Learning Adaptive Axis Attentions in Fine-tuning: Beyond Fixed Sparse Attention Patterns

Zihan Wang[1*] **Jiuxiang Gu**[2] **Jason Kuen**[2] **Handong Zhao**[2] **Vlad I. Morariu**[2]
Ruiyi Zhang[2] **Ani Nenkova**[2] **Tong Sun**[2] **Jingbo Shang**[1]
[1]University of California, San Diego [2]Adobe Research
[1]{ziw224, jshang}@ucsd.edu
[2]{jigu, kuen, hazhao, morariu, ruizhang, nenkova, tsun}@adobe.com

Abstract

We present a comprehensive study of sparse attention patterns in Transformer models. We first question the need for pre-training with sparse attention and present experiments showing that an efficient fine-tuning only approach yields a slightly worse but still competitive model. Then we compare the widely used local attention pattern and the less-well-studied global attention pattern, demonstrating that global patterns have several unique advantages. We also demonstrate that a flexible approach to attention, with different patterns across different layers of the model, is beneficial for some tasks. Drawing on this insight, we propose a novel Adaptive Axis Attention method, which learns—during fine-tuning—different attention patterns for each Transformer layer depending on the downstream task. Rather than choosing a fixed attention pattern, the adaptive axis attention method identifies important tokens—for each task and model layer—and focuses attention on those. It does not require pre-training to accommodate the sparse patterns and demonstrates competitive and sometimes better performance against fixed sparse attention patterns that require resource-intensive pre-training.

1 Introduction

The wide adoption of the Transformer architecture (Vaswani et al., 2017) in contextual language representations such as BERT (Devlin et al., 2019) has spurred interest in making transformers more efficient via sparse attention patterns (Li et al., 2019; Guo et al., 2019; Gong et al., 2019; Zaheer et al., 2020; Child et al., 2019).

The typical process for learning a transformer model (*e.g.*, BERT) with a sparse attention pattern is to replace the full attention calculation with that pattern, then pre-train the model with the usual pre-training task and fine-tune the model to downstream tasks. The use of sparse attention pattern does not necessarily significantly improve the run time of the models[1] but it does reduce the model memory requirement during inference time. This reduction is helpful when deploying models on mobile devices or other memory-limited devices.

In this paper we offer an extensive analysis of attention patterns, organized around the following questions: (1) is *pre-training* essential or is it possible to employ sparse patterns during *fine-tuning* only? (2) which *types* of *attention patterns* are important? (3) should *the same* attention pattern be applied to different downstream tasks and to all layers of the model?

The answer to the first question carries critical implications for the practical adoption of sparse attention approaches. Most current transformer-based approaches learn fixed patterns during pre-training and then apply these to fine-tuning as well. However, it is costly and impractical to pre-train a new model from scratch when a different attention pattern is expected to be more appropriate for a task. Learning the sparse attention pattern model during fine-tuning is more reasonable.

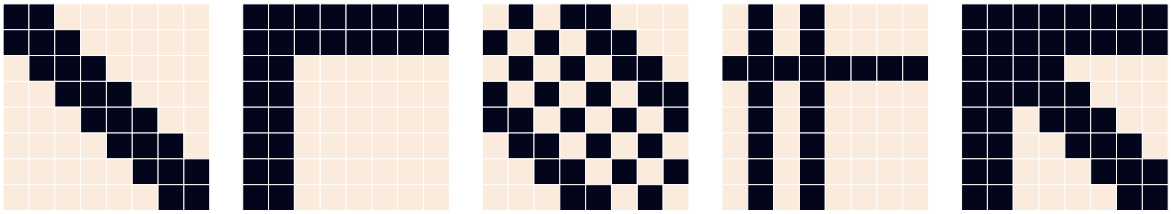
With this motivation in mind, we perform a controlled experiment on the eight tasks in the GLUE (Wang et al., 2019a) benchmark. We find that pre-training with sparse patterns is not a crucial ingredient for good performance—learning the model solely during fine-tuning sacrifices only one or two performance points on most tasks. Grounded in this finding, we perform all other experiments efficiently, starting with the same pre-trained model and varying sparse attention patterns during fine-tuning alone.

We start to answer the second question by analyzing the two most popular patterns: local and global (Tay et al., 2020). Local patterns allow each token to attend only to other tokens within a given window. Global patterns allow some specially des-

[*] This work was done during the author's internship at Adobe Research.

[1] Due to efficient vectorizations and cache locality of full attention calculations.

Findings of the Association for Computational Linguistics: ACL 2022, pages 916 - 925



(a) Local (b) Global (c) Diagonal (d) Axis (e) Local+Global

Figure 1: Five attention patterns (with $N = 8$): Local, Global, their generalized forms: Diagonal and Axis, and a combination of Local and Global attention: Local+Global.

ignated tokens to attend to all other tokens while the remaining tokens are allowed to attend only to the specially designated tokens. We show that global pattern exhibits unique and complementary strengths that local patterns cannot capture. This finding is aligned with the design choices for recent models that benefit from the combination of both patterns (Beltagy et al., 2020; Zaheer et al., 2020).

