Abstract

Transformers have been essential to pretraining success in NLP. While other architectures have been used, downstream accuracy is either significantly worse, or requires attention layers to match standard benchmarks such as GLUE. This work explores pretraining without attention by using recent advances in sequence routing based on state-space models (SSMs). Our proposed model, Bidirectional Gated SSM (BiGS), combines SSM layers with a multiplicative gating architecture that has been effective in simplified sequence modeling architectures. The model learns static layers that do not consider pair-wise interactions. Even so, BiGS is able to match BERT pretraining accuracy on GLUE and can be extended to long-form pretraining of 4096 tokens without approximation. Analysis shows that while the models have similar average accuracy, the approach has different inductive biases than BERT and scales more efficiently to longer sequences.

1 Introduction

Transformers are the de facto model architecture for NLP pretraining (Vaswani et al., 2017). Since BERT (Devlin et al., 2018), they have proven central to NLP tasks with their ability to learn effectively on large unlabeled datasets. Specifically, the use of attention as a central routing component seems to be critical to empirical success on downstream tasks. Other architectures have been proposed but require attention layers for high-accuracy (Tay et al., 2020b; Lee-Thorp et al., 2021).

Is the centrality of attention in pretraining due to inductive bias or computational convenience? This question is complicated by the properties of common sequence routing layers: recurrent neural network (RNN) models do not scale as well as attention, whereas convolutional neural networks (CNNs) can not easily model long-distance dependencies.

State-space models (SSMs) for deep learning provide a promising alternative. Recent works show that SSMs are a competitive architecture for long-range sequence modeling (Gu et al., 2021). SSMs achieve strong results on speech generation (Goel et al., 2022) and on the Long Range Arena benchmark (Tay et al., 2020a) outperform standard and long-range transformer architectures (Gu et al., 2021; Gupta, 2022; Gu et al., 2022; Smith et al., 2022). In addition to improving accuracy, SSM-based routing does not have quadratic complexity as the length of the sequence grows. Concretely, the model provides a way to achieve RNN-like long-range dependencies with CNN-like training speed.

This work proposes an architecture for applying SSMs using a Bidirectional Gated SSM (BiGS) model for BERT-style pretraining. BiGS uses SSM-routing at its core as a replacement for attention. However, this change alone significantly degrades the representational capacity of the model. To target this issue, we develop a multiplicative gating architecture (Dauphin et al., 2017; Hua et al., 2022; Mehta et al., 2022). In combination, this leads to a simpler routing approach that remains surprisingly effective at modeling necessary interactions.

Experiments compare SSMs to standard NLP pretraining. While we find that SSMs by themselves underperform on NLP pretraining tasks, BiGS is able to match the performance of a BERT model when trained on the same data in a controlled setting. By additionally pretraining on longer-length instances, the model is able to grow without approximation to extend to input sequences of length 4,096. Analysis shows the importance of multiplicative gating in fixing specific issues of variable-length textual input. All models from this work will be available open-source (Apache 2.0 license) upon release.
2 Related Work

Prior to BERT, promising pretraining approaches for learning contextual representations were learned using RNN-based models (McCann et al., 2017; Peters et al., 2018). While important precursors, their accuracy did not scale with data or compute as well as Transformers. This gap remains even when back-porting best-practices from Transformer pretraining (Peters et al., 2019). Recently Tay et al. (2021) explored pretraining with several convolutional (CNN) variants. Results show that CNN without attention does not perform well, although they note benefits in routing speed. Lee-Thorp et al. (2021) propose FNet which replaces the attention layer with a Fourier transform. Without attention, this achieves 92-97% results on GLUE (Wang et al., 2018). Other works have used CNN-based models with multiplicative gating for NLP tasks such as machine translation (Dauphin et al., 2017). We believe BiGS is the first model to achieve BERT-level transfer learning on the GLUE benchmark without attention.

Researchers have begun to use state-space models for NLP tasks, and have primarily focused on auto-regressive language modeling. In S4 (Gu et al., 2021) and its variants (Gupta, 2022; Gu et al., 2022), researchers experimented with language modeling, achieving promising results, though slightly worse than transformers. Gated State Space adapts a SSM plus gating approach to language modeling (Mehta et al., 2022). Concurrent to this work, Dao et al. (2022b) propose H3 which closes the gap in auto-regressive language modeling, and with two attention layers outperforms transformers on OpenWebText. Finally, a related method, MEGA (Ma et al., 2022) combines exponential moving average routing with a simple attention unit to outperform transformer baselines. Our approach instead focuses on bidirectional masked language modeling and questions of downstream generalization.

