Fourier Transformer: Fast Long Range Modeling by Removing Sequence Redundancy with FFT Operator

Ziwei He$^{\diamond}$, Meng Yang†, Minwei Feng†, Jingcheng Yin†,
Xinbing Wang$^{\diamond}$, Jingwen Leng$^{\diamond}$ and **Zhouhan Lin$^{\diamond *}$**
$^{\diamond}$Shanghai Jiao Tong University † Netease BizEase
{ziwei.he, xwang8, leng-jw}@sjtu.edu.cn   *lin.zhouhan@gmail.com

Abstract

The transformer model is known to be computationally demanding, and prohibitively costly for long sequences, as the self-attention module uses a quadratic time and space complexity with respect to sequence length. Many researchers have focused on designing new forms of self-attention or introducing new parameters to overcome this limitation, however a large portion of them prohibits the model to inherit weights from large pretrained models. In this work, the transformer's inefficiency has been taken care of from another perspective. We propose Fourier Transformer, a simple yet effective approach by progressively removing redundancies in hidden sequence using the ready-made Fast Fourier Transform (FFT) operator to perform Discrete Cosine Transformation (DCT). Fourier Transformer is able to significantly reduce computational costs while retain the ability to inherit from various large pretrained models. Experiments show that our model achieves state-of-the-art performances among all transformer-based models on the long-range modeling benchmark LRA with significant improvement in both speed and space. For generative seq-to-seq tasks including CNN/DailyMail and ELI5, by inheriting the BART weights our model outperforms the standard BART and other efficient models. [1]

1 Introduction

Transformers (Vaswani et al., 2017), especially when equipped with large-scale pre-training (Devlin et al., 2018; Lewis et al., 2019; Raffel et al., 2020) have become the core architecture in most tasks in natural language processing (NLP), including both encoder-only tasks such as sentence classification, sequence tagging (Liu et al., 2019), and encoder-decoder tasks such as text summarization

and question answering (Lewis et al., 2019). However, due to the quadratic complexity of its self-attention module (Lin et al., 2017), applying these models on long sequences can be prohibitively costly. As a result, great efforts have been put into developing various efficient Transformer variants (Tay et al., 2020b), as well as establishing standardized test-beds for long sequences such as the Long Range Arena (LRA) (Tay et al., 2020a).

Most efficient Transformers devise special attention variants to lower its complexity (Tay et al., 2020b). Some of them achieve this by projecting components in self-attention into its lower-rank approximations (Wang et al., 2020; Zhu et al., 2021; Winata et al., 2020, *inter alia*), or rely on kernelization to implicitly compute the attention matrix (Katharopoulos et al., 2020; Choromanski et al., 2020b; Peng et al., 2021; Choromanski et al., 2020a, *inter alia*).

Due to the introduction of projection matrices or extra parameters, these models are not able to inherit pre-trained model parameters. However, since pre-trained large language models (LLMs) have fundamentally influenced the NLP community, deviating model architecture from LLMs requires pre-training from scratch on the designed model, which is prohibitively resource-demanding for most practitioners.

Other approaches target at computing part of the attention matrix, by following some predefined patterns (Child et al., 2019; Qiu et al., 2020; Ho et al., 2019, *inter alia*). Some of them allow the pattern to be learnable (Sukhbaatar et al., 2019; Roy et al., 2021, *inter alia*). Most of the patterns require customized CUDA kernels or special operators to achieve the claimed speedup (Wu et al., 2019; Child et al., 2019; Beltagy et al., 2020), which casts extra challenge in deploying these models on edge devices or special hardware such as TPUs. Moreover, some of the approaches involve considerable additional computation steps, which in

---

practice could counterweight the time and memory complexity they reduce, especially for short and medium-length sequences (Kitaev et al., 2020; Roy et al., 2021).

One core factor behind various approaches is the existence of redundancy in attention matrices and hidden states. For example, Wang et al. (2020) provides spectrum analysis on the self-attention matrix, indicating that the attention matrix learns to be low-rank, which allows them to learn a low-rank approximation of the attention matrix. Inspired by this line of research, in this work, we analyze the power spectrum of the hidden states in the time dimension through different layers in Fig 1, and show that the power spectrum increasingly concentrates on lower frequency bins as the layer gets deeper.

In this work, we propose Fourier Transformer, which doesn't even require to learn the projection matrix in order to approximate the self-attention. Fourier Transformer leverages our observation on power spectra of hidden states, it progressively removes sequence redundancies through different layers by downsampling hidden states with the Discrete Cosine Transform (DCT), a variant of Fourier transform that generates real values.