For the third question, we extend SparseBERT (Shi et al., 2021) to an adaptive diagonal attention model. With this model, we are able flexibly learn task-wise and/or layer-wise diagonal patterns. Adapting attention patterns to tasks and layers improves performance over fixed attention pattern baselines and yields equivalent memory gains/sparsity levels.

Motivated by these findings, we design an adaptive sparse pattern that is learned during fine-tuning and that adapts to the task, layer as well as to the input sample. Our pattern is an instance of axis patterns (Figure 1(d)), which are a more general form of global patterns; we name it Adaptive Axis Attention (AAA). AAA samples the *important* tokens by applying a fully connected layer that is followed by Gumbel Softmax (Jang et al., 2017) applied to the token representations on each Transformer layer. The tokens identified as important are then designated as the global tokens and are used to form an axis-aligned attention pattern.

Through extensive experiments we verify that learning such an adaptive axis attention can outperform the fixed patterns adopted in Longformer (Beltagy et al., 2020), BigBird (Zaheer et al., 2020) and SparseBERT (Shi et al., 2021). AAA rivals or outperforms the fixed patterns even when compared with their pre-trained variants, which require extensive time and resources for pre-training.

We also show that AAA can be integrated into lightweight models, *e.g.*, MobileBERT (Sun et al., 2020). The benefits for MobileBERT indicate that our work is complementary to other methods for reducing hidden dimensions or attention heads.

Our comprehensive study of different sparse attention patterns in Transformers advances the field with several key insights.

- We show that pre-training sparse attention pattern models does bring benefits but that a fine-tuned only approach maintains competitive performance while saving cost and time for pre-training.
- We present an in-depth comparison between the two most common patterns in sparse attention design and verify that they provide different complementary strengths.
- We demonstrate that adapting attention patterns to tasks and layers is an impactful aspect of sparse pattern designs. We propose a new attention pattern—Adaptive Axis Attention and demonstrate that AAA outperforms fixed attention patterns.

2 Background

Here we highlight some of the core definitions related to self-attention and describe prior work on sparse self-attention.

2.1 Revisiting Self-Attention

BERT (Devlin et al., 2019) uses Masked Language Modeling (MLM), a self-supervised pre-training objective that allows a transformer encoder to encode a sequence from both directions simultaneously. Specifically, for an input sequence of N tokens, let $\boldsymbol{X}^{\ell} \in \mathbb{R}^{N \times D}$ be the encoded features at the ℓ-th transformer layer, where D denotes the embedding dimension. The features at the $(\ell + 1)$-th

layer are obtained by applying a transformer block:

$$\boldsymbol{H}^{\ell+1} = \mathrm{LN}\left(\boldsymbol{X}^{\ell-1} + f^{\ell}_{\mathrm{MHA}}(\boldsymbol{X}^{l})\right) \quad (1)$$

$$\boldsymbol{X}^{\ell+1} = \mathrm{LN}\left(\boldsymbol{H}^{\ell+1} + f^{\ell}_{\mathrm{FF}}(\boldsymbol{H}^{\ell+1})\right) \quad (2)$$

where LN denotes the layer normalization, $f_{\mathrm{FF}}(\cdot)$ is composed of two fully-connected sub-layers, wrapped in residual connection.

The Multi-Head Self-Attention (MHA) operation $f^{\ell}_{\mathrm{MHA}}(\cdot)$ in Eq. 1 is calculated as:

$$f^{\ell}_{\mathrm{MHA}}(\boldsymbol{X}) = [f^{\ell,1}_{\mathrm{Head}}(\boldsymbol{X}); \ldots; f^{\ell,h}_{\mathrm{Head}}(\boldsymbol{X})]\boldsymbol{U} \quad (3)$$

$$f^{\ell,i}_{\mathrm{Head}}(\boldsymbol{X}) = \sigma\left(\boldsymbol{A}/\sqrt{D_h}\right)\boldsymbol{V} \quad (4)$$

where $\sigma(\cdot)$ is a softmax function, $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{K}^T$ is the self-attention matrix, d is the model dimension, h is the number of heads, $\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_q, \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_k, \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_v \in \mathbb{R}^{N\times D_h}$. $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{D\times D_h}$ are the head-specific weights for query, key, and value vectors respectively, $D_h = D/h$ is the head dimension size, and $\boldsymbol{U}$ is the weight matrix that combines the outputs of the heads. The computing of self-attention matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ requires multiplying $\boldsymbol{Q} \in \mathbb{R}^{N\times D_h}$ and $\boldsymbol{K}^T \in \mathbb{R}^{D_h\times N}$, which is $O(N^2)$ in time and space complexity. This quadratic dependency on the sequence length has become a bottleneck for Transformers (Wang et al., 2020; Mehta et al., 2021).

2.2 Attention Patterns

Attention patterns can be classified into two general categories: (1) the diagonally shaped *Diagonal Patterns* and their particular case *Local Patterns*; (2) the vertically and horizontally shaped *Axis Patterns*, and their particular case *Global Patterns*. A pictorial representation of the categories is shown in Figure 1.