3 Background

3.1 State Space Models

A state space model (SSM) is a general-purpose tool for describing the relationship between a continuous-time scalar input \( u(t) \) to scalar output \( y(t) \) by the following differential equations:

\[
x'(t) = Ax(t) + Bu(t), \quad y(t) = Cx(t) + Du(t).
\]

Figure 1: A SSM learns a one-dimensional kernel \( K \), which is convolved with the input sequence \( u \) to produce output \( y \). Unlike attention, routing is static and does not depend on the input. In BiGS, we use only two kernels per layer (forward and backward). Figure 3 shows all the kernels used in the fully trained model.

Where \( x(t) \in \mathbb{R}^N \) is a continuous-time state vector, \( x'(t) \) is its derivative, and the equation is parameterized by \( A \in \mathbb{R}^{N \times N}, B \in \mathbb{R}^{N \times 1}, C \in \mathbb{R}^{1 \times N}, D \in \mathbb{R}^{1 \times 1} \).

When applied to a discrete-time scalar input sequence \( u_1, \ldots, u_L \), the SSM equations and parameters can be discretized, leading to the following recursion,

\[
x_k = \overline{A} x_{k-1} + \overline{B} u_k, \quad y_k = \overline{C} x_k + \overline{D} u_k.
\]

Where \( \overline{A}, \overline{B}, \overline{C}, \overline{D} \) are functions of the original parameters and a discretization rate.

This equation can be computed like an RNN where \( x_k \in \mathbb{R}^N \) is a hidden state at time \( k \). Unlike an RNN though, the linearity of the recursion allows \( y_1 \ldots y_L \) to be computed directly using a convolution with precomputed kernel \( K \in \mathbb{R}^L \),

\[
K = (CB, CABA, \ldots, CA^{L-1}B), \quad y = K \ast u
\]

The process is illustrated in Figure 1. In a practical sense, after training, this kernel \( K \) fully characterizes the SSM, i.e. the model is a 1D convolution with a very long kernel.

3.2 Learning SSMs

Gu et al. (2020, 2021) demonstrate an effective approach for using SSMs in neural networks. The core insight is to propose an initialization of the transition matrix \( A \), known as HiPPO,

\[
A_{nk} = \begin{cases} 
(2n + 1)^{1/2}(2k + 1)^{1/2} & \text{if } n > k \\
\frac{n+1}{n+1} & \text{if } n = k \\
0 & \text{if } n < k 
\end{cases}
\]
This matrix yields a stable training regime that can also be efficiently trained. The full model, S4, retains the SSM ability to model long-term sequences while being more efficient than RNNs to train.

Recently, researchers (Gu et al., 2022; Gupta, 2022) have proposed simplified diagonalized versions of S4, which achieve comparable results with a simpler approximation of the original parameterization. In preliminary experiments, we used several different S4 parameterizations but did not find a significant difference in accuracy. Throughout the work, we use S4D as the parameterization.

While the specifics of SSM discretization, parameterizations, and training are beyond the scope of this work, at a high-level, we note that each variant of SSMs leads to a similar convolution form. The model can therefore be trained by backpropagation through the convolution without the serial bottleneck of RNNs, and applied without the quadratic cost of attention.

3.3 Multiplicative Gating

Gating units have been widely used to improve the performance of various architectures such as MLP, CNN, and Transformers (Dauphin et al., 2017; Shazeer, 2020; Narang et al., 2021). One example of such a gating unit is the Gated Linear Unit (GLU) which has been used effectively for CNN-based NLP systems (Dauphin et al., 2017). Let \( u \) represent an input activation. GLU first computes both a gating vector and a linear transform, \( \sigma(W_u) \) and \( V_u \) respectively. The output of the layer is then the element-wise product \( \sigma(W_u) \odot V_u \).

Recent work has shown that gating can increase the performance of models using simplified routing. Hua et al. (2022) show that linear time attention models can benefit from improved gating. Mehta et al. (2022) propose a Gated State Space architecture using gating for unidirectional SSM models. Multiplicative gating may restore some of the interaction capacity from full attention-based interactions.

4 BiGS Model

We consider two different architectures for SSM pretraining: a stacked architecture (STACK) and a multiplicative gated architecture (GATED) shown in Figure 2.

Transformer Architecture  The STACK architecture with self-attention is equivalent to the BERT / transformer model. We replace the attention block with two sequential SSM blocks to mimic the na-
ture of bi-directional self-attention.