The DCT in our proposed Fourier Transformer can be implemented with the Fast Fourier Transform (FFT) operator. Thanks to its profound application in image compression and signal processing, it is one of the most widely available and highly optimized operators in a wide variety of frameworks and even on edge devices, providing $O(n \log n)$ complexity and up to $O(\log n)$ in parallel implementations with negligible overhead. As a result, Fourier Transformer is easily deployable on a wide range of devices, not necessary to devise special CUDA kernels. In addition, experimental results on LRA tasks show that it performs significantly faster than many other efficient Transformers, while achieving the state-of-the-art performance among Transformer-based efficient models.

On the other hand, since DCT is a linear, reversible transformation, and the self-attention is not interfered in our model, the proposed Fourier Transformer can inherit pretrained weights from large language models without hurting performance. Experimental results on CNN-DailyMail (Hermann et al., 2015) and ELI5 (Fan et al., 2019c) show that our model could outperform BART (Lewis et al., 2019) and other efficient Transformers by inheriting and fine-tuning on BART. Moreover, with tiny amount of further pretraining before fine-tuning, its performance could be further improved.

2 Related Work

Downsampling hidden states There are not many work that downsample sequence length for natural language. The closest work is Funnel Transformer (Dai et al., 2020), which progressively reduces the *query* sequence length through strided mean pooling, while keeping *key* and *value* sequence lengths intact. Fourier Transformer compresses the three sequences altogether and delivers more computational speedup compared with Funnel Transformer. Note that Funnel Transformer needs to re-invest the saved computations to build a larger model to achieve better performance, which disables its ability to inherit pretrained weights. For other work, Charformer (Tay et al., 2021b) devises a differentiable tokenization module that also relies on strided mean pooling to downsample its byte sequence. Nyströmformer (Xiong et al., 2021) approximates the attention matrix through the Nyström method, which effectively downsamples *query* and *key* sequences. Due to the extra depth-wise convolution, it is again not able to leverage pretrained models.

In a border view, downsampling has been more favorable in computer vision. (Chen et al., 2020) aggressively downsamples the raw input to a 1D vector. Perceiver (Jaegle et al., 2021) adopts an asymmetric attention mechanism to distill inputs into a tight latent bottleneck. Almost all of these vision models are designed for encoder-only vision tasks rather than encoder-decoder-style NLP tasks.

Fourier transform for Transformer There are multiple recent works that incorporate Fourier transform into Transformer. FNet (Lee-Thorp et al., 2021) takes a more radical approach by replacing the entire self-attention with 2D FFT, discarding the entire imaginary part to avoid complex numbers. Performer (Choromanski et al., 2020a) introduced orthogonal random Fourier features to approximate the softmax attention. FSAT (Zhuang et al., 2022) uses 1D FFT along the sequence dimension to learn the sparse structure of attention matrix. DCT-Former (Scribano et al., 2022) translates sequences into frequency domain and conducts self-attention there before projecting them back, due to the non-linearity in the network, self-attention trained in the frequency domain significantly deviates from that in the time domain. Therefore, all the models dis-
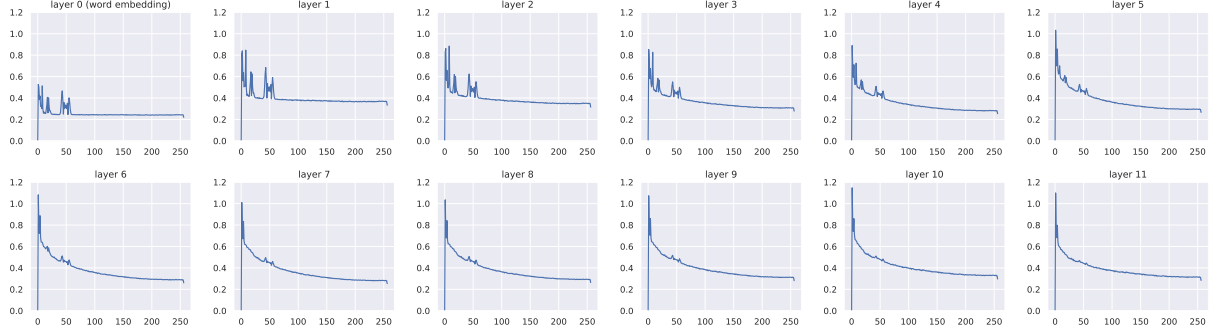
Figure 1: The power spectrum of input hidden states from different layers in the pretrained RoBERTa (Liu et al., 2019) model. The horizontal axes stand for frequency bins, starting from low frequency components on the left. The vertical axes are the corresponding amplitudes. Amplitudes are averaged over all hidden dimensions and over the entire validation set of Wiki-103 (Merity et al., 2016). Since the inputs are real numbers, the positive and negative frequency components are pairwise conjugate. Thus we only plot the amplitude of the positive half of the frequencies.

cussed above lose the ability to inherit pretrained weights as well.

