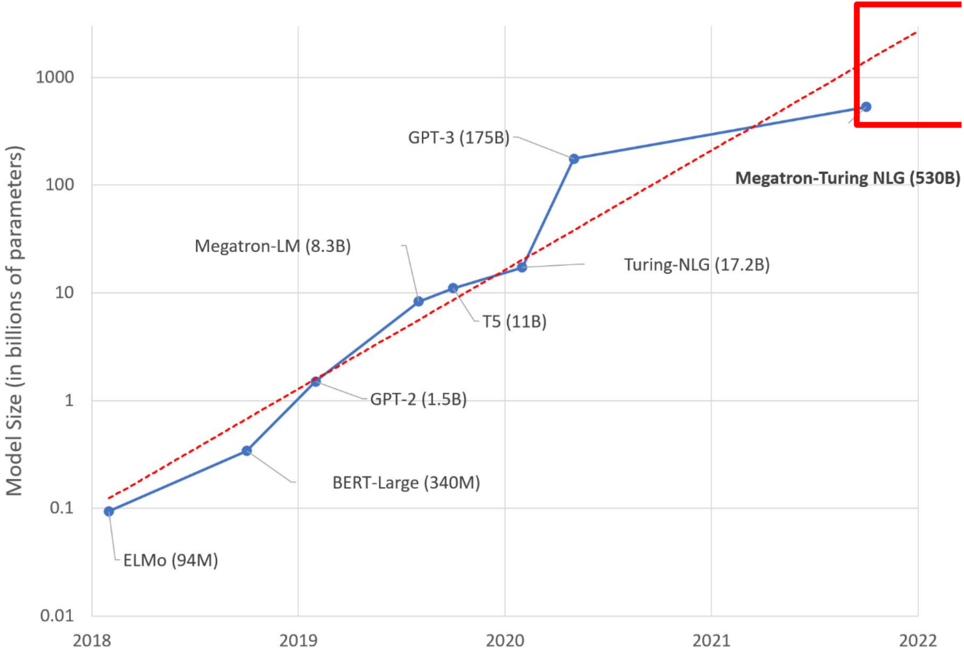
Pretraining Without Attention

Junxiong Wang   Jing Nathan Yan   Albert Gu   Sasha Rush

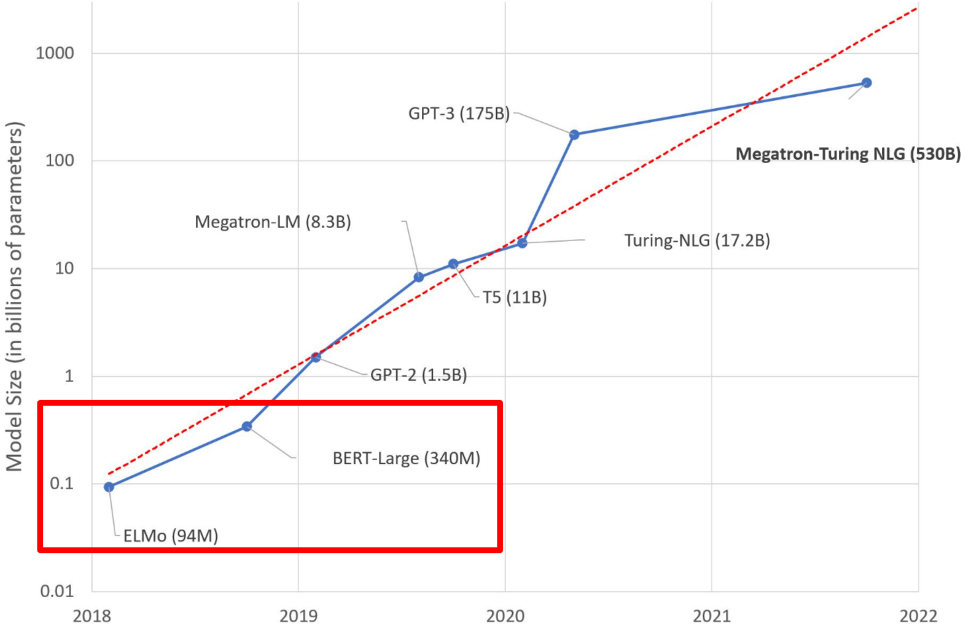
Cornell University, CMU

Caveats

- LLMs are remarkable, we should use them for most things
- This talk is not about LLMs



