Improving Transformer with an Admixture of Attention Heads

Anonymous Author(s)
Affiliation
Address
email

Abstract

Transformers with multi-head self-attention have achieved remarkable success in sequence modeling and beyond. However, they suffer from high computational and memory complexities for computing the attention matrix at each head. Recently, it has been shown that those attention matrices lie on a low-dimensional manifold and, thus, are redundant. We propose the Transformer with a Finite Admixture of Shared Heads (FiSHformers), a novel class of efficient and flexible transformers that allow the sharing of attention matrices between attention heads. At the core of FiSHformer is a novel finite admixture model of shared heads (FiSH) that samples attention matrices from a set of global attention matrices. The number of global attention matrices is much smaller than the number of local attention matrices generated. FiSHformers directly learn these global attention matrices rather than the local ones as in other transformers, thus significantly improving the computational and memory efficiency of the model. We empirically verify the advantages of the FiSHformer over the baseline transformers in a wide range of practical applications including language modeling, machine translation, and image classification. On the WikiText-103, IWSLT'14 De-En and WMT'14 En-De, FiSHformers use much fewer floating-point operations per second (FLOPs), memory, and parameters compared to the baseline transformers.

1 Introduction

Transformers have become the state-of-the-art model for solving many challenging problems in natural language processing [67, 1, 16, 72, 20, 9, 30, 53] and computer vision [18, 60, 21, 65]. Transformers learn from unlabeled data effectively and take advantage of the pre-trained models on downstream tasks that involve different data modalities with limited supervision [51, 52, 20, 73, 38]. The success of transformer is credited to the multi-head self-attention (MHA) mechanism as their fundamental building block. For each token in the sequence, self-attention in transformers aggregates information from other tokens by computing a weighted average of their feature representations with a weight proportional to a similarity score between the representations. This attention mechanism allows arbitrary input-dependent interaction between tokens in the sequence where a token can pay attention to other tokens and attain a contextual representation [5, 67, 33]. Multi-head self-attention captures multiple such contextual representations, one at each head, thereby increasing the representation capacity of the self-attention. It has been argued that the representation capacity of the attention mechanism [64] and its flexibility in capturing diverse syntactic and semantic relationships [64, 68, 14, 69, 28] account for the impressive performance of transformers in practice.

1.1 Background: Self-Attention

For a given input sequence $\mathbf{X} := [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]^\top \in \mathbb{R}^{N \times D_x}$ of N feature vectors, self-attention transforms $\mathbf{X}$ into the output sequence $\mathbf{H}$ in the following two steps:

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

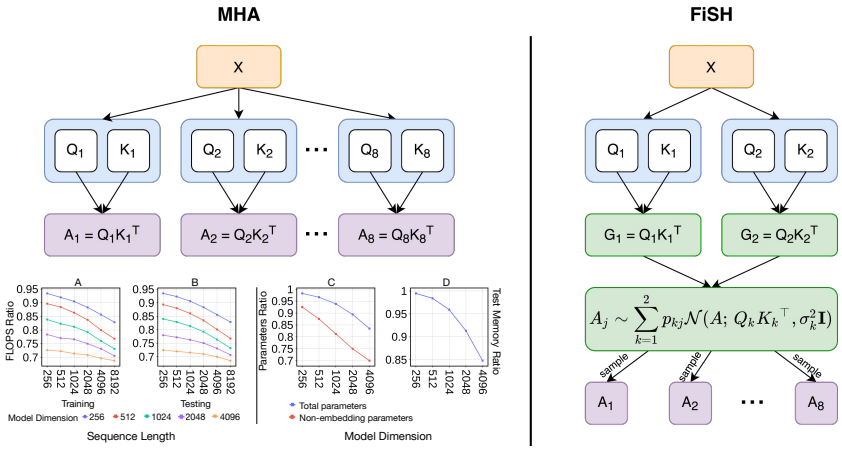

Figure 1: Our proposed finite admixture model of shared heads (FiSH) vs. the standard multi-head (MHA) attention. FiSH samples local attention matrices from a finite admixture of global attention matrices. Compared to MHA, FiSH is more efficient, saving computation and memory (See Fig. 3 and Section 4).

Step 1. The input sequence $\mathbf{X}$ is projected into the query matrix $\mathbf{Q}$, the key matrix $\mathbf{K}$, and the value matrix $\mathbf{V}$ via three linear transformations

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q^\top; \mathbf{K} = \mathbf{X}\mathbf{W}_K^\top; \mathbf{V} = \mathbf{X}\mathbf{W}_V^\top,$$

where $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{D\times D_x}$, and $\mathbf{W}_V \in \mathbb{R}^{D_v\times D_x}$ are the weight matrices. We denote $\boldsymbol{Q} := [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_N]^\top, \mathbf{K} := [\boldsymbol{k}_1, \cdots, \boldsymbol{k}_N]^\top$, and $\mathbf{V} := [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N]^\top$, where the vectors $\boldsymbol{q}_i, \boldsymbol{k}_i, \boldsymbol{v}_i$ for $i = 1, \cdots, N$ are the query, key, and value vectors, respectively.