**Gated Architecture** The **GATED** architecture is a bidirectional adaptation of the gated unit of Hua et al. (2022). Specifically, let \( X_i \in \mathbb{R}^{L \times d} \) be activations at the \( i \)-th layer where the length is \( L \), and the model size is \( d \). We use the activation GELU (Hendrycks and Gimpel, 2016) for \( \sigma \). The first stage computes,

\[
X = \text{LayerNorm}(X_i) \in \mathbb{R}^{L \times d}
\]

\[
V = \sigma(W_vX) \in \mathbb{R}^{L \times 3d}
\]

\[
F = \sigma(W_fX) \in \mathbb{R}^{L \times d}
\]

\[
B = \sigma(W_b\text{Flip}(X)) \in \mathbb{R}^{L \times d}
\]

The second stage uses 2 sequential blocks (i.e., a forward and backward SSM layer) with a multiplicative gate.

\[
U_1 = W_{u_1}\text{SSM}(F) \in \mathbb{R}^{L \times d}
\]

\[
U_2 = W_{u_2}\text{SSM}(B) \in \mathbb{R}^{L \times d}
\]

\[
U = \sigma(W_u(U_1 \otimes \text{Flip}(U_2))) \in \mathbb{R}^{L \times 3d}
\]

The third stage uses a feed-forward layer again with gating, to replace the two dense blocks in the traditional transformer architecture. We sum this output \( O \) with the original input \( X_i \) finally as the input \( X_{i+1} \) of the next layer \( i+1 \).

\[
O = W_o(U \otimes V) \in \mathbb{R}^{L \times d},
\]

\[
X_{i+1} = O + X_i \in \mathbb{R}^{L \times d}
\]

The number of parameters per layer in gated SSM is roughly \( 13d^2 \) while the number of parameters per layer in the stack is \( 12d^2 \). We compensate for this difference by using fewer gated layers.

Different from (Mehta et al., 2022), we find that the hidden dimension size of SSM layers is critical. Reducing that hidden dimension results in a notable decrease of the perplexity (−0.67) in the MLM in our 11B (short) training setting.

**SSM Layer** The SSM layer under both architectures is a map over vector sequences, \( \text{SSM}(X) : \mathbb{R}^{L \times d} \rightarrow \mathbb{R}^{L \times d} \). However, we defined SSM over scalar sequences. Past work, creates \( d \) differently parameterized SSMs for each dimension (Gu et al., 2021). Experimentally though, we found it just as practical to use the same parameterization (and therefore kernel \( \mathbf{K} \)) for each hidden dimension. This simplifies model analysis and makes the total number of SSM parameters negligible.

## 5 Experimental Setup

Experiments compare the performance of SSM-based models to attention-based models on several standard fine-tuning benchmarks. Experiments control for total parameter-size and amount of pretraining in terms of the number of tokens. All models are on the order of magnitude of BERT-Large at around 350M parameters; all GATED SSM models use 23 layers and STACK models 24 to match parameter count. In order to run ablation tests, we consider three different pretraining scales: 11B (short), 29B (medium), and 97B (full) tokens. Models and architectures are roughly similar in training speed at this length. The 11B (short) training scale is roughly equivalent to the "24h BERT" setting typically used in research studies (Izsak et al., 2021). Full training is closer to the original BERT model which was trained on 128B tokens.

For all pretraining, we follow the training data and masking strategy of Izsak et al. (2021). Since RoBERTa (Liu et al., 2019) shows it does not hurt accuracy, we use only masked language modeling and not next-sentence prediction. We preprocess and mask tokens offline for all models for consistency, with maximal sequence length to be 128. We use a grid search on perplexity to select configurations of weight decay and learning rate; other hyperparameters follow Izsak et al. (2021). For SSM, we use a cosine decay learning rate scheduler, which starts at 0, warms up to the peak learning rate, and then decays back (Gu et al., 2021).

Pretraining is done with length 128 token sequences. In order to adapt to longer sequences we apply continued pretraining. To adapt to 512 tokens for the SQuAD dataset, we follow the protocol of Wettig et al. (2022) and train on longer sequences of the same pretraining dataset. To adapt to 4,096 tokens, we follow the Longformer (Beltagy et al., 2020) protocol and continue training the BiGS model on the text of length up to 4,096 tokens long, for 10k more steps using their proposed training corpus of longer documents. For 4,096 tokens, we also use a smaller BiGS model (around 130M) so that it is comparable in size Longformer-base and BART-base models. We note that Longformer (LED) and BART are based on superior underlying models that are trained significantly longer.