Model Size (in billions of parameters)
1000
100
10
1
0.1
0.01
2018
2019
2020
2021
2022
ELMo (94M)
BERT-Large (340M)
GPT-2 (1.5B)
Megatron-LM (8.3B)
T5 (11B)
Turing-NLG (17.2B)
GPT-3 (175B)
Megatron-Turing NLG (530B)

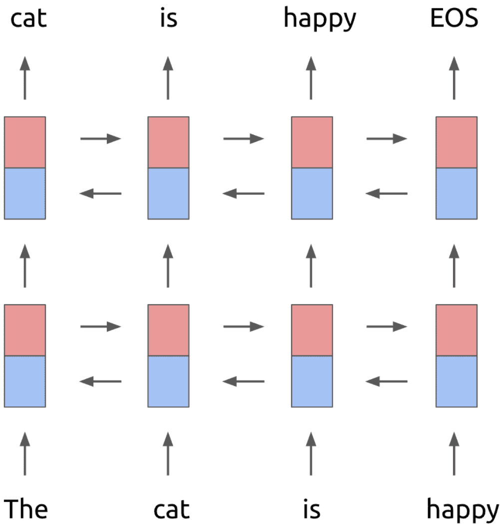

Model Size (in billions of parameters)
1000
100
10
1
0.1
0.01
2018
2019
2020
2021
2022
GPT-3 (175B)
Megatron-Turing NLG (530B)
Megatron-LM (8.3B)
Turing-NLG (17.2B)
T5 (11B)
GPT-2 (1.5B)
BERT-Large (340M)
ELMo (94M)

Context

- BERT used to require non-trivial compute
- Belief: Open architecture questions in NLP
- Today's Talk: How important is *attention*?

ELMo

Bidirectional RNN



ELMo For Pretraining

Model	GLUE
ELMo	67.7
ELMo+Attn	71.0

[Peters et al., 2018, Devlin et al., 2018]

ELMo For Pretraining

Model	GLUE
ELMo	67.7
ELMo+Attn	71.0
BERT-Base	79 - 83

[Peters et al., 2018, Devlin et al., 2018]

Architecture?

- Several confounding differences, e.g. frozen model.
- Followup: *To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks* [Peters et al., 2019]

Architecture?

- Several confounding differences, e.g. frozen model.
- Followup: *To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks* [Peters et al., 2019]
- Conclusion: Transformers significantly beat BiLSTMs

Other Models

Maybe there are other models

- Convolutions?
- Mixers?

Pretraining with CNNs

Are Pre-trained Convolutions Better than Pre-trained Transformers? [Tay et al., 2020]

Pretraining with CNNs

Are Pre-trained Convolutions Better than Pre-trained Transformers? [Tay et al., 2020]

Answer: No.

Model	SST-2
ELMo	91.8
Best CNN	92.2
BERT-Base	93.5

Pretraining with FNet

FNet: Mixing Tokens with Fourier Transforms
[Lee-Thorp et al., 2021]

Replaces attention with 2D FFT mixing-layer.

Pretraining with FNet

FNet: Mixing Tokens with Fourier Transforms
[Lee-Thorp et al., 2021]

Replaces attention with 2D FFT mixing-layer.

Model	GLUE (dev)
Best FNet	76.3
BERT-Base	83.3

Transformers are Great...

- Highly optimized training
- Long-range ability
- Expensive $O(n^2)$, but we have the money...

Transformers are Great...

- Highly optimized training
- Long-range ability
- Expensive $O(n^2)$, but we have the money...

(But aren't you curious...)

Outline

Context

State Space Models

Model Architectures

Experiments

State Space Models (SSM)

- Think hybrid RNN / CNN
- SOTA on speech generation and long-range tasks
- Tutorial at *The Annotated S4*

[Gu et al., 2020, Gu et al., 2021b, Gu et al., 2021a]

State Space Model - Continuous Time

Let $u(t) \in \mathbb{R}$ be a continuous input and $y(t) \in \mathbb{R}$ be output.