To represent the patterns intelligibly, we view such sparse attention patterns as an attention mask $\boldsymbol{B}^S \in \mathbb{R}^{N\times N}$, and treat it as an additive mask to the original self-attention mask $\boldsymbol{A}$. The new attention mask $\bar{\boldsymbol{A}}$ can be written as:

$$\bar{\boldsymbol{A}} = \boldsymbol{A} + C \cdot \boldsymbol{B}^S \quad (5)$$

where C is a large negative constant value, and $B^S_{ij} \in \boldsymbol{B}^S$ is 1 if and only if token i needs to attend to token j, and is zero otherwise.

Local vs. Diagonal Patterns Formally, we define diagonal pattern of size N_o as a set of user-designed offsets $\mathcal{O} = \{o_k\}_{k=1}^{N_o}$, and define diagonal attention mask as:

$$B^L_{ij} = 1 \quad \Longleftrightarrow \quad |i-j| \in \mathcal{O} \quad (6)$$

where $o_k \in [0, N-1]$ is the offset value that measures the distance between token i and token j.

Most sparse attention pattern designs contain a local pattern constraint on the window around each token where attention is allowed. Specifically, local patterns can be viewed as a special case of diagonal patterns, where $o_k = k$, and the offset set is $\{0\} \cup \mathcal{O}$. For simplicity, and with a slight overriding of the definition of sizes, we refer to a local attention of size N_o as a diagonal attention with offsets $\{0, 1, ..., N_o\}$.

Global vs. Axis Patterns As shown in Figure 1(d), the Axis Attention mask is composed of two separate sets $\mathcal{R} = \{r_k\}_{k=1}^{N_r}$ and $\mathcal{C} = \{c_l\}_{l=1}^{N_c}$, and we define the axis attention mask as:

$$B^G_{ij} = 1 \quad \Longleftrightarrow \quad i \in \mathcal{R} \text{ or } j \in \mathcal{C} \quad (7)$$

where $r_k \in [1, N]$ and $c_l \in [1, N]$ are offset values indicating the selected k-th row or l-th column.

Global patterns are a special case of axis patterns, where $r_k = k$ and $c_l = l$. In other words, in global patterns, there is no difference between horizontal (row) patterns and vertical (column) patterns, and picked rows and columns are at the start of the input. In most prior work, global patterns are discussed as a way to enable long range dependencies.

Random Patterns We introduce random patterns mainly for the sake of completeness. They were proposed in BigBird (Zaheer et al., 2020) and are obtained by randomly selecting some positions in the attention mask $\boldsymbol{B}^S$. We refer to the size N_r of a random pattern as the number of positions selected divided by $2N$ to approximately match the definition of the size of local and global patterns.

Prior work typically combines local and global patterns rather than committing to only using one of these broad categories. The combination of two patterns involves an *or* operation between them. Given the fixed sparse patterns defined in Eq. 6 and Eq. 7, we have the combined sparse pattern represented by:

$$\bar{\boldsymbol{A}} = \boldsymbol{A} + C \cdot (\boldsymbol{B}^L \vee \boldsymbol{B}^G) \quad (8)$$

where $\vee$ denotes the logical OR operation. Note that the size of the attention mask when local pattern size increases by one, is very similar to the

size of the mask when the size of a global pattern increases by one. We will use this property to compare local and global patterns.

2.3 Sparse Self-Attention

Several sparse attention variants have been introduced to reduce the quadratic complexity of the full attention model (Guo et al., 2019; Shi et al., 2021). Longformer (Beltagy et al., 2020) and BigBird (Zaheer et al., 2020) are two notable models that make use of pre-defined patterns. Both utilize a combination of local and global attention patterns; BigBird also introduces a randomly generated and a fixed attention pattern.

Most closely related to our approach is SparseBERT (Shi et al., 2021). The authors of SparseBERT study the importance of the main diagonal attention pattern and propose a method to learn diagonal attention. Their method learns layer-agnostic diagonal patterns during pre-training, therefore the pattern is both layer- and task-unaware. Their experiments are designed to show that the main diagonal attention is not important. In contrast we carry out experiments to show that 1) the global attention is an important component in sparse attention designs, and 2) task adaptiveness and layer-awareness can bring good improvements to sparse attention designs, 3) combining the findings above, we can design a task and layer (and also input) adaptive global sparse attention pattern, and such pattern performs extremely well even without pre-training the model to adapt the pattern.

Traditional sparse attention approaches usually learn the sparse attention by replacing the full attention with pre-defined sparse attention pattern in a transformer model, then learning to operate with such patterns via a normal pre-training and fine-tuning pipeline. Despite the promising results achieved by the recent sparse attention approaches, rarely have there been studies done to provide a good understanding of such practices. Our paper is a comprehensive study on the roles of pre-training, different attention patterns, and the power of adaptiveness of the patterns.

3 Fixed Sparse Attention: A Comprehensive Analysis

In this section, we address the first two questions related to fixed attention patterns: *(i)* is pre-training with these really necessary or does fine-tuning alone suffice, and *(ii)* what are the strengths and complementary aspects of local and global patterns.