For downstream task fine-tuning, we average the embeddings of non-padding tokens to create the sentence representation and add a classification head. This approach yields superior performance
<table>
<thead>
<tr>
<th>Arch / Route</th>
<th>MNLI</th>
<th>QNLI</th>
<th>QQP</th>
<th>RTE</th>
<th>SST2</th>
<th>MRPC</th>
<th>COLA</th>
<th>STS$_B$</th>
<th>AVG</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Short Training / ∼11B Tokens</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BERT STACK / ATT</td>
<td>82.7</td>
<td>90.1</td>
<td>87.7</td>
<td>76.8</td>
<td>91.5</td>
<td>90.8</td>
<td>58.6</td>
<td>88.6</td>
<td>83.3</td>
</tr>
<tr>
<td>STACK / SSM</td>
<td>78.4</td>
<td>83.5</td>
<td>85.6</td>
<td>60.5</td>
<td>91.6</td>
<td>83.9</td>
<td>53.1</td>
<td>81.3</td>
<td>77.2</td>
</tr>
<tr>
<td>GATED / ATT</td>
<td>82.2</td>
<td>88.3</td>
<td>87.4</td>
<td>71.7</td>
<td>91.3</td>
<td>88.5</td>
<td>58.8</td>
<td>86.5</td>
<td>81.8</td>
</tr>
<tr>
<td>BiGS GATED / SSM</td>
<td>82.6</td>
<td>89.2</td>
<td>87.6</td>
<td>73.8</td>
<td>92.8</td>
<td>88.9</td>
<td>63.2</td>
<td>88.4</td>
<td>83.3</td>
</tr>
<tr>
<td><strong>Medium Training / ∼29B Tokens</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BERT STACK / ATT</td>
<td>85.0</td>
<td>90.9</td>
<td>87.9</td>
<td>80.5</td>
<td>93.0</td>
<td>90.9</td>
<td>60.8</td>
<td>89.2</td>
<td>84.8</td>
</tr>
<tr>
<td>STACK / SSM</td>
<td>80.1</td>
<td>86.5</td>
<td>87.2</td>
<td>65.6</td>
<td>92.3</td>
<td>86.5</td>
<td>56.5</td>
<td>83.4</td>
<td>79.8</td>
</tr>
<tr>
<td>GATED / ATT</td>
<td>83.5</td>
<td>90.2</td>
<td>87.6</td>
<td>72.0</td>
<td>91.7</td>
<td>88.7</td>
<td>61.6</td>
<td>87.5</td>
<td>82.9</td>
</tr>
<tr>
<td>BiGS GATED / SSM</td>
<td>84.5</td>
<td>90.2</td>
<td>88.3</td>
<td>78.6</td>
<td>94.4</td>
<td>89.6</td>
<td>63.9</td>
<td>89.3</td>
<td>84.8</td>
</tr>
<tr>
<td><strong>Full Training / ∼97B Tokens</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BiGS GATED / SSM</td>
<td>86.2</td>
<td>90.9</td>
<td>88.3</td>
<td>79.4</td>
<td>94.6</td>
<td>89.5</td>
<td>67.3</td>
<td>90.1</td>
<td>85.8</td>
</tr>
<tr>
<td><strong>Non-Attention Based Pretraining</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNN STACK / CNN</td>
<td>∼75</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>92.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ELMo STACK / RNN</td>
<td>68.6</td>
<td>71.2</td>
<td>84.3</td>
<td>53.4</td>
<td>91.5</td>
<td>70.5</td>
<td>44.1</td>
<td>82.3</td>
<td>68.7</td>
</tr>
<tr>
<td>FNet$_L$ STACK / FNT</td>
<td>78.0</td>
<td>85.0</td>
<td>85.0</td>
<td>69.0</td>
<td>94.0</td>
<td>88.0</td>
<td>-</td>
<td>84.0</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arch / Route</th>
<th>MNLI</th>
<th>QNLI</th>
<th>QQP</th>
<th>RTE</th>
<th>SST2</th>
<th>MRPC</th>
<th>COLA</th>
<th>STS$_B$</th>
<th>AVG</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GLUE Test Result</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BERT$_1$ STACK / SSM</td>
<td>86.7/85.9</td>
<td>92.7</td>
<td>72.1</td>
<td>70.1</td>
<td>94.9</td>
<td>88.9</td>
<td>60.5</td>
<td>86.5</td>
<td>79.6</td>
</tr>
<tr>
<td>BERT$_2$ STACK / SSM</td>
<td>86.0/85.2</td>
<td>92.6</td>
<td>72.0</td>
<td>78.3</td>
<td>94.5</td>
<td>89.9</td>
<td>60.9</td>
<td>87.5</td>
<td>83.0</td>
</tr>
<tr>
<td>BiGS GATED / SSM</td>
<td>86.1/85.0</td>
<td>91.6</td>
<td>71.2</td>
<td>77.6</td>
<td>94.9</td>
<td>88.7</td>
<td>64.4</td>
<td>87.5</td>
<td>83.0</td>
</tr>
</tbody>
</table>