State Space Model - Continuous Time

Let $u(t) \in \mathbb{R}$ be a continuous input and $y(t) \in \mathbb{R}$ be output.

SSM is a differential equation.

$$\begin{aligned} \boldsymbol{x}'(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}u(t) \\ y(t) &= \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}u(t). \end{aligned}$$

State Space Model - Continuous Time

Let $u(t) \in \mathbb{R}$ be a continuous input and $y(t) \in \mathbb{R}$ be output.

SSM is a differential equation.

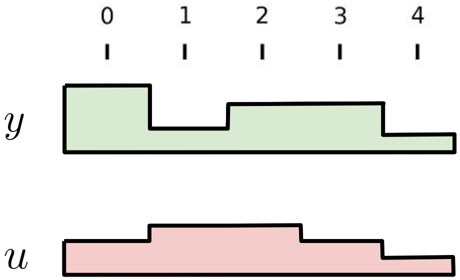
$$\begin{aligned} \boldsymbol{x}'(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}u(t) \\ y(t) &= \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}u(t). \end{aligned}$$

Where $\boldsymbol{x}(t) \in \mathbf{R}^N$ is a hidden state and model parameters,

$$\boldsymbol{A} \in \mathbb{R}^{N\times N}, \boldsymbol{B} \in \mathbb{R}^{N\times 1}, \boldsymbol{C} \in \mathbb{R}^{1\times N}, \boldsymbol{D} \in \mathbb{R}^{1\times 1}$$

Discrete Time Sequence

Goal: Map scalar sequence $u_1, \ldots, u_L$ to $y_1, \ldots, y_L$,



Discrete Time SSM

SSM on discretize time data,

$$
\begin{aligned}
\boldsymbol{x}_k &= \overline{\boldsymbol{A}}\boldsymbol{x}_{k-1} + \overline{\boldsymbol{B}}u_k \\
y_k &= \overline{\boldsymbol{C}}\boldsymbol{x}_k \quad\; + \overline{\boldsymbol{D}}u_k.
\end{aligned}
$$

Using discretization with (learned) sampling rate parameter Δ,

$$
\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \overline{\boldsymbol{C}} = \text{discretize}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \Delta)
$$

Recurrent Form

Output sequence $y_1, \ldots, y_L$ can be computed as a linear RNN,

$$\begin{aligned} \boldsymbol{x}_k &= \overline{\boldsymbol{A}}\boldsymbol{x}_{k-1} + \overline{\boldsymbol{B}}u_k \\ y_k &= \overline{\boldsymbol{C}}\boldsymbol{x}_k \quad\ \ + \overline{\boldsymbol{D}}u_k. \end{aligned}$$

Note $\boldsymbol{x}_k \in \mathbb{R}^N$ is the bigger hidden state for $u_k \in \mathbb{R}$, and $\boldsymbol{x}_0 = \mathbf{0}$.

Convolutional Form

Alternative: 1D convolution with kernel $\overline{\boldsymbol{K}}$ (width L),

$$\overline{K} = (\overline{\boldsymbol{C}}\overline{\boldsymbol{B}}, \overline{\boldsymbol{C}}\overline{\boldsymbol{A}}\overline{\boldsymbol{B}}, \dots, \overline{\boldsymbol{C}}\overline{\boldsymbol{A}}^{L-1}\overline{\boldsymbol{B}})$$
$$y = \text{conv1d}(\overline{K}_L \dots \overline{K}_1, u_1 \dots u_L)$$

Intuition:

Convolutional Form

Alternative: 1D convolution with kernel $\overline{\boldsymbol{K}}$ (width L),

$$\overline{K} = (\overline{\boldsymbol{C}\boldsymbol{B}}, \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}}, \dots, \overline{\boldsymbol{C}\boldsymbol{A}}^{L-1}\overline{\boldsymbol{B}})$$

$$y = \text{conv1d}(\overline{K}_L \dots \overline{K}_1, u_1 \dots u_L)$$

Intuition:

$$y_1 = \overline{\boldsymbol{C}\boldsymbol{B}}u_1$$

Convolutional Form

Alternative: 1D convolution with kernel $\overline{\boldsymbol{K}}$ (width L),

$$\overline{K} = (\overline{\boldsymbol{C}\boldsymbol{B}}, \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}}, \dots, \overline{\boldsymbol{C}\boldsymbol{A}}^{L-1}\overline{\boldsymbol{B}})$$
$$y = \text{conv1d}(\overline{K}_L \dots \overline{K}_1, u_1 \dots u_L)$$