3 Preliminaries

3.1 Discrete Cosine Transform

The Discrete Cosine Transform (DCT) expresses a sequence of real numbers in terms of a sum of cosine functions with different frequencies. Since DCT only yields real values, it is a substitution for Fourier transform in the field of real numbers. It has been the core transform behind the JPEG [2] lossy image compression format.

Formally, for a sequence of N real numbers $\{x_n\} = \{x_0, x_1, ...x_{N-1}\}$, DCT transforms it into frequency domain through[3]:

$$y_k = \alpha_k \sum_{n=0}^{N-1} x_n cos\left(\frac{\pi k(2n+1)}{2N}\right) \quad (1)$$

where $k \in \{0, ..., N-1\}$ and α_k is an coefficient related to k:

$$\alpha_k = \begin{cases} \sqrt{\frac{1}{N}} & if \ k = 0, \\ \sqrt{\frac{2}{N}} & otherwise \end{cases} \quad (2)$$

The original sequence $\{x_n\}$ can be recovered with the inverse DCT (IDCT):

$$x_n = \alpha_k \sum_{k=0}^{N-1} y_k cos\left(\frac{\pi k(2n+1)}{2N}\right) \quad (3)$$

which we'll note as $\{x_n\} = IDCT(\{y_k\})$.

---

[2]https://jpeg.org/jpeg/
[3]There are several slightly different variants of DCT. Here we use the most common type-II variant in this paper.

Practically, DCT can be computed by using the FFT operator. First, let $\{u_n\}$ be the shuffled $\{x_n\}$ by interleaving its values on even and odd positions. Formally, when N is an odd integer, $\{u_n\}$ is given by

$$\{u_n\} = \{x_0, x_2, ..., x_{N-1}, x_{N-2}, x_{N-4}, ..., x_1\} \quad (4)$$

When N is even, a similar shuffling applies. We then transform $\{u_n\}$ into its frequency domain through FFT:

$$\{v_k\} = FFT(\{u_n\}) \quad (5)$$

where $k \in \{0, ..., N-1\}$ and $\{v_k\}$ is a sequence of length N. The DCT of the original sequence $\{x_n\}$ can thus be computed from $\{v_k\}$:

$$y_k = cos\left(\frac{\pi k}{2N}\right) \mathfrak{Re}\left(v_k\right) - sin\left(\frac{\pi k}{2N}\right) \mathfrak{Im}\left(v_k\right) \quad (6)$$

where $\mathfrak{Re}\left(\cdot\right)$ and $\mathfrak{Im}\left(\cdot\right)$ stand for the real and imaginary part, respectively.

3.2 The Power Spectrum of Transformer Hidden States

The power spectrum of a discrete sequence describes the distribution of signal power w.r.t. frequency components, which is the amplitudes of frequency components yielded by the Fourier transform. For a certain layer in Transformer, its hidden states can be considered as a sequence of hidden vectors, along the time dimension. To analyze the power spectrum of the layer, we conduct 1D Fourier transform independently along the time
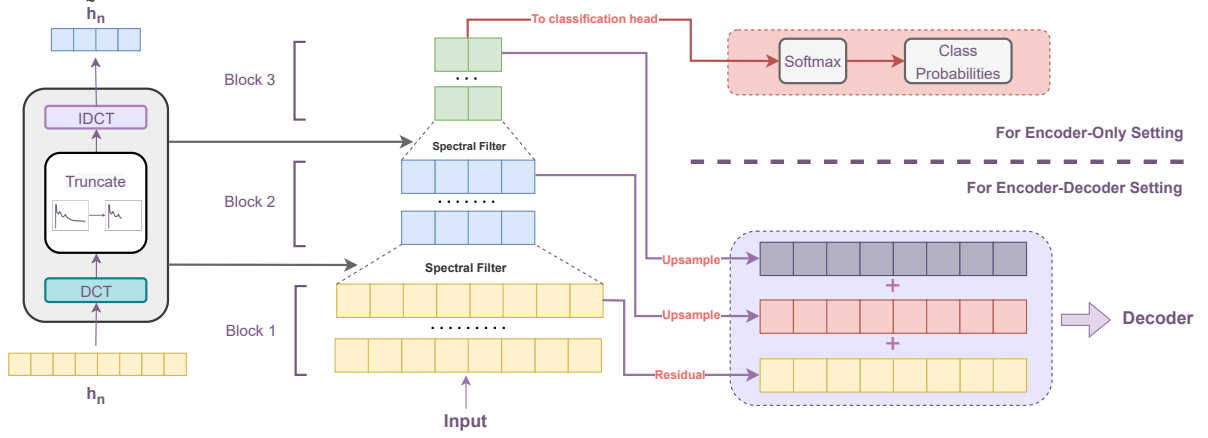
Figure 2: Overall Model Architecture

dimension for the hidden vectors, calculate the corresponding amplitudes, and avreage over all dimensions in that layer. In addition, we calculate the mean spectrum over many text sequences to eliminate example-wise noise.