Table 1: GLUE Results. (Top) Comparison of different architectures and routing in a controlled setting (Izsak et al., 2021). See Figure 2 for details. We fine-tune RTE, MRPC, and STS-B from a MNLI checkpoint following the convention by (Izsak et al., 2021). We average results of six runs and report accuracy for MNLI, QNLI, RTE, SST-2 and F1 score for QQP, MRPC and Matthew’s correlation for CoLA and Spearman’s correlation for STS-B. All models are comparable to BERT-Large in size. (Bottom) Reported comparable results for other non-attention-based pretraining models based on CNNs, LSTMs and FNet (Peters et al., 2018; Tay et al., 2021; Lee-Thorp et al., 2021; Wang et al., 2018). BERT$_1$ represents the official BERT result (Devlin et al., 2018), and BERT$_2$ represents the result using an MNLI checkpoint for other NLI tasks (Izsak et al., 2021). We use − to denote those results were not reported by previous research.

Our SSM implementation is based on the Annotated S4$^1$ (Rush, 2022), and our pretraining uses the template from Hugging Face Transformers$^2$ (Wolf et al., 2020). We experimented with variants of SSMs and found they performed similarly; experiments use S4D (Gu et al., 2022) for simplicity. Note that for a fair comparison, we keep the size of the gated architecture comparable to a stacked architecture and our BERT implementation.

### 6 Results

#### 6.1 GLUE

Table 1 (Top) shows the main results for different pretrained models on the GLUE benchmark. In short and medium training, we note that the STACK architecture is significantly better with attention than with SSM-routing. However, with the GATED architecture, the SSM achieves competitive results. To confirm this is not simply from a better architecture, we try gating with attention but find it does not improve. On full training, BiGS continues to improve in accuracy.

Table 1 (Bottom) compares the BiGS architecture to other reported results on GLUE. First, we compared to using the [CLS] token as the sentence representation.

---

$^1$https://srush.github.io/annotated-s4

$^2$https://github.com/huggingface/transformers
Table 2: SQuAD F1 Dev Results. Models are trained by adapting full 128 token models to 512 tokens (Wettig et al., 2022).

<table>
<thead>
<tr>
<th>Model</th>
<th>Length</th>
<th>QALT (T/H)</th>
<th>CNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT (512)</td>
<td></td>
<td>90.9</td>
<td></td>
</tr>
<tr>
<td>BERT (128 → 512)</td>
<td>87.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BiGS (128 → 512)</td>
<td>89.5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: SCROLLS Encoder Test set results. Baseline models are both encoder-decoder models, one based on Longformer (LED) (Beltagy et al., 2020) and the other on BART (Lewis et al., 2019). Inputs are truncated at length.

<table>
<thead>
<tr>
<th>Model</th>
<th>Length</th>
<th>QALT (T/H)</th>
<th>CNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED (162M)</td>
<td>1024</td>
<td>26.6/27.2</td>
<td>73.4</td>
</tr>
<tr>
<td>LED (162M)</td>
<td>4096</td>
<td>26.6/27.3</td>
<td>71.5</td>
</tr>
<tr>
<td>LED (162M)</td>
<td>16384</td>
<td>25.8/25.4</td>
<td>71.5</td>
</tr>
<tr>
<td>BART (140M)</td>
<td>256</td>
<td>26.0/25.8</td>
<td>69.8</td>
</tr>
<tr>
<td>BART (140M)</td>
<td>512</td>
<td>26.8/27.4</td>
<td>71.6</td>
</tr>
<tr>
<td>BiGS (130M)</td>
<td>128</td>
<td>32.3/30.0</td>
<td>68.7</td>
</tr>
<tr>
<td>BiGS (130M)</td>
<td>4096</td>
<td>32.8/31.7</td>
<td>71.4</td>
</tr>
</tbody>
</table>

We also apply BiGS to SQuAD (Rajpurkar et al., 2016). SQuAD requires extending the length of the model from 128 to 512 tokens through additional training. We report the F1 score in Table 2. We see that BiGS outperforms BERT when adapted with this procedure (Wettig et al., 2022). We note that both of these results underperform original BERT SQuAD results.