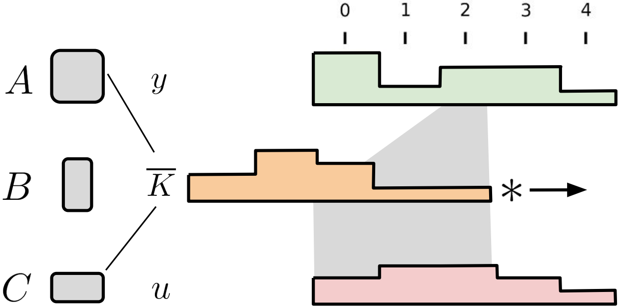
Intuition:

$$y_1 = \overline{\boldsymbol{C}\boldsymbol{B}}u_1$$

$$y_2 = \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}}u_1 + \overline{\boldsymbol{C}\boldsymbol{B}}u_2 = \overline{\boldsymbol{C}}(\overline{\boldsymbol{A}\boldsymbol{B}}u_1 + \overline{\boldsymbol{B}}u_2) = \overline{\boldsymbol{C}}(\boldsymbol{x}_1 + \overline{\boldsymbol{B}}u_2)$$

Convolutional Form

Step 1: Discretize (Training Only). Step 2: Apply 1D Conv



Implementation - Computing Kernel

$$\overline{\boldsymbol{K}} = (\overline{\boldsymbol{C}\boldsymbol{B}}, \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}}, \dots, \overline{\boldsymbol{C}\boldsymbol{A}}^{L-1}\overline{\boldsymbol{B}})$$

- Simple approximations work well (See S4D, DSS)

[Gu et al., 2021a, Gupta, 2022, Gu et al., 2022]

Implementation - Fourier Transform

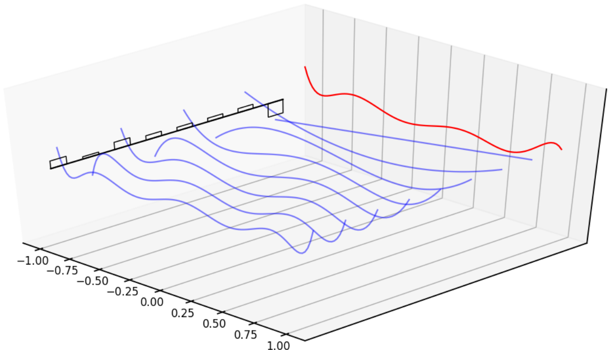
$$y = \overline{\boldsymbol{K}} * u$$

- At long L, convolution computed with FFT.
- More efficient than self-attention or standard RNN.

Important Training Initialization

- Parameter A is initialized with HiPPO Matrix [Gu et al., 2020]

- Kernel formed by Legendre coefficients



Summary: SSM

- Mapping from sequence-to-sequence
- Acts like an RNN, Computed like a CNN
- Fast to train and utilize

Outline

Context

State Space Models

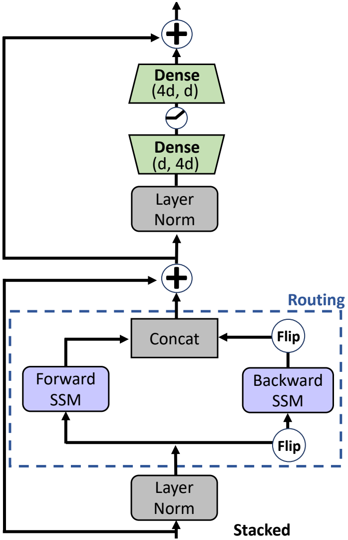
Model Architectures

Experiments

Objective: Replicate BERT with SSM

- Everything else identical (loss, number of parameters, data)

Naive Idea **Self-attention ⇒ SSM**



Can this work?