3.1 Pretraining vs. Finetuning

We start with a suite of experiments designed to find out if sparse attention models can be successful without pre-training. We compare performance on the tasks in the GLUE benchmark of: a model with full attention in pre-training and fine-tuning; a model with the same sparse attention pattern used in pre-training and fine-tuning; and a model pre-trained with full attention (as in standard off-the-shelf models) and fine-tuned on the specific task with sparse attention.

We report performance on the eight tasks from the GLUE benchmark (Wang et al., 2019b). Six of these tasks involve predictions about the degree or type of semantic equivalence between pairs of sentences and two are single sentence tasks, one involving linguistic accessibility judgements (CoLA) and the other sentiment prediction (SST-2). The amount of data for each task varies considerably from close to 400K for MNLI (one of the language inference tasks) to 2.5K examples in the RTE task. We do not perform experiments on the WNLI task, which contains fewer than one thousand samples for fine-tuning. In results presented later in the paper, the tasks are listed in decreasing order of fine-tuning data per-task.

We adopt all default training settings and hyperparameters from Huggingface (2021) for all experiments. For pre-training, we use eight Nvidia A100 GPUs and train for 1M steps with a per-device batch size of 32 on English Wikipedia[2]. We use all default configurations from bert-base-cased. We pre-train three models, one with full attention as in the official bert-base-cased and two with sparse attention patterns that we describe below.

For fine-tuning, we use four Nvidia A100 GPUs and train for 30k steps with a per-device batch size of 32 (effectively, each device runs about three epochs over the largest dataset, MNLI). Compared to the default setting of using one device, this guarantees the model can learn to converge from a full attention model to a sparse attention one.

In this section, we consider these patterns:

- **Full** is the full attention pattern as in traditional transformer models.
- **Local + Global** are the patterns used for Longformer. We use a subscript to indicate the size of the pattern. For example **Local**$_2$ **+ Global**$_2$

[2]wikipedia/20200501.en from huggingface datasets.

Table 1: Comparison of pre-trained fixed sparse attention patterns designs with fine-tuned only patterns. For the metrics, Acc stands for Accuracy, F_1 is the F_1 score, Mcc stands for Matthews correlation coefficient and Spr stands for Spearman's rank correlation. All metrics are measured out of 100 (percent), and the higher the better. The datasets are sorted by training set size, from largest (MNLI) to the smallest (RTE).

Dataset	MNLI	QQP	QNLI	SST-2	COLA	STS-B	MRPC	RTE
Metric	Acc (mm)	F_1	Acc	Acc	Mcc	Spr	F_1	Acc
Full Pattern (pre-train & fine-tune)	82	87	90	91	48	87	90	60
$Local_2$ + $Global_2$ (pre-train & fine-tune)	77	85	86	89	41	52	80	54
$Local_2$ + $Global_2$ (fine-tune)	75 (↓ 2)	78(↓ 7)	82(↓ 4)	89(↓ 0)	44(↑ 3)	29(↓ 23)	76(↓ 4)	51(↓ 3)
$Local_2$ + $Global_1$ + $Random_1$ (pre-train & fine-tune)	77	83	83	89	44	45	78	55
$Local_2$ + $Global_1$ + $Random_1$ (fine-tune)	75(↓ 2)	81(↓ 2)	80(↓ 3)	88(↓ 1)	40(↓ 4)	19(↓ 26)	78(↓ 0)	53(↓ 2)

Table 2: Experiment on the Text dataset in LRA. We vary the size of the Local Pattern with or without Global Patterns. "Pf." means the performance.

w/o Global Pattern		**w/ Global Pattern**	
Local Pattern	Pf.	Local Pattern	Pf.
512	62.80	512	61.73
128	57.72	128	63.12
16	55.58	16	71.34
2	52.88	2	77.62

stands for a Longformer that contains a local pattern of size 2 and a global pattern of size 2.

- **Local + Global + Random** are the patterns used for BigBird. Similarly, we use **$Local_2$ + $Global_1$ + $Random_1$** to denote a combination of local pattern of size 2, global pattern of size 1, and random pattern of size 1.

The last two patterns are also used in SparseBERT (Shi et al., 2021)[3].

Table 1 shows our comparison between fine-tuning only approach and pre-training approach for $Local_2$ + $Global_2$ and $Local_2$ + $Global_1$ + $Random_1$. The table also gives performance measures for the model using full attention. Performance drops for the sparse compared to full attention models. However the difference between the fine-tuning only approach and the pre-training sparse attention approach is not that big. Notably for the acceptability judgements task (CoLA), the fine-tuned sparse attention model without a random component, results are 3 points higher than for the respective pre-trained model; performance is the same for the fine-tuned only and pre-trained model for the sentiment task (SST-2). The biggest gap in performance is for the STS-B, which requires predictions about the degree of similarity on a five point scale between pairs of sentences. For this task already switching from full to sparse attention leads to a dramatic drop in performance. The average drop of performance across the task excluding this outlier is just under 3 absolute performance points.

For the sparse attention patterns with a random component, the pre-trained version is on average 2 absolute performance points better than the fine-tuned only model (again after the excluding the outlier for the STS-B task).