Figure 3: Complete SSM routing learned in BiGS. Shows forward and backward kernels $K$ at each layer (0-22). Values indicate the absolute value of the contribution of each relative position (-10, ..., 10) cropped from the full $2 \times 128$. Min-max scaling of absolute values is used for visual normalization.

Figure 4: Change in SSM kernel after finetuning. Shows $K$ after pretraining and after MNLI finetuning for Layer 14, Layer 18, and Layer 17 over all relative positions (-128, ..., 128).

6.2 Long-Form Classification

An advantage of SSM-based routing is that models can extend to longer-ranges without requiring approximation. To adapt to longer range classification, we continue pretraining on longer data (4,096). Table 3 shows results on encoder-only experiments in SCROLLS (Shaham et al., 2022), a recent long-range language modeling benchmark. We can compare the model to Longformer Encoder-Decoder (LED) and BART. On these long-range tasks, it performs as well or better, taking advantage of the long-range context.

7 Analysis

7.1 Role of SSM

Compared to multi-head attention where routing is determined by $L^2$ attention coefficients per head per layer, the BiGS SSM routing is relatively compact. Each layer has only $2L$ static values in $K$. 

63
Figure 5: Role of gating in downstream accuracy. Compares MNLI accuracy with respect to MLM loss. BERT values from Devlin et al. (2018). Gated SSM shows similar pretraining transfer as BERT, whereas Stack SSM does not.

Figure 3 shows these values in the form of the forward and backward kernels. These kernels correspond partially to local aggregations such as the previous word (layer 1) or a subsequent trigram (layer 6), and partially to long-term future or past information (layer 14, layer 17).

Figure 4 shows how these kernels change during finetuning. In particular, during MNLI finetuning, the model needs to look at more long-distance information to match between sentences. This results in most local kernels remaining the same, but long-distance kernels adjusting. The figure shows three kernels expanding their scope outward.

7.2 Role of Gating

GLUE results show a significant improvement in downstream accuracy with the GATED model; however, we actually find that the worse STACK SSM model has a similar pretraining MLM loss. Figure 5 illustrates the difference of MLM loss and MNLI accuracy for both GATED and STACK SSM, compared to the MLM loss and expected MNLI values presented in BERT (Devlin et al., 2018). The figure shows that for the GATED model downstream accuracy tracks MLM loss, while for STACK it does not. We speculate that multiplicative gating helps the SSM model recover some of the generalization ability of attention, particularly for handling long sequences. For example, table 6 compares accuracy of examples binned by length on the QNLI task. We see that the GATED SSM maintains accuracy as examples get longer and required dependencies move further apart.

Table 4: FLOP comparison between BiGS and BERT with respect to input token length. We calculated FLOP with a batch size of 1 and considered both the forward and backward passes.

<table>
<thead>
<tr>
<th>Length</th>
<th>BiGS</th>
<th>BERT</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>8.1E+10</td>
<td>7.9E+10</td>
</tr>
<tr>
<td>512</td>
<td>3.2E+11</td>
<td>3.4E+11</td>
</tr>
<tr>
<td>1024</td>
<td>6.5E+11</td>
<td>7.2E+11</td>
</tr>
<tr>
<td>4096</td>
<td>2.6E+12</td>
<td>4.1E+12</td>
</tr>
</tbody>
</table>

7.3 Efficiency Analysis

A benefit of BiGS is the ability to scale to much longer sequences without a quadratic increase in Floating Point Operations (FLOPs). In Table 4, we compare theoretical FLOPs of BiGS and BERT for different input token lengths to better understand their relative scalability. At lengths up to 512, the cost of both models is dominated by the feed-forward networks, but when growing beyond 1024, the BiGS approach has a significant FLOP advantage over attention. This increasing efficiency gap trend continues nonlinearly with token lengths of 1024 and 4096 respectively, implying that BiGS is better equipped to handle applications with longer input sequences.

In practice, efficiency is dependent on hardware and implementation. Figure 7 shows an empirical comparison between two versions of BERT - HuggingFace BERT (Wolf et al., 2020) and BERT with FlashAttention (Dao et al., 2022a) - to BiGS equipped with FlashConv (Dao et al., 2022c). FlashAttention is highly optimized FP16 implementation of attention while FlashConv is implemented using FP32 internally for long-range convolution. These models were tested under iden-
tical conditions on a single NVIDIA RTX A6000 GPU for one forward pass of the large model. The results show that BiGS outperforms basic attention, and outperforms highly-optimized FlashAttention when sequence length passes 3k. When comparing to a model without any routing, we can see that the efficiency bottleneck of BiGS lies in the dense layers, while the SSM adds relatively little overhead, even past 8k tokens.