- SSM is significantly less expressive than self-attention.
- Static routing through the model like a CNN.
- Can it learn to do matching across sentences?

Can this work?

- SSM is significantly less expressive than self-attention.
- Static routing through the model like a CNN.
- Can it learn to do matching across sentences?

Test: Matching Across Gaps

Task: QNLI [Wang et al., 2018]

What percentage of farmland grows wheat?

$\sim\sim\sim$

More than 50% of this area is sown for wheat and 33% for barley.

Test: Matching Across Gaps

Task: QNLI [Wang et al., 2018]

What percentage of farmland grows wheat?

$\sim\sim\sim$

More than 50% of this area is sown for wheat and 33% for barley.

Arch	H P	H $\sim$ P
stack / ssm	77.4	69.7

Proposed Fix: **Multiplicative Gating**

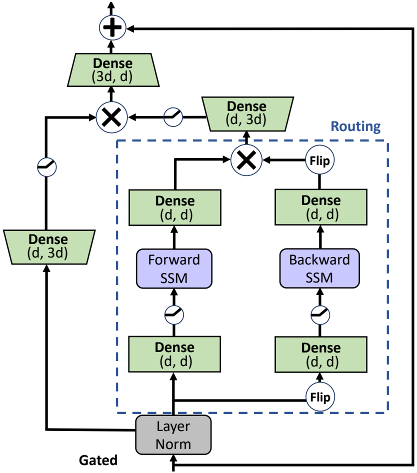
Add dynamism to stacked model with multiplicative gating.

$$\sigma(\mathbf{Wu}) \otimes (\mathbf{Vu})$$

Positive results with CNN, Transformer, and SSM models.

[Dauphin et al., 2017, Shazeer, 2020, Narang et al., 2021]

Proposed Architecture: BiGS



Gating Adaptation

What percentage of farmland grows wheat?

$\sim\sim\sim$

More than 50% of this area is sown for wheat and 33% for barley.

Arch	H P	H $\sim$ P
stack / ssm	77.4	69.7
gated / ssm	77.4	77.7

Gating Adaptation

What percentage of farmland grows wheat?

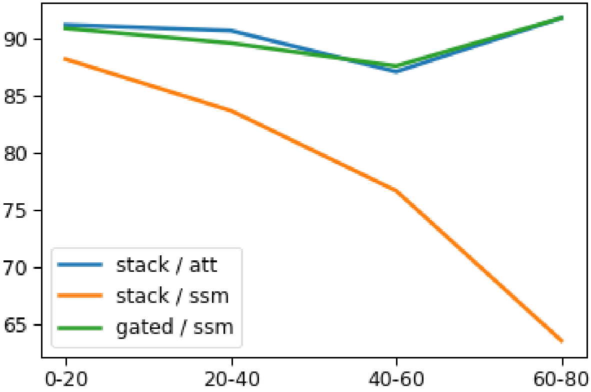
$\sim\sim\sim$

More than 50% of this area is sown for wheat and 33% for barley.

Arch	H P	H $\sim$ P
stack / ssm	77.4	69.7
gated / ssm	77.4	77.7

Full Experiment: QNLI

Preview: Experimental results, pretraining for QNLI.



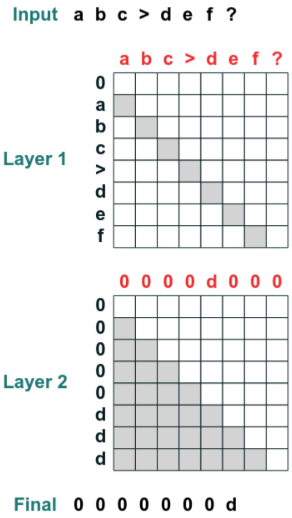
Related Result: Induction Heads (H3)

