{Q}$ and $\tilde{K}$ involves $O(md_q)$ cost and storing four Nyström approximation matrices has a $O(nn^2 + mn + mn + nd_v)$ cost. Thus, the memory footprint is $O(mnd_q + mn + mn + nd_v + nd_v)$. When the number of landmarks $m \ll n$, the time and memory complexity of our Nyström approximation is $O(n)$, i.e., scales linearly w.r.t. the input sequence length $n$.

**Analysis of Nyström Approximation.** The following simple result states that the Galerkin discretization of $\phi_R(k)^T \phi_R(k)$ with the same set of quadrature and landmark points, induces the same Nyström matrix, in particular, the same $n \times n$ Nyström approximation $\tilde{S}_{ij}$. This result agrees with the discussion in (Bremer 2012).

**Lemma 2.** Given the input data set $Q = \{q_1; \ldots; q_n\}$ and $K = \{k_1; \ldots; k_n\}$, and the corresponding landmark point set $\tilde{Q} = \{\tilde{q}_1; \ldots; \tilde{q}_m\}$ and $\tilde{K} = \{\tilde{k}_j\}_{j=1}^m$. Using (17), the Nyström approximate self-attention converges to true self-attention if there exist landmarks points $\tilde{q}_p$ and $\tilde{k}_t$ such that $\tilde{q}_p = q_i$ and $\tilde{k}_t = k_j$, $\forall i = 1, \ldots, n$, $j = 1, \ldots, m$.

Lemma 2 suggests that if the landmark points overlap sufficiently with the original data points, the approximation to self-attention will be good. While the condition here is problem dependent, we note that it is feasible to achieve an accurate approximation without using a large number of landmarks. This is because (Oglic and Gärtner 2017) points out that the error of Nyström approximation depends on the spectrum of the matrix to be approximated and it decreases with the rank of the matrix. When this result is compared
with the observation in [Wang et al. 2020] that suggests that self-attention is low-rank, stronger guarantees based on structural properties of the matrix that we wish to approximate are possible.

Our Model: Nyströmformer

Architecture. Our proposed architecture is shown in Fig. 4. Given the input key $K$ and query $Q$, our model first uses Segment-means to compute landmark points as matrices $\tilde{K}$ and $\tilde{Q}$. With the landmark points, our model then calculates the Nyström approximation using approximate Moore-Penrose pseudoinverse. A skip connection of value $V$, implemented using a 1D depthwise convolution, is also added to the model to help the training.

Experiments

We now present our experiments and results. Our experiments follow a transfer learning setting that consists of two stages. In the first stage, we train our Nyströmformer on a large-scale text corpus, and report the language modeling performance of our model on a hold-out validation set. In the second stage, we fine-tune the pre-trained Nyströmformer across several different NLP tasks in GLUE benchmarks (Wang et al. 2019) and IMDB reviews (Maas et al. 2011), and report the performance on individual dataset for each task. In both stages, we compare our results to a baseline Transformer model – BERT (Devlin et al. 2019). Specifically, we consider two variants of BERT:

- **BERT-small** is a light weighted BERT model with 4 layers. We use BERT-small to compare to linear Transformers, including ELU linearized self-attention (Katharopoulos et al. 2020) and Linformer (Wang et al. 2020).

- **BERT-base** is the base model from (Devlin et al. 2019). We use this model as our baseline when fine-tuning on downstream NLP tasks.

Our Nyströmformer replaces the self-attention in BERT-small and BERT-base using the proposed Nyström approximation. We acknowledge that several very recent articles (Zaheer et al. 2020; Beltagy, Peters, and Cohan 2020), concurrent with our work, have also proposed efficient $O(n)$ self-attention for Transformers. An exhaustive comparison to a rapidly growing set of algorithms is prohibitive unless extensive compute resources are freely available. Thus, we only compare runtime performance and the memory consumption of our method to Linformer (Wang et al. 2020) and Longformer (Beltagy, Peters, and Cohan 2020) in Table 1.