3.2 Comparing Local and Global Patterns

Global patterns have been somewhat neglected. For example, in the Long Range Arena (LRA) benchmark (Tay et al., 2021), the Longformer baseline does not include a global pattern.

In Table 2 we present a comparison between local patterns alone and a combination of local and global patterns on the Text dataset in the LRA benchmark. The comparison reveals the possible reason why partial evidence may suggest that adding global patterns is not helpful but that more complete evidence indicates that a combination of local and global patterns yields substantial benefits.

The first row of Table 2, shows that performance with global patterns and a local pattern of size 512 actually is a bit worse than without the global patterns. However, subsequent rows in the table reveal that as we decrease the size of the local pattern while keeping the global pattern, performance improves. Performance can reach as high as 77.62 with the global patterns, while the best performance from other baselines reported in the LRA benchmark paper is about 65.90. Global patterns bring unique information that local patterns do not capture and they should be included in future sparse attention pattern designs or baseline comparisons.

We further empirically compare local and global patterns and evaluate the performance of models with different degrees of focus on the two patterns

[3]The use a Random Pattern of size 2. But our definition of Random Pattern selects two times more positions than theirs, so the expected pattern size is still the same

in Figure 2. To obtain the model's performance with a certain pattern, we start with a pre-trained full attention model and fine-tune it on the datasets with the sparse pattern. We compare models that focuses on vastly different amount of local and global patterns, while controlling the overall sparsity of the attention pattern. Comparing local-pattern only models with global-pattern only models would be naive, given that most prior approaches to sparse attention combine the two. In our experiments we consider models with a baseline size of two on both local and global patterns. Then, to analyze how the global pattern affects performance, for example, we fix the size of the local pattern to be 2 and vary the size of global patterns from 1 to 8. A similar set of experiments is done for the local patterns. Recalling the previous observation that we can compare local and global attention patterns with the same size, the experiments with different focus on local and global patterns can be compared.

We present experiments only for the three tasks with the largest amount of fine-tuning data in the GLUE benchmark. Figure 2 shows that, for both types of patterns, increasing the size of the patterns from the base size improves the performance. However, the areas of improvement are different on different tasks for local and global patterns. We can see that for MNLI and QNLI, increasing global patterns is more helpful than increasing local ones, while for QQP, the local patterns are more helpful. Intuitively, this is because different tasks require differing information types for language understanding — QQP requires more local information to distinguish the sentence pairs than MNLI and QNLI.

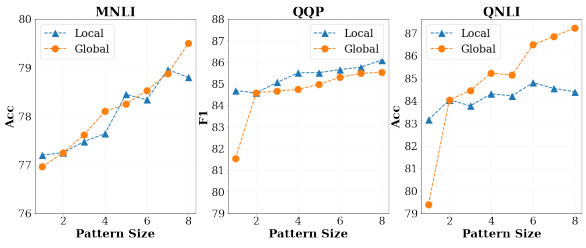


Figure 2: Comparison of Local Attention Pattern and Global Attention Pattern. We experiment with two sets of models, the first of 8 models of different sizes of local patterns and the second set of 8 models of different sizes of global patterns.

4 Beyond Fixed Sparse Attention

In this part, we discuss the importance of adaptiveness and propose an adaptive axis attention pattern.

4.1 Adaptiveness of Patterns

In the previous section we discussed evidence that global patterns and local patterns contribute differently to performance in different tasks. Should we then design *different patterns* for *different tasks*, and how can we do so? Moreover, given that different layers of BERT capture different linguistic knowledge (Clark et al., 2019; Michel et al., 2019; Kovaleva et al., 2019; Li et al., 2019)—should the patterns be *adaptive* to the *layers* as well?

We set out to study whether such adaptations to task and layer will indeed lead to better perfroamnce. To this end, we generalize SparseBERT(Shi et al., 2021) to suit our needs and conduct experiments with it. SparseBERT as originally introduced learns a diagonal attention pattern (along with a fixed global pattern) model during pre-training. The learned model is applied to downstream tasks, keeping the patterns learned during pre-training fixed. However, the attention pattern learning aspect of their approach is applicable to fine-tuning as well. In our work we make use of it to train diagonal attention pattern models during fine-tuning only, thus allowing the model to learn different patterns for different tasks.

Before proceeding with these comparisons, we introduce the notion of attention sparsity and discuss a controllable method for obtaining models with similar sparsity levels. This is necessary for a meaningful comparison of sparse attention approaches, because in general reductions from full to sparse attention leads to drop in performance, as we saw for example in the tasks from the GLUE benchmark.

Sparsity Sparsity measures the size of the sparse attention (fixed or learned) when compared with the full attention. The sparsity used in (Shi et al., 2021) is defined as: $1 - |\boldsymbol{B}^S|/N^2$, where $|\boldsymbol{B}^S| = |\{(i,j)|B^S_{ij} \neq 0\}|$ is the number of ones in the sparse attention mask matrix $\boldsymbol{B}^S$. This definition is suitable for patterns that are fixed during fine-tuning. In our work, different tasks may yield different patterns. Therefore, we propose a generalized definition of sparsity:

$$\rho = \frac{1}{|\mathcal{D}|Lh} \sum_{i=1}^{|\mathcal{D}|} \left(\sum_{l=1}^{L} \sum_{a=1}^{h} \left(1 - \frac{|\boldsymbol{B}^S_{i,l,a}|}{N_i^2} \right) \right) \quad (9)$$

where $|\mathcal{D}|$ is the size of the dataset $\mathcal{D}$, N_i denotes the sequence length of the i-th input sample, which can be different from the fixed value (128) in Shi et al. (2021) [4], L the number of transformer layers, and h number of attention heads. $\boldsymbol{B}^S_{i,l,a}$ refers to the sparse attention mask matrix for the i-th input sample, l-th layer, and a-th attention head.