Step 2. The output sequence $\mathbf{H} := [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_N]^\top$ is then computed as follows

$$\mathbf{H} = \mathrm{softmax}\Big(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D}}\Big)\mathbf{V} := \mathrm{softmax}(\frac{\mathbf{A}}{\sqrt{D}})\mathbf{V}, \tag{1}$$

where the softmax function is applied to each row of the matrix $\mathbf{A} = (\mathbf{Q}\mathbf{K}^\top)$. This matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ and its component a_{ij} for $i,\ j = 1, \cdots, N$ are called the attention matrix and attention scores, respectively. For each query vector $\boldsymbol{q}_i$ for $i = 1, \cdots, N$, an equivalent form of Eqn. (1) to compute the output vector $\boldsymbol{h}_i$ is given by

$$\boldsymbol{h}_i = \sum_{j=1}^{N} \mathrm{softmax}\Big(\boldsymbol{q}_i^\top \boldsymbol{k}_j/\sqrt{D}\Big)\boldsymbol{v}_j. \tag{2}$$

The self-attention computed by Eqn. (1) and (2) is called the scaled dot-product or softmax attention. In our paper, we call a transformer that uses this attention the softmax transformer. The structure that the attention matrix $\mathbf{A}$ learns from training determines the ability of the self-attention to capture contextual representation for each token.

Multi-head Attention (MHA) Each output sequence $\mathbf{H}$ forms an attention head. In MHA, multiple heads are concatenated to compute the final output. Let H be the number of heads and $W^O \in \mathbb{R}^{HD\times HD}$ be the projection matrix for the output. The multi-head attention is defined as

$$\mathrm{MultiHead}(\{\mathbf{H}\}_{i=1}^H) = \mathrm{Concat}(\mathbf{H}_1, \ldots, \mathbf{H}_H)\mathbf{W}^O. \tag{3}$$

1.2 Eigenvalue Analysis of the Attention Matrices

The multi-head mechanism allows transformers to capture more diverse attention patterns and increase the capacity of the model. However, in many practical tasks, transformers learn redundant heads [42, 70], whose learned attention matrices lie on a low-dimensional manifold [7]. To confirm this claim, in Figure 2, we follow the eigenvalue analysis in [7] and investigate the eigenvalues of the covariance matrix of vectorized attention matrices aggregated over each layer of a transformer model trained on the WikiText-103

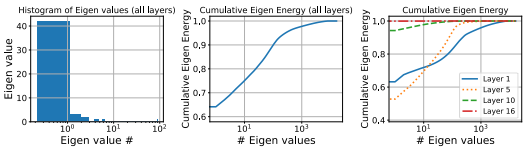

Figure 2: (Left) Histogram of the top 50 eigen values and (Middle, Right) cumulative sum of eigen values of the layer-average attention scores covariance matrix.

dataset for language modeling. We observe that this covariance matrix is low rank with top 200 (1.2%) eigenvalues capturing more than 90% of the energy. This result verifies that the variability of learned attention matrices in transformers can be explained by a relatively small number of principal components, and those attention matrices lie on a low-dimensional manifold. Therefore, in multi-head attention, the effective number of heads is much smaller than the actual number of heads, and a more effective way to compute multi-head attention is needed to improve the efficiency of transformers.

1.3 Contribution

Leveraging the idea of the finite admixture model (FAM) [49, 8], we propose a new class of efficient transformer architectures, namely the Transformer with a Finite Admixture of Shared Heads (FiSHformer). At the core of FiSHformer is to sample local attention matrices from an admixture of a small number of global attention matrices. This sharing mechanism between heads helps reduce the computational complexity and the model size compared to the MHA softmax transformer. Our contribution is three-fold:

1. We construct an admixture model for shared attention matrices between heads and propose FiSHformer, a novel class of transformers that take advantage of this admixture model to efficiently compute multi-head attention.
2. We introduce a nonlinearity mapping from global heads to local heads into FiSH and propose the Generalized FiSHformer (GFiSHformer). We then explore different possibilities to design FiSHformer and GFiSHformer.
3. We empirically verify that FiSHformer and GFiSHformer achieve similar or even better accuracy but with much less computational cost in terms of FLOPs and smaller model complexity measured by the number of parameters. The advantages of our methods grow with the model/feature dimension D and the input sequence length N.

We also show that FiSHformer-based models help reduce head redundancy in our experiments.

Organization: We structure this paper as follows: In Section 2, we develop a finite admixture model of shared heads and then present our FiSHformer and its extensions. In Section 2.5, we analyze the reduction in model complexity and computational cost from FiSH. In Section 3 and 4, we validate and empirically analyze the efficiency and accuracy of FiSHformer, as well as conducting ablation studies on the model. We discuss related works in Section 5. The paper ends up with concluding remarks. More results and details are provided in the Appendix.

2 Transformer with a Finite Admixture of Shared Heads

We first review the finite admixture model (FAM) and derive a FAM of shared heads for the multi-head self-attention. We then define Transformer with a Finite Admixture of Shared Heads (FiSHformer).

2.1 A Probabilistic Viewpoint of Attention Matrices

Let $\mathbf{A}_j$ denote the attention matrix at the jth head, $j = 1, 2, \ldots, H$. From a probabilistic viewpoint, to have diversity among $\mathbf{A}_1, \ldots, \mathbf{A}_H$, we can assume that $\mathbf{A}_j$ comes from a distribution $\mathbb{P}_j$ for all j. Since the distributions $\mathbb{P}_1, \ldots, \mathbb{P}_H$ can have complex forms and be difficult to compute, our approach is to consider approximated distributions of $\mathbb{P}_j$ and these approximated distributions have simple forms. A natural choice for each of these approximated distributions is via a finite mixture of Gaussian distributions, which can be summarized in the following lemma below.