Implementation details. Our model is pre-trained with the masked-language-modeling (MLM) and sentence-order-prediction (SOP) objectives (Lan et al. 2020). We use a batch size of 256, optimizer Adam with learning rate 1e-4,
Table 1: Memory consumption and running time results on various input sequence length. We report the average memory consumption (MB) and running time (ms) for one input instance with different input length through self-attention module. Nyströmformer-64 denotes Nyströmformer self-attention module using 64 landmarks and Nyströmformer-32 denotes Nyströmformer module using 32 landmarks. Linformer-256 denotes Linformer self-attention module using linear projection dimension 256. Longformer-256 denotes Longformer self-attention using sliding window size 257×(128×2+1). Our Nyström self-attention offers favorable memory and time efficiency over standard self-attention and Longformer self-attention. With a length of 8192, our model offers 1.2× memory saving and 3× speed-up over Longformer, and 1.7× memory saving over Linformer with similar running time.

\[ \beta_1 = 0.9, \beta_2 = 0.999, \text{L2 weight decay of 0.01, learning rate warm-up over the first 10,000 steps, and linear learning rate decay to update our model. Training BERT-base with 1M update steps takes more than one week on 8 V100 GPUs. To keep cost compute reasonable, our baseline (BERT-base) and our model are trained with 0.5M steps. We also train our model with \sim 0.25M steps, initialized from pre-trained BERT-base to see training speed-up. For BERT-small, we train for 0.1M steps. More details are available in the supplement.} \]

Results on accuracy and efficiency. We report the validation accuracy and inference efficiency of our model and compare the results to transformer based models. In Fig. 5 and 6 we plot MLM and SOP pre-training validation accuracy, which shows that Nyströmformer is comparable to a standard transformer and outperforms other variants of efficient transformers. We also note the computation and memory efficiency of our model in Table 1. To evaluate the inference time and memory efficiency, we generate random inputs for self-attention module with sequence length \( n \in [512, 1024, 2048, 4096, 8192] \). All models are evaluated on the same machine setting with Nvidia 1080Ti and we report the improved inference speed and memory saving.

Fine-tuning on Downstream NLP tasks

Our second experiment is designed to test the generalization ability of our model on downstream NLP tasks. To this end, we fine-tune the pretrained model across several NLP tasks.

Datasets and metrics. We consider the datasets of SST-2 [Socher et al. 2013], MRPC (Dolan and Brockett 2005), and MRPC, due to the sensitivity of a smaller dataset, we follow [2e-5, 3e-5, 4e-5, 5e-5] with candidate batch size \([0, 32, 64, 128, 256] \). We fine-tune our pretrained model on GLUE benchmark datasets and IMDB reviews (Maas et al. 2011). We follow the standard evaluation protocols, fine-tune the pretrained BERT-base, and initializing from pretrained BERT-base help speed up training.

Implementation details. We fine-tune our pre-trained model on GLUE benchmark datasets and IMDB reviews respectively and report its final performance. For larger datasets (SST-2, QNL, QQP, MNLI, IMDB reviews), we use a batch size of 32 and the AdamW optimizer with learning rate 3e-5 and fine-tune our models for 4 epochs. For MRPC, due to the sensitivity of a smaller dataset, we follow (Devlin et al. 2019) by performing a hyperparameter search with candidate batch size \([8, 16, 32] \) and learning rate \([2e-5, 3e-5, 4e-5, 5e-5] \), and select the best validation result. As these downstream tasks do not exceed the maximum input sequence length 512, we fine-tune our model trained on an input sequence length of 512.

Results. Table 2 presents our experimental results on natu-
r al language understanding benchmarks with different tasks. Our results compares favorably to BERT-base across all downstream tasks. Moreover, we also experiment with fine-tuning our model using longer sequences ($n = 1024$), yet the results remain almost identical to $n = 512$, e.g., 93.0 vs. 93.2 accuracy on IMDB reviews. These results further suggest that our model is able to scale linearly with input length. Additional details on longer sequences is in the supplement and project webpage.