Synthetic induction head experiment from [Dao et al., 2022]

$$a\ b\ c\ d\ e \Rightarrow f\ g\ h\ i \ldots x\ y\ z \Rightarrow \quad f$$

Arch	Induction
ssm	35.6
gating + ssm	100
attention	100

Induction Heads




```
shift_filter = key(indices) == query(indices - 1)
# Filter 1
out = shift_filter.value(tokens)
# Gate
out = where(out == ">", tokens, 0)
# Filter 2
move_right_filter = key(indices) < query(indices)
out = move_right_filter.value(out)
# Gate
out = where((out != 0) & (tokens == "?"), out, 0)
out("abc>def?")
```

Outline

Context

State Space Models

Model Architectures

Experiments

Experiment 1: **BERT**

- Models trained using “24 Hour” BERT [Izsak et al., 2021]
  - All BERT-Large Size
  - Training length (Short 11B, Medium 22B, Full >100B)
  - 128 Length Sequences

- Codebase in JAX (from Annotated S4 [Rush, 2022]) using S4D

- Training data and masking is identical

Short Training $\sim$11B Tokens

Model	GLUE (Dev)
ELMo	68.7
BERT	83.3
Stacked-SSM	77.2
BiGS	83.3

Is it just Gating?

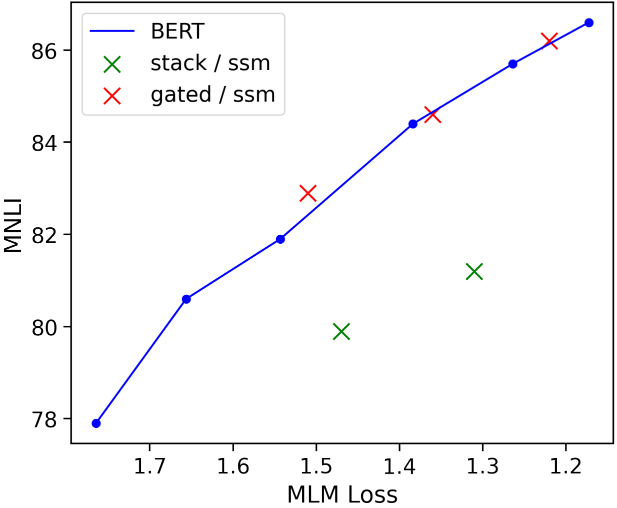
Model	GLUE
BERT	83.3
Gated-BERT	81.8

BERT Large > 100B Tokens

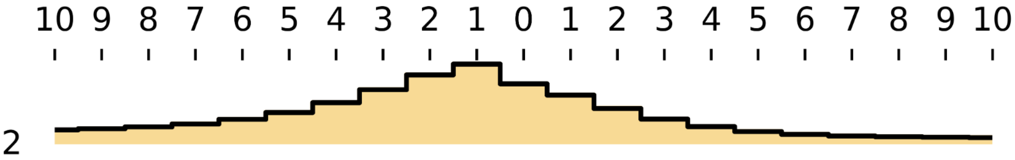
Model	GLUE (Test)
BERT-Large*	83.0
BiGS	83.0

*Best reported BERT-Large Results.