Lemma 1. *Assume that $P \in \mathbb{R}^{D'}$ is a probability distribution supported on some compact set and admits differentiable and bounded density estimation p. Then, for any scale parameter $\sigma > 0$ and for any $\epsilon > 0$, there exists universal constant C and $M \leq (C \log(1/\epsilon))^{D'}$ such that we can find a mixture of M components $\sum_{i=1}^{M} p_i \mathcal{N}(\theta_i, \sigma^2 \mathbf{I}_{D'})$ where $p_1, \ldots, p_K$ are weight parameters and $\theta_1, \ldots, \theta_K$ are location parameters that satisfy the following inequality*

$$\sup_{x \in \mathbb{R}^d} |p(x) - \sum_{i=1}^{M} p_i \phi(x|\theta_i, \sigma^2 \mathbf{I}_{D'})| \leq \epsilon,$$

where $\phi(.|\theta, \sigma^2 \mathbf{I})$ is Gaussian density function with location parameter θ and covariance matrix $\sigma^2 \mathbf{I}_{D'}$.

The proof of Lemma 1 is in Appendix E. In light of Lemma 1, for each scale parameter $\sigma > 0$ and for each distribution P_j, we can find the corresponding number of components M_j,

weight parameters $p_{1j}, \ldots, p_{M_j j}$, and location parameters $\theta_{1j}, \ldots, \theta_{M_j j}$ such that the mixtures $\mathbb{P}'_j = \sum_{i=1}^{M_j} p_{ij} \mathcal{N}(\theta_{ij}, \sigma^2 \mathbf{I}_{D'})$ can approximate the distribution P_j up to a given accuracy ϵ. However, these approximations still involve $\prod_{j=1}^{H} M_j$ number of location parameters, which can be computationally expensive. To overcome this issue, we assume that $M_1 = M_2 = \ldots = M_H$ and the location parameters $(\theta_{1j}, \ldots, \theta_{M_j j}) = (\theta_1, \ldots, \theta_M)$ for all j, i.e., these approximated mixtures share a similar set of location parameters. This sharing information of location parameters has a deep connection to finite admixture models, which we are going to elaborate in the next sections.

2.1.1 Background

Finite admixture models (FAM) are extensions of finite mixture models (FMMs), which served as a workhorse in stochastic modeling. A finite mixture distribution of M components for a random array $\mathbf{X} \in \mathbb{R}^{N \times J}$ is given by

$$\boldsymbol{x}_j \sim \sum_{k=1}^{M} p_k f(\boldsymbol{x}; \theta_k), \quad \sum_{k=1}^{M} p_k = 1, \quad p_k \geq 0, \tag{4}$$

where $\boldsymbol{x}_j \in \mathbb{R}^N$ is the j-th row of $\mathbf{X}$ randomly sampled from the mixture distribution, f is a chosen probability measure, such as a Gaussian distribution, $p = \{p_1, p_2, \ldots, p_M\}$ are mixture weights, and θ_k denotes the parameter values for the k-th component.

A FAM is a generalization of a FMM where rows $\boldsymbol{x}_j$, $j = 1, \ldots, H$, are drawn from different mixture distributions that share the components $f(\boldsymbol{x}; \theta_k)$, $k = 1, \ldots, M$ but with different mixture weights

$$\boldsymbol{x}_j \sim \sum_{k=1}^{M} p_{kj} f(\boldsymbol{x}; \theta_k), \quad \sum_{k=1}^{M} p_{kj} = 1, \quad p_{kj} \geq 0. \tag{5}$$

2.2 Multi-head as a Finite Admixture Model of Shared Heads (FiSH)

As demonstrated in Section 2.1, we propose a Finite Admixture Model of Shared Heads (FiSH), in which $\mathbf{A}_j$ follows finite admixture distribution of M components given by

$$\mathbf{A}_j \sim \sum_{k=1}^{M} p_{kj} f(\mathbf{A}; \theta_k), \quad \sum_{k=1}^{M} p_{kj} = 1, \quad p_{kj} \geq 0. \tag{6}$$

Here $M < H$ and $f(\mathbf{A}; \theta_k)$ are chosen probability measures. In particular, we choose $f(\mathbf{A}; \theta_k)$ to be Gaussian distributions $\mathcal{N}(\mathbf{A}; \mathbf{G}_k, \boldsymbol{\Sigma}_k)$, where $\mathbf{G}_k = \mathbf{Q}_k {\mathbf{K}_k}^\top$ and $\boldsymbol{\Sigma}_k = \sigma_k^2 \mathbf{I}$ are the cluster means and coveriances, respectively. FiSH is then defined as follows:

Definition 1 (Finite Admixture Model of Shared Heads)**.** *The multi-head attention admits a finite admixture model of shared heads if the attention matrices $\mathbf{A}_j$ at the jth head are sampled from the following finite admixture model:*

$$\mathbf{A}_j \sim \sum_{k=1}^{M} p_{kj} \mathcal{N}(\mathbf{A};\, \mathbf{Q}_k {\mathbf{K}_k}^\top, \sigma_k^2 \mathbf{I}), \quad \sum_{k=1}^{M} p_{kj} = 1, \; p_{kj} \geq 0. \tag{7}$$

In FiSH defined in Def. 1, we call $\{\mathbf{G}_k = \mathbf{Q}_k {\mathbf{K}_k}^\top\}_{k=1,\ldots,M}$ global attention matrices and $\{\mathbf{A}_j\}_{j=1,\ldots,H}$ local attention matrices. FiSH computes M global attention matrices $\mathbf{G}_k$, and H local attention matrices $\mathbf{A}_j$ are sampled from FiSH as in Eqn. 7 with $M < H$.