**Conclusion**

It is becoming clear that scaling Transformer based models to longer sequences, desirable in both NLP as well as computer vision, will involve identifying mechanisms to mitigate its compute and memory requirements. Within the last year, this need has led to a number of results describing how randomized numerical linear algebra schemes based on random projections and low rank assumptions can help [Katharopoulos et al. 2020, Wang et al. 2020, Beltagy, Peters, and Cohan 2020, Zaheer et al. 2020]. In this paper, we approach this task differently by showing how the Nyström method, a widely used strategy for matrix approximation, can be adapted and deployed within a deep Transformer architecture to provide an approximation of self attention with high efficiency. We show that our design choices enable all key operations to be mapped to popular deep learning libraries in a convenient way. The algorithm maintains the performance profile of other self-attention approximations in the literature but offers additional benefit of resource utilization. Overall, we believe that our work is a step towards running Transformer models on very long sequences. Our code and supplement is available at our project webpage [https://github.com/mlpen/Nystromformer](https://github.com/mlpen/Nystromformer).

**Acknowledgments**

This work was supported in part by American Family Insurance, NSF CAREER award RI 1252725 and UW CPCP (U54AI117924). We thank Denny Zhou, Hongkun Yu, and Adam Yu for discussions and help with some of the experiments.

**References**

Baker, C. T. 1977. *The numerical treatment of integral equations*. Clarendon press.

Beltagy, I.; Peters, M. E.; and Cohan, A. 2020. Longformer: The Long-Document Transformer. *arXiv preprint arXiv:2004.05150*.

Blanc, G.; and Rendle, S. 2018. Adaptive sampled softmax with kernel based sampling. In *Proceedings of the International Conference on Machine Learning (ICML)*, 590–599.

Bremer, J. 2012. On the Nyström discretization of integral equations on planar curves with corners. *Applied and Computational Harmonic Analysis* 32(1): 45–64.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chen, T.; Xu, B.; Zhang, C.; and Guestrin, C. 2016. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*.

Chen, Z.; Zhang, H.; Zhang, X.; and Zhao, L. 2018. Quora question pairs. URL [https://www.kaggle.com/c/quora-question-pairs](https://www.kaggle.com/c/quora-question-pairs).

Child, R.; Gray, S.; Radford, A.; and Sutskever, I. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Clark, K.; Luong, M.-T.; Le, Q. V.; and Manning, C. D. 2019. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *International Conference on Learning Representations (ICLR)*.

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J. G.; Le, Q.; and Salakhutdinov, R. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2978–2988.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Dolan, W. B.; and Brockett, C. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Drineas, P.; and Mahoney, M. W. 2005. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research (JMLR)* 6(Dec): 2153–2175.

Fanuel, M.; Schreurs, J.; and Suykens, J. A. 2019. Nyström landmark sampling and regularized Christoffel functions. *arXiv preprint arXiv:1905.12346*.

Howard, J.; and Ruder, S. 2018. Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (ACL)*, 328–339.

Huang, Y.; Cheng, Y.; Bapna, A.; Firat, O.; Chen, D.; Chen, M.; Lee, H.; Ngiam, J.; Le, Q. V.; Wu, Y.; et al. 2019. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In *Advances in Neural Information Processing Systems (NeurIPS)*, 103–112.

Katharopoulos, A.; Vyas, A.; Pappas, N.; and Fleuret, F. 2020. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Kitaev, N.; Kaiser, L.; and Levskaya, A. 2019. Reformer: The Efficient Transformer. In *International Conference on Learning Representations (ICLR)*.

Kumar, S.; Mohri, M.; and Talwalkar, A. 2009. Ensemble Nyström method. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1060–1068.

Lample, G.; Sablayrolles, A.; Ranzato, M.; Denoyer, L.; and Jégou, H. 2019. Large memory layers with product keys. In *Advances in Neural Information Processing Systems (NeurIPS)*, 8548–8559.

Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; and Soricut, R. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations (ICLR)*.
Zaheer, M.; Guruganesh, G.; Dubey, A.; Ainslie, J.; Alberti, C.; Ontanon, S.; Pham, P.; Ravula, A.; Wang, Q.; Yang, L.; et al. 2020. Big bird: Transformers for longer sequences. arXiv preprint arXiv:2007.14062.

Zhang, K.; and Kwok, J. T. 2010. Clustered Nyström method for large scale manifold learning and dimension reduction. IEEE Transactions on Neural Networks 21(10): 1576–1587.

Zhang, K.; Tsang, I. W.; and Kwok, J. T. 2008. Improved Nyström low-rank approximation and error analysis. In Proceedings of the International Conference on Machine Learning (ICML), 1232–1239.

Zhu, Y.; Kiros, R.; Zemel, R.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 19–27.