Analysis: Masked PPL Transfer

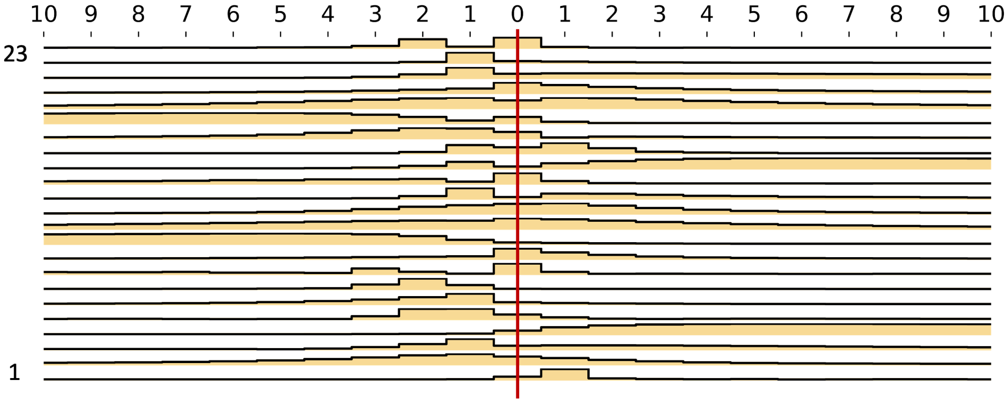


Analysis: Kernel Visualization

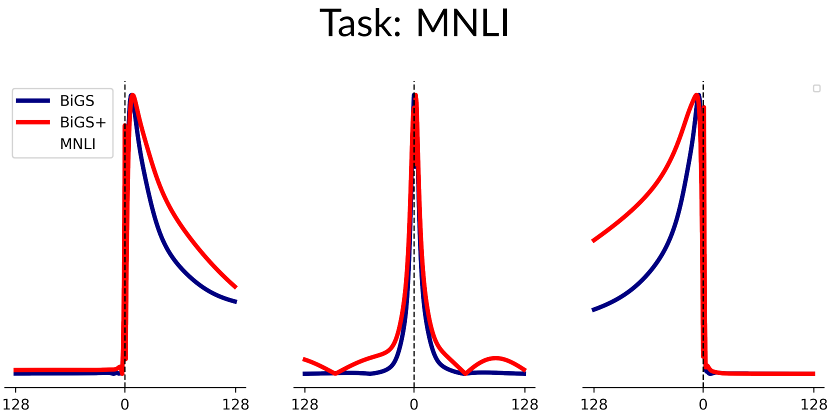


- Each BiGS layer only has 2 kernels (forward / backward).
- Shows all routing in layer 2! (vs $O(HT^2)$ attention coef.)

Analysis: All Kernels



Analysis: Change in Kernels during Finetuning



Analysis: Syntax

- Observation: SSM model seems to do better on syntax-centric tasks

- Hypothesis: Locality of features encourages a stack-like inductive bias.

Observation 1: COLA

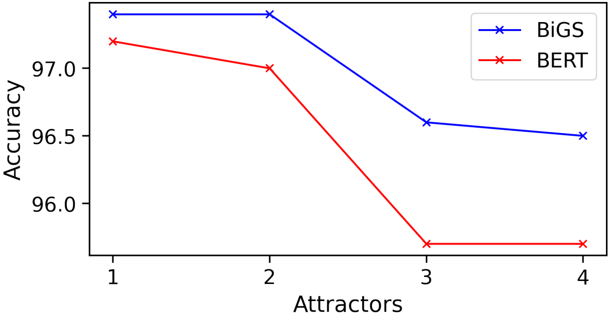
Model	COLA
BERT	60.5
BiGS	64.7

Statistically significant across runs.

Observation 2: Agreement Attractors

Task from [Linzen et al., 2016, Goldberg, 2019].

*Yet the **ratio** of men who survive to the women and children who survive [is] not clear in this story*



Observation 3: Diagnostics

From [Marvin and Linzen, 2018, Goldberg, 2019]:

	BiGS	BERT	LSTM
SUBJECT-VERB:			
Simple	100.0	100.0	94.0
Sentential complement	85.1	85.6	99.0
Short VP coordination	91.0	86.5	90.0
Long VP coordination	97.5	97.5	61.0
Across prep phrase	88.6	84.8	57.0
Across subj relative clause	88.4	84.9	56.0
Across obj relative clause	89.9	85.1	50.0
Across obj relative (-that)	86.9	81.1	52.0
In obj relative clause	97.2	99.1	84.0
In obj relative (-that)	88.7	81.6	71.0
REFL ANAPHORA:			
Simple	97.1	98.9	83.0
In a sentential complement	79.9	86.2	86.0
Across a relative clause	79.1	75.9	55.0

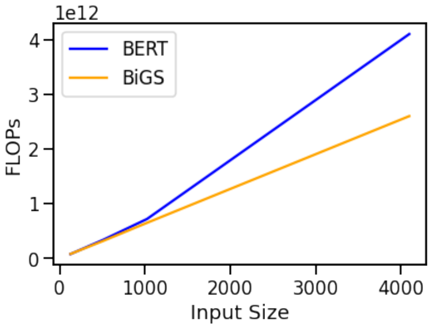
Experiment 2: **Longformer**

- Can we lengthen SSM $L \to L'$ without approximation?
- Continued training based on Longformer protocol.
- Two experimental scales

SCROLLS

	Length	QALT	CNLI
LED(162M)	1024	26.6/27.2	73.4
	4096	26.6/27.3	71.5
	16384	25.8/25.4	71.5
BART (140M)	256	26.0/25.8	69.8
	512	26.8/27.4	71.6
BiGS (130M)	128	32.3/30.0	68.7
	4096	32.8/31.7	71.4

FLOPs



Related Results: H3 - SSM For Language Modeling

- Alternative gating method for language modeling
- Use 2 attention layers + SSM and reach Transformer PPL.
- Efficient implementation targeting on GPUs.