7.4 Task Analysis: Syntactic Properties

While the average GLUE results are similar, BiGS underperforms on some tasks, and outperforms on syntactic tasks such as CoLA (Warstadt et al., 2019) (Appendix Figure 9 and 10). We speculate that these results indicate that SSM-routing may have different inductive biases than attention. We follow Goldberg (2019) in adapting two preliminary experiments with of syntactic tests for masked language modeling:

Linzen et al. (2016) test a model’s ability to distinguish agreement in the presence of spurious intervening “agreement attractors”. For example, the sentence “Yet the ratio of men who survive to the women and children who survive [is] not clear in this story” has three attractors for the masked work [is]. Figure 8 shows that BiGS consistently outperforms BERT as number of attractors grows.

Marvin and Linzen (2018) develop pairs of manually constructed examples targeting various syntactic phenomena and difficulties. Given a pair of examples from this stimuli: “No students have ever lived here” and “Most students have ever lived here”, we feed an adapted version “[MASK] students have ever lived here” into a model and compare the predicted scores for the masked position “No” and “Most” from it. Results are reported in Table 5 and again show that SSM outperforms BERT on several agreement phenomena. While more experiments are needed, it is possible that BiGS leads to an inductive bias to a more stack-like representation, since it cannot rely only on dynamic matching.

7.5 Annotated CoLA

The CoLA corpus collection, as described in (Warstadt et al., 2019), is a vital task within the GLUE benchmark (Wang et al., 2018) for evaluating the acceptability of language models. This corpus has been specifically annotated with 13 different syntactic phenomena in order to more accurately quantify the linguistic knowledge of pre-trained language models (LLMs) (Warstadt and Linzen, 2018). Numbers of LSTM models are taken from (Goldberg, 2019).
We break down the matthews correlation coefficient (MCC) of the BiGS and BERT model w.r.t sentence length in Figure 10. BiGS outperforms BERT on both short and long text.

8 Conclusion

We propose BiGS as a model for pretraining without attention. BiGS makes use of SSM-based routing and multiplicative gating. Results show that SSMs alone perform poorly in a stacked architecture, but gating helps them to generalize. As far as we are aware, this architecture is the first to replicate BERT results without attention.

This work opens up many interesting questions. We experimented with adapting to longer text, but SSM-based models could be pretrained fully on much longer sequences. Combining SSMs with reductions in feed-forward costs could give further optimizations. Finally, we took the steps in exploring the syntactic properties of SSMs, but need further probing of how their internal representations lead to these properties.

9 Limitations

While SSMs are a promising technology for pretraining, they are not yet a full replacement for attention. One limitation is that this work only considers an encoder model and not an encoder-decoder setup. This makes it challenging to compare to BART and LED in some longer-range evaluations. For example, in our preliminary studies in applying BiGS to long-range question answering WikiQA (Yang et al., 2015), TriviaQA (Joshi et al., 2017), we did not see direct benefits of SSM in an
encoder setting. Others have experimented with decoder SSM models, but it is not clear how cross-attention should work with these models. This work also considers SSMs for bidirectional pre-training, and not autoregressive modeling. Therefore, some benefits of SSMs are less apparent, such as the utilization of RNN generation.

10 Ethical Considerations

Our models are trained using a corpus consisting of existing collections of text from Wikipedia and books. Recent research has uncovered potential societal biases that are embedded within many established corpora. While it is beyond the scope of this paper to delve into these biases in depth, we acknowledge the potential risk that our pre-trained models may inherit these biases. In light of this, we are interested in exploring whether previous research on language bias detection can be applied to BiGS, as part of future work. Additionally, in this paper, we have focused solely on the English corpus, and it would be interesting to investigate how BiGS can contribute to multi-lingual language modeling in the future.

11 Acknowledgement

We gratefully acknowledge the support of Google’s TPU Research Cloud (TRC) program in providing Cloud TPU resources for this research. We thank Sanchit Gandhi and Matthew Carrigan from Huggingface for their valuable engineering and research suggestions. AMR is supported by a Sloan Fellowship and NSF CAREER #2037519.

References


A Appendix

A.1 Pretraining Details

All models are pretrained using a single cloud TPU-v3. Table 6 shows the hyperparameter configurations that we examine in our pretraining.