Figure 1 shows the power spectra for different layers in the pre-trained RoBERTa-base (Liu et al., 2019) model. The up-left subfigure shows that the power spectrum of word embeddings is relatively flat, distributing its energy almost uniformly on all frequency components with several spikes in low frequencies. As the layer gets deeper, the energy starts to concentrate toward low frequencies and the spikes start to smooth out, leaving a long tail on the high-frequency side. This trend indicates that the hidden states in deeper layers are more locally correlated, which leaves space for Fourier transform to squeeze out the redundancies.

4 Fourier Transformer

4.1 Model Architecture

The overall architecture of the Fourier Transformer is depicted in Figure 2. In general, we insert *spectral filters* between layers in Transformer, inside which we use DCT and IDCT to downsample sequence lengths. Multiple spectral filters can work together to split Transformer layers into different blocks, thus progressively reduce sequence lengths. We leave the self-attention intact in order to retain its ability to inherit pretrained weights.

As for the spectral filter, it consists of three steps, i.e., *transform*, *truncate*, and *reverse*. Formally, for an incoming hidden sequence $\{\boldsymbol{h}_n\}, 0 < n < N - 1$ that contains N hidden vectors $\boldsymbol{h}_n \in \mathbb{R}^D$ where D is the hidden size of the model, the spectral filter

first transforms it into frequency domain through 1D-DCT:

$$\{\boldsymbol{y}_k\} = DCT(\{\boldsymbol{h}_n\}), \qquad 0 < k < N - 1 \quad (7)$$

Note that the DCT is independently applied on all dimension in $\{\boldsymbol{h}_n\}$, therefore only transforming along the time dimension.

Next, $\{\boldsymbol{y}_k\}$ is truncated by chopping off the trailing dimensions on the high frequency side. For sequences of different lengths, we fix a ratio $r \in (0, 1)$, which is a hyperparameter, to determine the number of frequency components to retain. Thus the length of $\{\boldsymbol{y}_k\}$ is truncated from N into $\lceil rN \rceil$. [4]

Finally, the resulting shorter sequence $\{\boldsymbol{y}_k\}, 0 < k < \lceil rN \rceil - 1$ can be transformed back to time domain through IDCT, yielding a shorter sequence of $\{\tilde{\boldsymbol{h}}_n\}$:

$$\{\tilde{\boldsymbol{h}}_n\} = IDCT(\{\boldsymbol{y}_k\}), \qquad 0 < n < \lceil rN \rceil - 1 \quad (8)$$

Again, IDCT is also conducted in the time dimension only. The resulting shorter hidden states are passed towards upper layers.

Depending on the type of tasks, the subsequent parts differs. We'll elaborate them in encoder-only and encoder-decoder settings.

Encoder-Only Setting For encoder-only tasks such as text classification, the final output of the

---

[4]we've played with various ways of truncation here, such as cutting off the low-frequency components, cutting off the ones with lower mean amplitudes, removing the DC components and re-normalize the sequence, subtracting a common value on all components and re-normalize, or retaining the components corresponding to the spikes. Interestingly, the rather classical way of simply chopping off high frequency ones turns out to work the best.

encoder is expected to be a fixed-size vector, which is then fed into logistic regression for class probability predictions. In this work, while the model is trained from scratch, we simply use a mean pooling over the whole output sequence to yield this vector; otherwise when the model inherits a `[CLS]` token from pretrained models, we use the embedding at that token instead.

Encoder-Decoder Setting For language generation tasks that involve both an encoder and a decoder, there is an encoder-decoder attention that attends to the encoder states at each decoder step. However, the encoder-decoder attention requires fine-grained positional resolution in order to work well. As a result we follow Dai et al. (2020) to up-sample the shorter sequences back to their original length, and add the upsampled hidden sequences at all blocks together before feeding them to the decoder. More specifically, we use the parameter-free nearest neighbor interpolation for upsampling, and we re-normalize the sequence after adding the upsampled sequences.