The sparsity definition in Eq. 9 has several key advantages: 1) It is applicable when attention patterns are different across instances, layers, and attention heads rather than fixed; 2) It uses the actual sequence (text) length, more truthfully reflecting how much attention is used when processing a specific input. The original sparsity definition is involves only the model-wise maximum sequence length. For example, a local pattern of size 2 has a sparsity value[5]: $1 - 5/N + 6/N^2$. This is undesirable because by just changing the model maximum sequence length, sparsity changes without impacting the performance on individual inputs.

Sparsity Controllable Training Controlling the target sparsity of self-attention is beneficial for comparison purposes. Given the fixed target sparsity ρ_{target}, we define the training objective as:

$$\mathcal{L}_{\text{All}} = \underbrace{\mathcal{L}_{\text{task}}}_{\text{Finetune Loss}} + \underbrace{\alpha \cdot \max(0, \rho_{\text{target}} - \rho)}_{\text{Sparsity Loss}} \quad (10)$$

where the first term ($\mathcal{L}_{\text{task}}$) denotes the objective loss for the fine-tuning task, ρ is the sparsity during training, α is an amplifying factor of the sparsity loss. The hinge loss encourages the runtime sparsity to be close to the desired sparsity. In our experiments, we consider two variants of α: 1) a constant value and 2) an increasing linear value that reaches its maximum at half of the epochs and then stays constant. We pick the best variant of α among the two and gradually increase its absolute value until the target sparsity has been reached.

Results In our experiment, we consider three diagonal attention pattern models that have different levels of adaptiveness:

- **Fixed** is a fixed diagonal attention pattern model, where the pattern is copied from a pre-trained SparseBERT model.
- **Task-adaptive** is a model that learns the attention pattern during fine-tuning, therefore is different for different tasks.
- **Task- & Layer-adaptive** further allows different layers of the model to learn different patterns.

All attention patterns are paired with global attention, and the results are reported in Table 3. We can see clearly that the task-adaptive model is better than the fixed model, as the patterns are learned from the tasks. Further, adding adaptiveness into the layers also brings a small boost to the performance. These experiments show that having the patterns adaptive and learnable is beneficial for sparse pattern designs.

4.2 Adaptive Axis Attention

We show experiments highlighting the strengths of global attention (in Section 3.2) and of allowing adaptiveness of attention (in Section 4.1). To combine these strengths, we design a novel attention pattern that incorporates the learning of Axis Patterns, a more general form of Global Patterns. Intuitively, we want the model to learn which input tokens are important and focus on rows or columns in the attention map associated with these tokens.

Specifically, we learn a row/column-wise importance value for each token representation $\boldsymbol{x}_n \in \boldsymbol{X}$ through a fully-connected layer. This importance value is fed into a Gumbel-sigmoid operation to retrieve a 0/1 indicator:

$$\tilde{I}^k_n = f_{\text{Gumbel-sigmoid}}(f^k_{\text{FC}}(\boldsymbol{x}_n)),\ k \in \{r, c\} \quad (11)$$

where $\tilde{I}^k_n$ is the importance indicator for n-th token retrieved by the Gumbel-sigmoid operation, k indicates the column (c) or row (r). Specifically, $\tilde{I}^r_n = 1$ indicates that all attention values in row n of the attention matrix are kept. Equivalently, this means this token can attend to all other tokens in the input. Similarly, $\tilde{I}^c_n = 1$ indicates column n of the attention matrix is kept.

Given the importance indicators $\tilde{I}^r_i$ and $\tilde{I}^c_j$, the axis pattern $B^S_{ij} \in \boldsymbol{B}^S$ can be calculated as follows:

$$B^S_{ij} = \tilde{I}^r_i + \tilde{I}^c_j - \tilde{I}^r_i \cdot \tilde{I}^c_j \quad (12)$$

where $B^S_{ij} = 1$ means either the importance indicator for row i or column j is on[6]. Usually, this adaptive axis attention pattern is also paired up with some local patterns, especially the main diagonal local attention. This is to ensure that no rows are empty, which is needed because self-attention includes operations such as softmax and linear combinations, which are undefined over empty values.

[4] The consideration of the actual input sequence size makes our sparsity levels lower than the sparsity in SparseBERT.

[5] $1-\big(N+2(N-1)+2(N-2)\big)/N^2 = 1-5/N+6/N^2$.

[6] Such calculation allows gradient to backpropogate to the fully connected layer that calculates the importance value.