[Dao et al., 2022]

Next Steps

- Attention may not be required? Simpler routing + gating.
- More analysis on feed-forward contribution.
- Transfer from pretraining unclear.

References I

Dao, T., Fu, D. Y., Saab, K. K., Thomas, A. W., Rudra, A., and Ré, C. (2022).
Hungry hungry hippos: Towards language modeling with state space models.
arXiv preprint arXiv:2212.14052.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. (2017).
Language modeling with gated convolutional networks.
In *International conference on machine learning*, pages 933–941. PMLR.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018).
Bert: Pre-training of deep bidirectional transformers for language understanding.
arXiv preprint arXiv:1810.04805.

Goldberg, Y. (2019).
Assessing bert's syntactic abilities.
arXiv preprint arXiv:1901.05287.

References II

Gu, A., Dao, T., Ermon, S., Rudra, A., and Ré, C. (2020).
Hippo: Recurrent memory with optimal polynomial projections.
Advances in Neural Information Processing Systems, 33:1474–1487.

Gu, A., Goel, K., and Ré, C. (2021a).
Efficiently modeling long sequences with structured state spaces.
arXiv preprint arXiv:2111.00396.

Gu, A., Gupta, A., Goel, K., and Ré, C. (2022).
On the parameterization and initialization of diagonal state space models.
arXiv preprint arXiv:2206.11893.

References III

Gu, A., Johnson, I., Goel, K., Saab, K., Dao, T., Rudra, A., and Ré, C. (2021b).
Combining recurrent, convolutional, and continuous-time models with linear state space layers.
Advances in Neural Information Processing Systems, 34.

Gupta, A. (2022).
Diagonal state spaces are as effective as structured state spaces.
arXiv preprint arXiv:2203.14343.

Izsak, P., Berchansky, M., and Levy, O. (2021).
How to train bert with an academic budget.
arXiv preprint arXiv:2104.07705.

References IV

- Lee-Thorp, J., Ainslie, J., Eckstein, I., and Ontanon, S. (2021).
  Fnet: Mixing tokens with fourier transforms.
  arXiv preprint arXiv:2105.03824.
- Linzen, T., Dupoux, E., and Goldberg, Y. (2016).
  Assessing the ability of lstms to learn syntax-sensitive dependencies.
  Transactions of the Association for Computational Linguistics, 4:521–535.
- Marvin, R. and Linzen, T. (2018).
  Targeted syntactic evaluation of language models.
  arXiv preprint arXiv:1808.09031.

References V

Narang, S., Chung, H. W., Tay, Y., Fedus, W., Fevry, T., Matena, M., Malkan, K., Fiedel, N., Shazeer, N., Lan, Z., et al. (2021).

Do transformer modifications transfer across implementations and applications?

arXiv preprint arXiv:2102.11972.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018).

Deep contextualized word representations.

In Walker, M. A., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

References VI

Peters, M. E., Ruder, S., and Smith, N. A. (2019).
To tune or not to tune? adapting pretrained representations to diverse tasks.
arXiv preprint arXiv:1903.05987.

Rush, A. (2022).
The annotated s4.
International Conference on Learning Representations.

Shazeer, N. (2020).
Glu variants improve transformer.
arXiv preprint arXiv:2002.05202.

Tay, Y., Dehghani, M., Bahri, D., and Metzler, D. (2020).
Efficient transformers: A survey.
arXiv preprint arXiv:2009.06732.

References VII

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. (2018).

Glue: A multi-task benchmark and analysis platform for natural language understanding.

arXiv preprint arXiv:1804.07461.