4.2 Further Pretraining

Since the DCT is reversible through IDCT, the proposed model seamlessly approximates the vanilla Transformer as r goes up. Figure 3 shows that while fine-tuning directly on BART (Lewis et al., 2019) weights, the model performs comparatively well when up to 70% frequency components are truncated. Nevertheless, since the upsampling and addition of upsampled sequences still differs from the original Transformer, we can still squeeze the last drop out by applying a tiny amount of further pretraining before fine-tuning, and further improve the model performance. This type of further pretraining is much more favourable than a customized pretraining from scratch, which could take massive amount of computation resources.

As a concrete example, further pretraining our model on BART-Large consumes around 10GB of data and takes around 4 days on 2 NVidia A100 GPUs, while pretraining BART from scratch needs to consume 160GB data, taking roughly 1000 days with the same devices. Compared to a customized pre-training from scratch, leveraging BART weights and further pretraining takes 2 magnitudes less computation resources, while still able to bring the model to similar or even better performance.

4.3 Complexity Analysis

For a standard Transformer layer with model dimension D, which consists of self-attention and 2 feed-forward layers, the time and memory complexity of processing an input sequence with length N is $O(N^2D + ND^2)$ and $O(N^2 + ND)$, respectively. With FFT operator our model could compress the sequence length from N to $\lceil rN \rceil$ within $O(N \log N)$ time complexity. Hence the Fourier Transformer enjoys time and memory complexity of $O(r^2N^2D + rND^2 + N \log N)$ and $O(r^2N^2 + rND)$ every time the sequence length is reduced. Actually, given the parallel implementation of FFT, the additional $O(N \log N)$ time complexity term is negligible compared to the other two terms. The speedup could get even more impressive when the sequence length is relatively long. We refer the readers to Section 5.1 for more details.

5 Experiments

In this section, we experiment with our model in both of the two encoder-only and encoder-decoder settings in various datasets that involves long sequences.

5.1 Encoder-only Tasks

To test our model's ability on encoder-only tasks, we choose the 5 tasks in the widely-used Long Range Arena (LRA) benchmark (Tay et al., 2020a). LRA is designed for evaluating efficient transformers under long-context scenario, with the input sequence lengths ranging from 1K to 8K. The datasets in LRA come from rich sources, including natural languages, image pixels, math expressions etc. More specifically, they are:

ListOps A dataset of math expressions that asks the model to calculate the output value of a math expression with sequence lengths up to 2K.

Text A byte-level text classification task, with a fixed sequence length 4K which requires the model to deal with compositionality.

Retrieval A byte-level document retrieval task with a maximum length of 8K which test the model's ability to compress long sequences.

Image An image classification task of which requires the model to learn the 2D spatial relations between input pixels by sequentially reading the pixels. The sequence length is fixed to 1K.

Models	ListOps	Text	Retrieval	Image	Pathfinder	Avg.
Transformer (Vaswani et al., 2017)	36.37	64.27	57.46	42.44	71.40	54.39
Longformer (Beltagy et al., 2020)	35.63	62.85	56.89	42.22	69.71	53.46
Linformer (Wang et al., 2020)	35.70	53.94	52.27	38.56	76.34	51.36
Reformer (Kitaev et al., 2020)	37.27	56.10	53.40	38.07	68.50	50.67
Synthesizer (Tay et al., 2021a)	36.99	61.68	54.67	41.61	69.45	52.88
BigBird (Zaheer et al., 2020)	36.05	64.02	59.29	40.83	74.87	55.01
Performer (Choromanski et al., 2020a)	18.01	65.40	53.82	42.77	77.50	51.41
FNet (Lee-Thorp et al., 2021)	35.55	65.11	59.61	38.67	77.80	55.30
Nyström (Xiong et al., 2021)	37.15	65.52	79.56	41.58	70.94	58.95
Luna-256 (Ma et al., 2021)	37.25	64.57	79.29	47.38	77.32	61.24
FSAT (Zhuang et al., 2022)	**46.85**	65.95	81.11	49.97	77.32	64.24
Fourier Transformer (ours)	40.73	**75.02**	**85.35**	**53.17**	**83.43**	**67.54**

Table 1: The results on LRA benchmark. We report classification accuracy for each task and average accuracy across all tasks. Results from Longformer to Performer are from Tay et al. (2020a), the rest are fetched from their respective papers. For FSAT model on Text task, we only consider the result without convolutions.