Remark 1 (FiSH vs. Baseline MHA)**.** *The baseline MHA with H heads need to compute H, e.g. $H = 8$, attention matrices, each of which requires $\mathcal{O}(N^2)$ computational costs where N is the length of the input sequence. In contrast, FiSH only need to compute $M < H$, e.g. $M = 2$, global attention matrices, each of which also requires $\mathcal{O}(N^2)$ computational and memory costs. Then FiSH combines those global attention matrices to form a FAM from which H, e.g. $H = 8$, local attention matrices are sampled as in Eqn. 7. This second step of sampling local attention matrices from a set of global attention matrices in FiSH requires very few computations. Thus, FiSH is more efficient than MHA.*

Remark 2 (Connection to Topic Models)**.** *FiSH can be interpreted as a Probabilistic Latent Semantic Analysis (pLSA) model for topic modeling. Considering the document d that contains the word w whose topic is c, pLSA models the occurrence of the word w in the document d as a mixture of conditionally independent Multinomial distributions $p(w|d) = \sum_c p(c|d)p(w|c)$. Comparing this pLSA with Eqn. 7 of FiSH, we can associate the mixture weights p_{kj} and the distribution*

$\mathcal{N}(\mathbf{A};\, \mathbf{Q}_k{\mathbf{K}_k}^\top, \sigma_k^2\mathbf{I})$ *in FiSH with the distributions* $p(c|d)$ *and* $p(w|c)$ *in pLSA, respectively. Therefore, it can be interpreted that the global attention matrices in FiSH play the role of topics, and the local attention matrices in FiSH are words sampled from those topics. It is interesting to note that pLSA is equivalent to the famous Latent Dirichlet Allocation model under a uniform Dirichlet prior on the per-document topic distribution* $p(c|d)$.

2.3 Transformer with a Finite Admixture of Shared Heads

FiSHformers are transformers that use FiSH instead of MHA. FiSH, as defined in Def. 1, is not differentiable, which poses a difficulty in training FiSHformers. Applying the reparameterization trick [34], the attention matrices $\mathbf{A}_j$ can be written in a differentiable form as follows:

$$\mathbf{A}_j = \sum_{k=1}^{M} p_{kj}(\mathbf{Q}_k{\mathbf{K}_k}^\top + \sigma_k \odot \epsilon_j),\;\; \epsilon_j \sim \mathcal{N}(0, \mathbf{I}),\;\; \sum_{k=1}^{M} p_{kj} = 1,\;\; p_{kj} \geq 0. \tag{8}$$

FiSHformers use the formulation of local attention heads in Eqn. 8 to implement FiSH.

Transformer with a Hard Finite Admixture of Shared Heads (Hard FiSHformer) Hard FiSHformer takes the zero-noise limit of Eqn. 8 to reduce the computational cost. The attention matrices $\mathbf{A}_j$ in Hard FiSHformer are then calculated as

$$\mathbf{A}_j = \sum_{k=1}^{M} p_{kj}\mathbf{Q}_k{\mathbf{K}_k}^\top,\;\; \sum_{k=1}^{M} p_{kj} = 1,\;\; p_{kj} \geq 0. \tag{9}$$

Remark 3 (Discriminative Relaxation)**.** *To take the advantage of learning from data, the convex combination condition of* p_{kj}*, i.e.* $\sum_{k=1}^{M} p_{kj} = 1,\;\; p_{kj} \geq 0$*, can be relaxed, and those mixing coefficients are made learnable parameters that are learned from data during training*

Remark 4 (Transformers with a Mixture of Shared Heads)**.** *A transformer with a mixture of shared heads (MiSHformer) can be used to reduce the amount of computation with the cost of accuracy reduction. MiSH is a special case of FiSH when the mixture weights* p_{kj} *are the same for all* j*. The local attention matrices* $\mathbf{A}_j$ *in MiSHformer are given by*

$$\mathbf{A}_j = \sum_{k=1}^{M} p_{k}(\mathbf{Q}_k{\mathbf{K}_k}^\top + \sigma_k \odot \epsilon_j),\;\; \epsilon_j \sim \mathcal{N}(0, \mathbf{I}),\;\; \sum_{k=1}^{M} p_{k} = 1,\;\; p_{k} \geq 0. \tag{10}$$

An empirical comparison between FiSHformer and MiSHformer is provided in Section 4.

2.4 Transformer with a Generalized Finite Admixture of Shared Heads

In order to increase the representation capacity of attention heads, we follow a common approach in learning representation by replacing the linear mapping in Eqn. 8 and 9 by a nonlinear mapping such as a neural network with the rectified linear units (ReLU). The Transformer with a Generalized Finite Admixture of Shared Heads (GFiSHformer) is then formulated as

$$\mathbf{A}_j = \sum_{k=1}^{M} \phi(p_{kj}(\mathbf{Q}_k{\mathbf{K}_k}^\top + \sigma_k \odot \epsilon_j)),\;\; \epsilon \sim \mathcal{N}(0, \mathbf{I}),$$

where ϕ is a nonlinear mapping and p_{kj} are relaxed to be learnable parameters. Similarly, we formulate local attention matrices $\mathbf{A}_j$ in the Transformer with a Generalized Hard Finite Admixture of Shared Heads (Hard GFiSHformer) as $\mathbf{A}_j = \sum_{k=1}^{M} \phi(p_{kj}\mathbf{Q}_k{\mathbf{K}_k}^\top)$.

2.5 Reduction in Model Complexity and Computational Cost from FiSH

Compared to its H-head MHA counterpart, a FiSH attention of M global heads and H local heads saves $[2(H-M)D - 2MH)]N^2 + 2(H-M)D(2D_x - 1)N$ FLOPs in a forward pass and $2(H-M)DD_x - HM - M$ parameters. Detailed derivations are provided in Appendix D.