Table 3: Comparison of learnable diagonal attention models that have different levels of adaptiveness. ρ is the sparsity value defined in Eq. 9. We also show the relative difference from each row to the previous row.

Adaptiveness	MNLI		QQP		QNLI		SST-2		COLA		STS-B		MRPC		RTE	
	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.
Fixed	86	70	85	79	88	72	83	89	75	34	85	28	88	79	88	50
Task-adaptive	86	74(↑ 4)	87	79(↑ 0)	89	75(↑ 3)	83	83(↓ 6)	81	38(↑ 4)	85	36(↑ 8)	88	77(↓ 2)	89	56(↑ 6)
Task & Layer-adaptive	86	76(↑ 2)	85	81(↑ 2)	89	77(↑ 2)	83	86(↑ 3)	78	35(↓ 3)	86	38(↑ 2)	89	77(↑ 0)	89	55(↓ 1)

Following designs in Section 3.2, we pair it up with a local pattern of size 2. This adaptive axis pattern is also learned separately for each layer and different tasks, taking full advantage of the benefits of adaptiveness. Similar to the adaptive diagonal attention patterns introduced in Section 4.1, we optimize the model with Eq. 10.

4.3 Experiments with AAA

In this section, we verify empirically the effectiveness of our proposed AAA. Quantitative results are listed in Tables 4, 5, and 7.

Experiment Settings In this section, our experiments follow the setting described in Section 3.1. We also include some other patterns to show that findings are stable for different combinations:

- **Local$_3$ + Global$_1$** is a variant of the Longformer-like pattern in which we increase the size of the local attention but decrease global attention size. As discussed previously, this results in a model with comparable capacity but may provide different benefits.
- **Local$_1$ + Global$_1$ + Random$_2$** is similarly a variant for BigBird. Here we increase the size of the random patterns, so the resulting sparsity values are different from the corresponding Local$_2$ + Global$_1$ + Random$_1$ attention.
- **Diagonal + Global$_1$** represents patterns coming from SparseBERT. It combines a learned diagonal pattern with global pattern of size 1.

AAA outperforms fix pattern models We compare our AAA with several fixed attention patterns. We optimize AAA with Eq. 10, and set different targets of the final sparsity values ρ_{target} for each task. For all baselines, we report the sparsity values and performance on the development set in Table 4. We first point out an encouraging result related to sparsity: AAA exhibits a similar sparsity value in the development set as in the training set. For all datasets, AAA is able to reach the desired, and sometimes slightly better, sparsity values. Next, we compare the performance of the models. For all tasks, our model performs better than the fixed pattern approaches. For most tasks, the improvement is large. This success further confirms the strength of adaptiveness in designing attention patterns.

AAA rivals pre-trained pattern models Now we also compare with the pre-trained variant of the adaptive diagonal attention model. Rather than starting from a pre-trained BERT model with full attention, we pre-train a sparse adaptive diagonal attention model. The results, along with pre-trained variants of fixed pattern models, are shown in Table 5. We already know, from Section 3.1, that the pre-trained variants of fixed patterns improve a moderate amount of performance. The performance for the adaptive patterns is also comparable to the fine-tuned only AAA on most tasks. Furthermore, on the STS-B task where fixed patterns suffered a great drop in performance, AAA shows very strong performance. The pre-trained version of the diagonal patterns shows strong performance and is better than our model in most tasks. Overall, we show that AAA achieves a strong performance that is comparable to other sparse patterns that involve pre-training.

AAA focuses more on columns than rows AAA separates the importance learning of row-wise patterns and column-wise patterns. After fine-tuning, we examine for each input sample during evaluation the percentage of important tokens selected for rows and for columns. Table 6 shows the results. There are much more important column tokens than important row tokens. This means that for axis patterns, tokens that other tokens attended to are more important than tokens that attend to other tokens. This finding is another indication that fixed (global) patterns are not ideal.

AAA is orthogonal to MobileBERT Improving the efficiency of transformers is needed for real-world applications and several approaches have been developed to improve efficiency on resource-limited devices, such as reducing attention heads and hidden dimensions. To show that gains from

Table 4: Comparison of fixed sparse attention map designs with ours. In the first row, we show the performance when using the unchanged full attention. Since our method AAA has the ability to learn to a fixed sparsity ratio, we train our model to adapt to the specific sparsity ratio on each task when compared to other different fixed patterns.

Fine-tuning Pattern	MNLI		QQP		QNLI		SST-2		COLA		STS-B		MRPC		RTE	
	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.
Full	0	84	0	88	0	91	0	92	0	54	0	88	0	89	0	62
Local_2 + Global_2	76	77	70	85	82	84	64	90	34	48	70	42	83	78	84	53
Local_3 + Global_1	76	77	70	83	82	80	63	89	34	48	70	31	83	79	84	53
AAA	77	81(↑ 4)	73	85(↑ 0)	82	86(↑ 2)	65	89(↓ 1)	36	56(↑ 8)	72	79(↑ 37)	86	83(↑ 5)	85	58(↑ 5)
Local_2 + Global_1 + Random_1	80	77	76	84	85	79	70	90	45	44	75	44	86	82	87	56
AAA	81	80(↑ 3)	82	85(↑ 0)	85	86(↑ 7)	84	89(↓ 1)	76	50(↑ 6)	82	75(↑ 31)	89	80(↓ 2)	89	56(↑ 0)
Local_1 + Global_1 + Random_2	85	77	81	84	88	80	77	90	57	33	81	49	89	79	89	49
AAA	86	80(↑ 3)	86	85(↑ 1)	88	86(↑ 6)	84	89(↓ 1)	76	50(↑ 17)	86	67(↑ 18)	89	80(↑ 1)	89	56(↑ 7)

Table 5: Comparison of pretrained sparse attention map designs with ours.