	Steps per second ↑				Peak Memory Usage ↓			
Model	1K	2K	3K	4K	1K	2K	3K	4K
Transformer	1.0x	1.0x	1.0x	1.0x	1.0x	1.0x	1.0x	1.0x
Reformer	0.5x	0.4x	0.7x	0.8x	0.56x	0.37x	0.28x	0.24x
BigBird	0.9x	0.8x	1.2x	1.1x	0.91x	0.56x	0.4x	0.3x
Synthesizer	1.1x	1.2x	2.9x	1.4x	0.76x	0.75x	0.74x	0.74x
FSAT	1.1x	1.5x	2x	2.5x	0.53x	0.27x	0.21x	0.16x
Linformer	1.2x	1.9x	3.7x	5.5x	0.44x	0.21x	0.18x	**0.1**x
Performer	1.2x	1.9x	3.8x	5.7x	0.44x	0.22x	**0.15**x	0.11x
Fourier Transformer (ours)	**6.9**x	**12.2**x	**16.8**x	**17.7**x	**0.23**x	**0.19**x	0.18x	0.18x

Table 2: The speed and memory consumption on LRA benchmark over **Text** task with input lengths of 1K, 2K, 3K and 4K. The results from Reformer to Performer are from Zhuang et al. (2022). The speed and memory consumption are listed as the rate w.r.t. the vanilla Transformer.

Pathfinder An synthetic image classification task with a fixed input length of 1K which requires the model to capture long-range spatial dependencies.

5.1.1 Implementation Details

We run experiments on the LRA benchmark closely following the configurations in (Tay et al., 2020a), including data pre-processing, data split, model architecture, hyperparameters (number of layers, hidden dimensions, etc.). We evaluate in terms of classification accuracy. Our implementation is based on (Xiong et al., 2021). For the sake of simplicity, we report the results of our model over the five tasks with the same compression budget. We aggressively reduce 80% of the input sequence

length at the first layer.

5.1.2 Performance & Efficiency

The results on the aforementioned 5 tasks are summarized in Table 1. We compare Fourier Transformer with a bunch of previously published Transformer-based models, and it achieves new state-of-the-art results on four out of the five tasks. Our proposed model improves over the previous SOTA model (Zhuang et al., 2022) on *Text*, *Retrieval*, *Image* and *Pathfinder* by 9.07%, 4.24%, 3.20%, 6.11% absolute value respectively, which is a big margin. Notably, our model doesn't beat FSAT (Zhuang et al., 2022) on the ListOps task and ranks the 2nd in the list. We conjecture that it's because math expression values are more sensitive

to individual tokens in the sequence, thus is more sensitive to downsampling.

Next, taking the byte-level text classification task (the Text dataset) as a testbed, we quantitatively evaluate the time and memory efficiency of our model and the other competing models on various input lengths. The results are summarized in Table 2. Note that, due to the limitation of GPU memory for the vanilla Transformer, results on 1K, 2K and 3K lengths are run with a batch size of 32, and 4K are with a batch size of 16. We calculate the corresponding rates of our model w.r.t. vanilla Transformer on identical batch settings, and timed on an NVidia A100-80G GPU. Compared with other efficient transformers, Fourier Transformer significantly reduces time consumption on both short and long sequences, leaving the other model behind by a large margin, while keeping a steady memory savings as the sequence length grows.

5.2 Encoder-Decoder Tasks

The model for encoder-decoder tasks are equipped with a decoder to perform text generation. For this setting, we choose two long-text datasets in summarization and question answering tasks, i.e., CNN/DailyMail (Hermann et al., 2015) and ELI5 (Fan et al., 2019c), with average sequence lengths at 0.8K and 5K, respectively.

CNN/DailyMail A summarization dataset containing over 280K news articles (766 token counts on average) from news stories in CNN and Daily Mail websites paired with human-generated summaries (53 token counts on average). We follow the conversion and evaluate the performance in terms of Rouge scores (Rouge-1, Rouge-2, Rouge-L) (Lin, 2004).

ELI5 A question answering dataset containing over 270K complex, diverse and paragraph-length question-answer pairs gathered from subreddits, the average number of tokens for input and target are 5140 and 693 respectively. Following the conversion, we evaluate it in both Rouge-L and F1 scores.

5.2.1 Implementation Details

Since on both the two datasets pretrained models leave a large gap over non-pretrained ones, it makes less sense to report results without pretraining. Thus, we report results of our model inheriting BART-large (Lewis et al., 2019) weights. We generally test two settings, which is: 1) directly fine-tune

our model on the dataset, and 2) conduct further pretraining before fine-tuning. For convenience, we call them *Fourier-BART* and *Fourier-BART-FP* respectively in the rest of the paper.

Fourier-BART has the same architecture as BART-large. It simply adopts a 2-block design, the first block contains the first 2 consecutive transformer layers, the rest 10 layers belong to the second block. For CNN/DailyMail, 50% of the frequency components are truncated, while for ELI5 70% are truncated since it has much longer sequence lengths.