3 Experimental Results

In this section, we empirically study the advantages of FiSHformer on various tasks and benchmarks, including language modeling on WikiText-103 (Section 3.1), machine translation on IWSLT' 14 De-En and WMT'14 (Section 3.2), and image classification on ImageNet (Section 3.3). We aim to show that: (i) FiSHformers improve the efficiency and accuracy upon the MHA baseline; (ii) FiSH is a universal method that can be applied on state-of-the-art transformer models to improve their

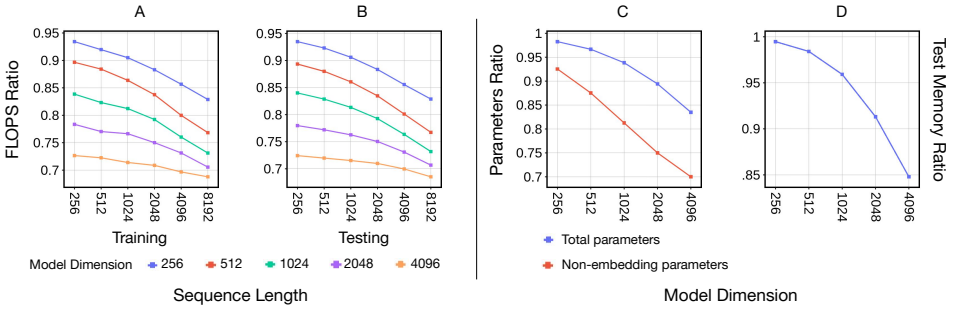

Figure 3: (Left) Training (A) and Inference (B) FLOP ratios between a 2-global-head GFiSHformers with 8-head MHA baselines across different model dimensions D and sequence lengths N trained on the WikiText-103 language modeling task. (Right) Number of parameters (C) and GPU memory usage at test time (D) ratios between 2-global-head GFiSHformers with 8-head MHA baselines across different model dimensions D. 2-global-head GFiSHformers are significantly more efficient than the baseline as D and N increase, indicating the benefits of our method for long-range and large-scale tasks.

Table 1: Perplexity (PPL) on WikiText-103 compared to the baselines.

Method	Valid PPL	Test PPL
Softmax 8 heads	33.15	34.29
Softmax 4 heads	34.80	35.85
Hard FiSHformer 4 global heads	33.10	34.11
Hard FiSHformer 2 global heads	34.14	35.24
FiSHformer 4 global heads	33.15	34.16
FiSHformer 2 global heads	34.01	34.96
Hard GFiSHformer 4 global heads	32.70	33.75
Hard GFiSHformer 2 global heads	33.31	34.63
GFiSHformer 4 global heads	**32.68**	**33.71**
GFiSHformer 2 global heads	33.21	34.48

performance on large-scale applications. In Section 4, we also show that FiSH helps reduce the redundancy between attention heads.

We compare FiSHformers, Hard FiSHformers, GFiSHformers, and Hard GFiSHformers with the baseline MHA softmax transformers. In our experiments, we apply the discriminative relaxation explained in Remark 3 on our FiSHformers to make the mixture weights p_{kj} learnable parameters. For GFiSHformers/Hard GFiSHformers, we choose the nonlinear mapping ϕ to be a ReLU followed by a linear neural network. All of our results are averaged over 5 runs with different seeds. More details on datasets, models, and training are provided in Appendix A.

3.1 WikiText-103 Language Modeling

Models and baselines We compare the 2 and 4-global-head FiSHformers with the 8-head softmax transformers [67]. Each model has 16 layers, and our training follows the setting from [58].

Results Perplexity: Table 3.1 demonstrates that our 2/4-global-head (G)FiSHformers and their hard versions obtain comparable or better PPLs than the corresponding 8 head MHA baseline on WikiText-103. Interestingly, the 2-global-head (G)FiSHformers perform on par with the 8-head baseline even though only 2 global attention matrices are used to span all local attention matrices, indicating that the attention matrices in MHA are indeed redundant and the representation capacity of local attention matrices in (G)FiSH, though being generated from only 2 global bases, is comparable to those in the 8-head MHA.

Efficiency: In Fig. 3A and 3B, we presents the reduction ratio of train and test FLOPS, respectively, of our 2-global-head GFiSHformer vs. the baseline 8-head MHA transformer as functions of model dimension D and sequence length N. In Fig. 3C and 3D, we show the reduction ratio of model size and GPU memory usage at test time, respectively, of our 2-global-head GFiSHformer vs. the same baseline. We observe that the efficiency advantage of GFiSHformer over the baseline grows with D and N, making it more suitable and superior for large-scale applications. Note that the model size in terms of the number of parameters does not depend on the sequence length N, and from our experiments, we observe that the GPU memory usage reduction ratio is almost the same for different sequence lengths. More efficiency analysis results on this language modeling task are

Table 2: Machine translation BLEU scores of 2-global-head (G)FiSHformers compared to the 4-head baseline on the IWSLT14 De-En dataset. Our methods perform on par or better than the baseline while being more efficient.

Method	BLEU score
Softmax 4 heads	34.42
Hard Fishformer 2 global heads	34.31
FiSHformer 2 global heads	34.38
Hard GFiSHformer 2 global heads	34.54
GFiSHformer 2 global heads	**34.71**

Table 3: Machine translation BLEU scores of 2-global-head (G)FiSH + linear transformers compared to the 4-head baseline linear on the IWSLT14 De-En benchmark. Our methods significantly outperform the linear baseline.