Pattern	MNLI		QQP		QNLI		SST-2		COLA		STS-B		MRPC		RTE	
	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.	ρ	Pf.
Local_2 + Global_2 (pre-train & fine-tune)	76	77	70	85	82	86	63	89	34	41	70	52	83	80	84	54
AAA (fine-tune)	77	79(↑ 2)	72	84(↓ 1)	83	84(↓ 2)	66	89(↑ 0)	48	41(↑ 0)	71	81(↑ 29)	86	85(↑ 5)	87	53(↓ 1)
Local_2 + Global_1 + Random_1 (pre-train & fine-tune)	80	77	76	83	85	83	70	89	45	44	75	45	86	78	87	55
AAA (fine-tune)	81	80(↑ 3)	78	84(↑ 1)	86	84(↑ 1)	71	88(↓ 1)	56	40(↓ 4)	76	80(↑ 35)	89	84(↑ 6)	90	53(↓ 2)
Diagonal + Global_1 (pre-train & fine-tune)	86	79	85	85	88	86	83	90	75	38	85	64	88	84	88	54
AAA (fine-tune)	87	78(↓ 1)	86	83(↓ 2)	88	84(↓ 2)	84	87(↓ 3)	77	36(↓ 2)	85	75(↑ 11)	91	86(↑ 2)	90	50(↓ 4)

Table 6: Percentage of row-wise important tokens and column-wise important tokens.

	MNLI	QQP	QNLI		MNLI	QQP	QNLI
row	0.8	0.6	1.0	column	1.6	1.3	1.7

Table 7: AAA can be integrated with MobileBERT.

Model	MNLI		QQP		QNLI	
	ρ	Pf.	ρ	Pf.	ρ	Pf.
BERT	0	84	0	87	0	91
BERT + AAA	77	81	73	85	82	86
MobileBERT	0	83	0	87	0	90
MobileBERT + AAA	78	78	74	83	83	86

our AAA are compatible with such approaches, we compare AAA with MobileBERT (Sun et al., 2020) in Table 7. The amount of performance dropped with the same sparsity is similar for both BERT and MobileBERT. Therefore, AAA's performance is not impeded by a model that is already compressed to reduce attention heads or hidden dimensions and can be integrated into such a model easily and effectively.

5 Conclusion

In this paper, we present a comprehensive analysis of sparse attention patterns. We demonstrate that while pre-training with sparse attention does improve performance on many tasks, using sparse attention only in fine-tuning sacrifices a bit of performance for a big gain in time and computational resource savings.

We compare the popular local and global patterns and conclude that either type provide an advantage depending on the task. We also show that allowing sparse patterns to be adaptive to the task or layers improves performance. Finally we present AAA which incorporated all these insights and learns important tokens during fine-tuning. Our model is consistently and considerably better than other sparse attention pattern models and rivals models that require extensive pre-training. For future work, we anticipate to integrate the adaptive diagonal pattern with our adaptive axis pattern to construct a fully learnable pattern.

Ethical Considerations

The work presented in this paper deals with foundations aspects of representation learning for language tasks. We present experiments on core tasks dealing with textual semantic equivalence, which do not pose ethical concerns.

Acknowledgments

This work was supported in part by Adobe Research. We thank anonymous reviewers and program chairs for their valuable and insightful feedback. Zihan Wang is supported by the UCSD Jacob School of Engineering Fellowship and the UCSD Halıcıoğlu Data Science Fellowship.

References

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

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of bert's attention. *CoRR*, abs/1906.04341.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Linyuan Gong, Di He, Zhuohan Li, Tao Qin, Liwei Wang, and Tieyan Liu. 2019. Efficient training of bert by progressively stacking. In *ICML*.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-transformer. In *NAACL*.

Huggingface. 2021. https://github.com/huggingface/transformers/tree/master/examples/pytorch.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *ICLR*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *EMNLP-IJCNLP*.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *NeurIPS*.

Sachin Mehta, Marjan Ghazvininejad, Srinivasan Iyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Delight: Deep and light-weight transformer. In *ICLR*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *NeurIPS*.

Han Shi, Jiahui Gao, Xiaozhe Ren, Hang Xu, Xiaodan Liang, Zhenguo Li, and James Tin-Yau Kwok. 2021. Sparsebert: Rethinking the importance analysis in self-attention. In *ICML*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic BERT for resource-limited devices. *CoRR*, abs/2004.02984.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021. Long range arena : A benchmark for efficient transformers. In *ICLR*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. *CoRR*, abs/2009.06732.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

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

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *NeurIPS*.