Fourier-BART-FP has the same setting as Fourier-BART, except that before fine-tuning on downstream tasks it is further pretrained for 1 epoch on 10GB of text with the original BART pretraining objectives. The text is randomly sliced from the Pile (Gao et al., 2020) corpus.

5.2.2 Performance & Efficiency

CNN/DailyMail On summarization task, inside the scope of efficient models, we compare our model with BigBird (Zaheer et al., 2020), ST-MoE (Zoph et al., 2022) and Switch Transformer (Fedus et al., 2021), which are strong baselines from recent literature. Both ST-MoE and Switch Transformer targeted at activating only part of the parameters to improve the efficiency. Bigbird approximates full attention matrix with a sparse one to improve on FLOPs. In addition, we put the standard BART (Lewis et al., 2019) performance as baseline.

The results are listed in Table 3. Our proposed Fourier-BART successively leverages the advantage of BART, achieving a performance at the level of pretrained model. With the tiny amount of further pretraining, it achieves the best performance among all competitors. Note that Fourier-BART is built upon BART and sharing the same model size with BART-400M with much less computation, however it is able to outperform the standard BART-400M with a sensible margin.

As for efficiency, it is almost impossible to reproduce all the models listed in Table 3 and investigate their efficiency, so we choose to only evaluate the standard BART-400M and proposed Fourier-BART-400M in terms of FLOPs. As elaborated in Section 5.2.1, we remove 50% from the hidden sequence on the third transformer layer, although the two models have the exact same size, the FLOPs invested in the standard BART-400M is 1.6 times of Fourier-BART-400M. Due to the upsampling and the auto-regressive decoding, the overall reduction

Model	R-1	R-2	R-L
BART-400M	44.16	21.28	40.90
ST-MOE-L-770M	-	20.7	-
Switch Trans.-223M	-	19.6	-
BigBird-Large	43.84	21.11	40.74
Fourier-BART-400M	44.65	21.48	41.30
Fourier-BART-FP-400M	**44.76**	**21.55**	**41.34**

Table 3: Rouge scores on CNN/DailyMail. The results are all fetched from their respective papers. The R-1 and R-L of ST-MOE and Switch Transformer are not reported in their paper. The number after model name denotes the model size. The model size for BigBird is not mentioned in their paper unfortunately.

Model	RL	F1
LayerDrop-240M	23.4	-
E-MCA-240M	24.0	-
c-REALM*-596M	23.2	22.9
EMAT*-446M	20.91	19.03
KID*-406M	26.3	-
BART-large-400M	26.8	26.6
Fourier-BART-400M	26.2	25.98
Fourier-BART-FP-400M	**26.9**	**26.73**

Table 4: Model performance on ELI5. The results from E-MCA to KID are fetched from their respective papers. * denotes results using the Kilt benchmark (Petroni et al., 2020), which has smaller dev and test sets.

in computation is not as significant as those on LRA.

ELI5   On question answering task, we compare our model with the LayerDrop (Fan et al., 2019b), E-MCA (Fan et al., 2019a), c-REALMS (Krishna et al., 2021), EMAT (Wu et al., 2022) and KID (Liu et al., 2022). To provide a fair comparison, the result of BART-large is our reproduced one on the bleeding-edge version of fairseq (Ott et al., 2019), which is much higher than the results reported in the original BART paper. Note that here we are even comparing with performance-sensitive models, as in the list only EMAT and LayerDrop are focusing on reducing complexity. As shown in Table 4, our Fourier-BART-FP has surpassed all the competing models on both Rouge-L and F1 scores.

As for efficiency, when removing 70% of the frequency components (elaborated in Section 5.2.1), the FLOPs invested in the standard BART is 1.9 times of Fourier-BART.
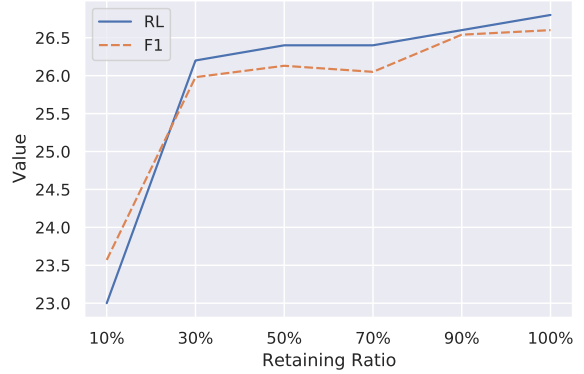

Figure 3: R1, R2, RL and F1 on ELI5. x-axis stands for the retraning ratio r.