Method	BLEU score
Linear 4 heads	28.22
Hard LFiSHformer 2 global heads	**30.24**
LFiSHformer 2 global heads	29.63
Hard GLFiSHformer 2 global heads	29.20
GLFiSHformer 2 global heads	29.46

provided in Section 4 and Appendix C. Also, Figure 5 in Appendix B.2 shows the train and test PPL of (G)FiSHformers and the MHA softmax transformers. In our experiment, 4-global-head GFiSHformers obtain the best validation PPL.

3.2 Machine Translation

In this section, we examine the performance of (G)FiSHformer on the neural machine translation task, an important task in natural language processing in which the sequence lengths of the input are not the same. We first compare (G)FiSHformers with the baselines MHA softmax transformers on the IWSLT' 14 De-En [10] and then scale up our experiments to the WMT'14 En-De [40]. On these tasks, we calculate the BLEU scores for evaluation.

Models and baselines For the IWSLT' 14 De-En task, we compare 2-global-heads (G)FiSHformers with the baseline 4-head softmax transformer. Each model consists of 12 layers, 6 layers for an encoder and the other 6 layers for a decoder. Our experiments follow the setting on fairseq. For the WMT'14 En-De task, we use similar models as in the IWSLT' 14 De-En task. However, we compare (G)FiSHformers of 8 and 4 global heads with the 16-head MHA softmax baseline. Our training and model setting are the same as those in [45].

Results As shown in Table 2 and 4, (G)FiSHformers outperform or at least are on par with the baseline MHA softmax transformers. Again, these results indicate rich representations of the local attention matrices generated by (G)FiSH. Furthermore, (G)FiSHformer outperforming Hard (G)FiSHformer in all settings suggests the positive value of adding noise into the models to turn them into a proper probabilistic model. Nevertheless, it is worth noticing that Hard (G)FiSHformer is more efficient than (G)FiSHformer. Fig. 4 in Appendix B.1 summarizes the advantage in efficiency of 2-global-head GFiSHformer over the 4-head baseline on the IWSLT' 14 De-En task. These advantages of GFiSHformer grow with the model dimension D.

3.3 Image Classification on ImageNet

The advantages of (G)FiSHformers also hold across different data modalities. To illustrate this point, in this section, we apply (G)FiSH to Swin transformer [39], a state-of-the-art vision transformer architecture, for the image classification task on the ImageNet dataset [19]. The baseline Swin-T we use has a total of 12 layers, across 4 stages of transformer blocks with 3, 6, 12, and 24 heads each. Our model and training follow the settings in [39]. We summarize our results in Table 5. Our GFiSHformer is only slightly more efficient than the baseline in this case because the sequence length N per window for this task is small, i.e. $N = 49$. However, like with the previous language tasks and as pointed by formula of computational cost and model complexity reduction in Section 2.5, these advantages grow with larger D and N.

3.4 FiSHformer is more effective than other methods for head-redundancy reduction

To futher demonstrate the effectiveness of our method, we compare FiSHformers against the head-redundancy reduction method in [15] on the WMT'14 machine translation task. [15] proposes the adaptively sparse transformer (AST), reducing redundancy within each head by zeroing out low-attention scores. In comparison, the results further confirm the effectiveness of our method since the BLEU score of FiSHformer and GFiSHformer are 27.26 and 27.67, respectively, better than that of AST, which is 26.93. All models share the same architecture, with 12 transformer layers, 6 encoder, and 6 decoder layers. Our FiSHformers have 4 global heads and 8 local heads per layer while 8 attention heads are used in each AST layer.

Table 4: BLEU scores of (G)FiSHformers, with various numbers of shared heads compared to the 16-head baseline on the WMT'14 En-De machine translation. Our methods obtain comparable/better results than the baseline while being more efficient.

Method	BLEU score
Softmax 16 heads	29.38
Hard FiSHformer 8 global heads	29.32
FiSHformer 8 global heads	**29.57**
Hard GFiSHformer 8 global heads	29.27
GFiSHformer 8 global heads	29.42
Hard GFiSHformer 4 global heads	28.97
GFiSHformer 4 global heads	29.34

Table 5: ImageNet Image Classification accuracy scores, FLOPs, and number of parameters on Swin Transformer, comparing between baseline Swin-T and our GFiSH Swin-T. Baseline results from [39] are provided in parentheses. Note that the FLOPs is calculated per iteration, with a batch size of 1024.

Method	Acc top-1	Acc top-5	FLOPs (10^9)	Params
Softmax-12/24	**81.20**	**95.50**	4618.24	28.3M
Hard-GFiSH-6/12	81.11	95.44	4372.48	26.2M

3.5 Beyond Multi-Head Softmax Transformers

We show that (G)FiSH can be applied on top of many transformer architectures to improve their performance including the linear transformers [32] and the SoTA transformer with noisy back-translation [23]. More results of combining (G)FiSH with efficient transformers are in the Appendix.

Applying (G)FiSH on Linear Transformers Linear transformers [32] is a class of efficient transformers that linearize the softmax kernel in Eqns. 1 and 2 when computing attention matrices. We apply (G)FiSH on linear transformers trained for the IWSLT14 De-En machine translation task and summarize the results in Table 3. The empirical results verify that applying (G)FiSH using only 2-global heads on a 4-head linear transformer improves the accuracy of the baseline model.

(G)FiSH Improves the State-of-the-Art Noisy Back-Translation We apply an 8-global-head Hard GFiSH on the transformers trained with noisy back-translation for the WMT'14 En-De translation task and obtain the BLEU score of 33.45. This result is comparable to the SoTA result of 33.52 from the transformers trained with noisy back-translation but our model is more efficient.

4 Empirical Analysis

We study models trained for the WikiText-103 language modeling task in this section.