BiGS with 512 token length model is trained with 10,000 steps (53,248 tokens per batch) using learning rate 4e-5.

To compare with LED (Beltagy et al., 2020) and BART (Lewis et al., 2019) in the scroll experiment, we first train a BiGS with 12 layers (around 130M parameters in total) and 128 maximal sentence length using 500,000 steps and later extend it to 4096 token length with 10k more training steps using learning rate 3e-5.

A.2 Finetuning Details

All models are finetuned using either a single cloud TPU-v3 or TPU-v2.

A.2.1 GLUE

Table 7 shows hyperparameter configurations used to finetune GLUE tasks.

A.2.2 Other tasks

Table 8 shows hyperparameter configurations used to finetune SQuAD and QALT/CNLI tasks.

A.3 SCROLLS Validation Result

We conducted experiments to evaluate the performance of BiGS and LED calibrating their performance under the setting of long sequences.

From results shown in the Table 9, we can see that the performance of LED actually degrades as the sequence length increases, whereas BiGS demonstrates improved accuracy with longer sequence lengths.

While it is true that RoBERTa performs better at a shorter length, it is also trained for much longer on more data. With a head-to-head comparison, the length benefits of BiGS can be shown very clearly.

---

Table 6: Hyperparameters used for pretraining BiGS and BERT models

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>BiGS</th>
<th>BERT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Layers</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>Hidden size</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>Intermediate size</td>
<td>3072</td>
<td>4096</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Learning Rate Decay</td>
<td>{Cosine, Linear}</td>
<td>{Linear}</td>
</tr>
<tr>
<td>Weight Decay</td>
<td>{0.05, 0.01}</td>
<td>{0.01}</td>
</tr>
<tr>
<td>Learning Rate</td>
<td>{2e-4, 4e-4, 6e-4, 8e-4}</td>
<td>{2e-4, 4e-4}</td>
</tr>
<tr>
<td>Optimizer</td>
<td>AdamW</td>
<td>AdamW</td>
</tr>
<tr>
<td>Adam ,</td>
<td>1e-6</td>
<td>1e-6</td>
</tr>
<tr>
<td>Adam ,</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>Adam ,</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>Gradient Clipping</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Batch Size</td>
<td>{760, 1048, 1136}</td>
<td>{840}</td>
</tr>
<tr>
<td>Warmup Proportion</td>
<td>(1%)</td>
<td>(2%)</td>
</tr>
</tbody>
</table>

Table 7: Hyperparameters used for finetuning our model on GLUE benchmark tasks.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>SQuAD</th>
<th>QALT/CNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning Rate</td>
<td>{4e-5, 6e-5}</td>
<td>{1e-5, 3e-5, 5e-5}</td>
</tr>
<tr>
<td>Weight Decay</td>
<td>{0.01, 0.1}</td>
<td>{0.01}</td>
</tr>
<tr>
<td>Batch Size</td>
<td>{32}</td>
<td>{8, 16, 24}</td>
</tr>
<tr>
<td>Max Epochs</td>
<td>{2}</td>
<td>{5, 8, 10}</td>
</tr>
<tr>
<td>Warmup Proportion</td>
<td>{0.1}</td>
<td>{0.1}</td>
</tr>
</tbody>
</table>

Table 8: Hyperparameters used for finetuning our model in SQuAD and QALT/CNLI tasks.

<table>
<thead>
<tr>
<th>Length</th>
<th>QALT</th>
<th>CNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoBERTa (355M)</td>
<td>512</td>
<td>28.3</td>
</tr>
<tr>
<td>LED (162M)</td>
<td>1024</td>
<td>27.9</td>
</tr>
<tr>
<td>LED (162M)</td>
<td>4096</td>
<td>24.0</td>
</tr>
<tr>
<td>LED (162M)</td>
<td>16384</td>
<td>20.3</td>
</tr>
<tr>
<td>BiGS (350M)</td>
<td>128</td>
<td>28.4</td>
</tr>
<tr>
<td>BiGS (350M)</td>
<td>512</td>
<td>29.5</td>
</tr>
<tr>
<td>BiGS (350M)</td>
<td>1024</td>
<td>30.6</td>
</tr>
<tr>
<td>BiGS (350M)</td>
<td>4096</td>
<td>31.4</td>
</tr>
</tbody>
</table>

Table 9: SCROLLS validation set results. Inputs are truncated at length. LED experiments are from SCROLLS official repository. We report an average over three runs.