5.3   Analysis on Retaining Ratio r

An important question that arises is how sensitive the model is w.r.t. the ratio of retaining frequency components. To investigate this, we experiment our model in ELI5 dataset. by sweeping r from 0.1 to 1. We didn't conduct further pretraining on each setting due to computation limit. Results are shown in Fig 3. The performance remains pretty good up until less than 30% of frequency components are retained. When we try to truncate more components passing that ratio, the performance starts to drop significantly. This is a fairly satisfying result that shows the model performs reliably stable in a wide range of reasonable r's.

6   Conclusion

In this work, we introduce the discrete cosine transformation to progressively downsample the hidden states in the Transformer model by leveraging the local correlations between hidden states in upper layers. Our approach is able to significantly reduce the computation required by the vanilla Transformer, while being able to achieve even better performance in various tasks. Moreover, it is able to inherit the pretrained model weights, which is an notable advantage over most efficient Transformers.

7   Limitations

Although our approach exhibits great speedups in encoder-only settings, it doesn't yield as impressive speedups in encoder-decoder setting. This is due to the autoregresive decoding steps in the decoder, that has to be conducted sequentially. Accelerating that with DCT requires to incrementally update DCT outputs step by step based on outputs of pre-

vious timesteps, which is theoretically possible but not easy to optimize its efficiency. We plan to further accelerate it in this direction in future work.

Acknowledgement

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. 2020. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, David Belanger, Lucy Colwell, et al. 2020a. Masked language modeling for proteins via linearly scalable long-context transformers. *arXiv preprint arXiv:2006.03555*.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. 2020b. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*.

Zihang Dai, Guokun Lai, Yiming Yang, and Quoc Le. 2020. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. *Advances in neural information processing systems*, 33:4271–4282.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2019a. Using local knowledge graph construction to scale seq2seq models to multi-document inputs. *arXiv preprint arXiv:1910.08435*.

Angela Fan, Edouard Grave, and Armand Joulin. 2019b. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019c. ELI5: long form question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3558–3567. Association for Computational Linguistics.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. 2019. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. 2021. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. *arXiv preprint arXiv:2103.06332*.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2021. Fnet: Mixing tokens with fourier transforms. *arXiv preprint arXiv:2105.03824*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Ruibo Liu, Guoqing Zheng, Shashank Gupta, Radhika Gaonkar, Chongyang Gao, Soroush Vosoughi, Milad Shokouhi, and Ahmed Hassan Awadallah. 2022. Knowledge infused decoding. *arXiv preprint arXiv:2204.03084*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. 2021. Luna: Linear unified nested attention. *Advances in Neural Information Processing Systems*, 34:2441–2453.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A Smith, and Lingpeng Kong. 2021. Random feature attention. *arXiv preprint arXiv:2103.02143*.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, et al. 2020. Kilt: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252*.

Jiezhong Qiu, Hao Ma, Omer Levy, Wen-tau Yih, Sinong Wang, and Jie Tang. 2020. Blockwise self-attention for long document understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2555–2565.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.

Carmelo Scribano, Giorgia Franchini, Marco Prato, and Marko Bertogna. 2022. Dct-former: Efficient self-attention withdiscrete cosine transform. *arXiv preprint arXiv:2203.01178*.

Sainbayar Sukhbaatar, Édouard Grave, Piotr Bojanowski, and Armand Joulin. 2019. Adaptive attention span in transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 331–335.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2021a. Synthesizer: Rethinking self-attention for transformer models. In *International conference on machine learning*, pages 10183–10192. PMLR.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2020a. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020b. Efficient transformers: A survey. *ACM Computing Surveys (CSUR)*.

Yi Tay, Vinh Q Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. 2021b. Charformer: Fast character transformers via gradient-based subword tokenization. *arXiv preprint arXiv:2106.12672*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Genta Indra Winata, Samuel Cahyawijaya, Zhaojiang Lin, Zihan Liu, and Pascale Fung. 2020. Lightweight and efficient end-to-end speech recognition using low-rank transformer. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6144–6148. IEEE.

Felix Wu, Angela Fan, Alexei Baevski, Yann N Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. *arXiv preprint arXiv:1901.10430*.

Yuxiang Wu, Yu Zhao, Baotian Hu, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2022. An efficient memory-augmented transformer for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2210.16773*.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. 2021. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14138–14148.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297.

Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. 2021. Long-short transformer: Efficient transformers for language and vision. *Advances in Neural Information Processing Systems*, 34:17723–17736.

Yimeng Zhuang, Jing Zhang, and Mei Tu. 2022. Long-range sequence modeling with predictable sparse attention. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 234–243.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. Designing effective sparse expert models. *arXiv preprint arXiv:2202.08906*.