Efficiency Analysis In this section, we further investigate the efficiency reduction of 2-global-head GFiSHformers over the 8-head baseline as a function of the number of heads in Fig. 7 and 8 and compare the efficiency of our FiSH-based models in Fig. 6. Fig. 6, 7, 8 and details on our setting are provided in Appendix C. From Fig. 7 and 8, we observe that when using fewer number of global heads, GFiSHformers achieve significantly more computation reduction (in both training and inference). Furthermore, Fig. 6 shows that the efficiency measures, i.e. FLOPs, model size, and GPU memory usage, of all FiSH-based models we study in this paper follow similar patterns.

FiSHformer Helps Reducing Head Redundancy We show that (G)FiSHformers attain more significant distances between heads than the baseline. Thus, our models capture more diverse patterns across heads than the baseline. For a given pre-trained model, we compute the pair-wise $\mathcal{L}_2$ distances between heads in the same layer. We show the layer-average mean and variance of distances between heads in GFiSHformers compared with those in the MHA softmax baselines in Table 6. We provide additional results for Hard GFiSHformers and Hard FiSHformer in Table 10 in the Appendix.

Eigen Analysis We show that heads in GFiSHformer lie on a higher-dimensional subspace compared to those in FiSHformer. This justifies our use of nonlinearity mapping to generate local heads from global heads. Using a pre-trained model, we first compute the covariance matrix of the vectorized attention scores of the l-th layer: $\mathcal{C}^l = \frac{1}{M \cdot H} \sum_{m=1}^{M} \sum_{h=1}^{H} (A_m^{l,h})(A_m^{l,h})^\top$. We use spectral decomposition to represent $\mathcal{C}^l$ in terms of eigenvalues and eigenvectors, namely, $\mathcal{C}^l = \sum_{i=1}^{n^2} \lambda_i v_i v_i^\top$. Without losing generality, we assume that eigenvalues are sorted in descending order. We illustrate the layer-average number of principle components that are needed to explain 95% variance in Table 7. Interestingly, Table 7 shows that attention matrices in all of our proposed FiSH-based models lie on higher-dimensional subspace than those in the baseline MHA softmax transformers, which indicates that our models achieve better representational capacity than the baseline, confirming the advantage of (G)FiSH over MHA. Table 11 in the Appendix provide additional results for Hard (G)FiSHformer.

Table 6: Laver-Average mean and variance of $\mathcal{L}_2$ distances between heads of models trained for the WikiText-103 language modeling task.

Method	Mean	Variance
Softmax 8 heads	1.62	0.66
GFiSHformer 2 global heads	2.93	2.62
GFiSHformer 4 global heads	**3.59**	**3.95**
GFiSHformer 6 global heads	3.37	2.78

Table 7: Layer-average number of principle components for 95% variance explained of the covariance of attention matrices (WikiText-103 models).

Method	Mean
Softmax 8 heads	296
GFiSHformer 2 global heads	895
GFiSHformer 4 global heads	**1408**
GFiSHformer 6 global heads	1228

Table 8: Perplexity (PPL) on WikiText-103 of 2-global-head FiSHformer vs. 2-global-head MiSHformer compared to the 8-head baseline. MiSHformer attains worse PPL than FiSHformer.

Method	Valid PPL	Test PPL
Softmax 8 heads	**33.15**	**34.29**
FiSHformer 2 global heads	34.01	34.96
MiSHformer 2 global heads	35.11	36.28

Table 9: Perplexity on WikiText-103 of GFiSHformer with various number of global heads compared with the 8-head baseline.

Method	Valid PPL	Test PPL
Softmax 8 heads	33.15	34.29
GFiSHformer 6 global heads	32.80	33.80
GFiSHformer 4 global heads	**32.68**	**33.71**
GFiSHformer 2 global heads	33.21	34.48

Admixture vs. Mixture of Heads We compare the transformer with a mixture of heads and the transformer with an admixture of heads. We show that the transformer with a mixture of heads yields worse accuracy. We summarize our results on the WikiText-103 language modeling task in Table 8.

Ablation Study on the Effect of the Number of Global Heads on FiSH-based Models We investigate the accuracy, efficiency, and representation capacity of FiSH-based models under different numbers of global heads on the WikiText-103 language modeling task. Since GFiSH obtains the best accuracy on this task, we use GFiSH as a study case in this section and report our results on accuracy in Table 9. Ablation results on efficiency are summarized in Fig. 7, 8 in Appendix C, and ablation results on representation capacity are provided in Table 6, 7.

5 Related Work

Efficient Transformers Efficient transformers have been developed to reduce the quadratic computational and memory cost of transformers [54]. A class of efficient transformers are sparse transformers which design the attention matrix to have sparse structure [47, 37, 50, 11, 6]. Another class of efficient transformers are models that integrate different access patterns for better coverage [11, 29]. These access patterns can also be learned from the data [35, 54, 63]. In other works, a side memory module is used to access multiple tokens simultaneously [36, 61, 3, 6]. Recently, low-rank and kernelization methods have been proposed to improve the computational and memory efficiency of computing self-attention [66, 71, 32, 13, 59, 44, 48]. Our (G)FiSHformers are complementary to these methods.

Redundancy in Transformers It has been shown that most of the neurons and heads in the pre-trained transformer are redundant and can be pruned when applied on a downstream task [17, 42, 22]. The contextualized embeddings in pre-trained networks under this redundancy due to overparameterization have also been studied to demonstrate that the representations learned within these models are highly anisotropic [43, 24]. Knowledge distillation and sparse approximation have also been used to enhance the efficiency of transformers, including [57, 62, 70, 56]. Our FiSH-based approach can be used along these methods to improve their performance.

Mixture Models for Transformers Recently, mixture models have been employed to study and enhance transformers. Among these works is switch transformers [25] that uses the routing algorithm in Mixture of Experts (MoE) to decrease the communication and computational costs in transformers. Other works that combine mixture models with transformers include [12, 27, 31].

6 Concluding Remarks

In this paper, we proposed the FiSHformer, a class of transformers that samples local attention matrices from a finite admixture of global attention matrices. FiSHformers and their generalized version GFiSHformers attain better computational cost and model complexity than their baseline MHA softmax transformer counterparts. Furthermore, (G)FiSHformers help increase the diversity between attention heads. It is worth noting that there is no potential negative societal impacts of FiSHformers. Also, global attention matrices in FiSHformers currenly do not have any structure, and this is a limitation of our model. It is interesting to impose additional structures such as low-rank and sparsity into the global attention matrices.

References

[1] Rami Al-Rfou, DK Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In *Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

[2] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3159–3166, 2019.

[3] Akari Asai and Eunsol Choi. Challenges in information seeking qa: Unanswerable questions and paragraph retrieval. *arXiv preprint arXiv:2010.11915*, 2020.

[4] A. G. Bacharoglou. Approximation of probability distributions by convex mixtures of Gaussian measures. *Proceedings of the American Mathematical Society*, 138:2619–2628, 2010.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

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

[7] Srinadh Bhojanapalli, Ayan Chakrabarti, Himanshu Jain, Sanjiv Kumar, Michal Lukasik, and Andreas Veit. Eigen analysis of self-attention and its reconstruction from partial computation. *arXiv preprint arXiv:2106.08823*, 2021.

[8] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[9] Tom Brown and et al. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[10] Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, volume 57, 2014.

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

[12] Sung Min Cho, Eunhyeok Park, and Sungjoo Yoo. Meantime: Mixture of attention mechanisms with multi-temporal embeddings for sequential recommendation. In *Fourteenth ACM Conference on Recommender Systems*, pages 515–520, 2020.

[13] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021.

[14] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy, August 2019. Association for Computational Linguistics.

[15] Gonçalo M Correia, Vlad Niculae, and André FT Martins. Adaptively sparse transformers. *arXiv preprint arXiv:1909.00015*, 2019.

[16] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

[17] Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov. Analyzing redundancy in pretrained transformer models. *arXiv preprint arXiv:2004.04010*, 2020.

[18] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

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

[21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[22] Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. Analyzing individual neurons in pre-trained language models. *arXiv preprint arXiv:2010.02695*, 2020.

[23] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[24] Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *arXiv preprint arXiv:1909.00512*, 2019.

[25] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

[26] S. Ghosal and A. van der Vaart. Entropies and rates of convergence for maximum likelihood and bayes estimation for mixtures of normal densities. *Ann. Statist.*, 29:1233–1263, 2001.

[27] Maosheng Guo, Yu Zhang, and Ting Liu. Gaussian transformer: a lightweight approach for natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6489–6496, 2019.

[28] John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China, November 2019. Association for Computational Linguistics.

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

[30] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[31] Junyan Jiang, Gus G Xia, Dave B Carlton, Chris N Anderson, and Ryan H Miyakawa. Transformer vae: A hierarchical model for structure-aware and interpretable music representation learning. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 516–520. IEEE, 2020.

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

[33] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. Structured attention networks. *arXiv preprint arXiv:1702.00887*, 2017.

[34] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[35] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

[36] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pages 3744–3753. PMLR, 2019.

[37] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*, 2018.

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

[39] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, October 2021.

[40] Matouš Macháček and Ondřej Bojar. Results of the wmt14 metrics shared task. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 293–301, 2014.

[41] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[42] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[43] Jiaqi Mu and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*, 2018.

[44] Tan M. Nguyen, Vai Suliafu, Stanley J. Osher, Long Chen, and Bao Wang. Fmmformer: Efficient and flexible transformer via decomposed near-field and far-field attention. *arXiv preprint arXiv:2108.02347*, 2021.

[45] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[46] Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. Bleu: a method for automatic evaluation of machine translation. pages 311–318, 2002.

[47] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4055–4064. PMLR, 10–15 Jul 2018.

[48] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2021.

[49] Jonathan K Pritchard, Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000.

[50] Jiezhong Qiu, Hao Ma, Omer Levy, Scott Wen-tau Yih, Sinong Wang, and Jie Tang. Blockwise self-attention for long document understanding. *arXiv preprint arXiv:1911.02972*, 2019.

[51] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI report*, 2018.

[52] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[53] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

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

[55] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[56] Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. Poor man's bert: Smaller and faster transformer models. *arXiv e-prints*, pages arXiv–2004, 2020.

[57] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[58] Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning*, pages 9355–9366. PMLR, 2021.

[59] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3531–3539, 2021.

[60] David R So, Chen Liang, and Quoc V Le. The evolved transformer. *arXiv preprint arXiv:1901.11117*, 2019.

[61] Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, and Armand Joulin. Augmenting self-attention with persistent memory. *arXiv preprint arXiv:1907.01470*, 2019.

[62] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*, 2019.

[63] Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse Sinkhorn attention. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9438–9447. PMLR, 13–18 Jul 2020.

[64] Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy, July 2019. Association for Computational Linguistics.

[65] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[66] Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[68] Jesse Vig and Yonatan Belinkov. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, Florence, Italy, August 2019. Association for Computational Linguistics.

[69] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy, July 2019. Association for Computational Linguistics.

[70] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, 2019.

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

[72] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, June 2018.

[73] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]