ADDRESSING SOME LIMITATIONS OF TRANSFORMERS WITH FEEDBACK MEMORY

Angela Fan†, Thibaut Lavril, Edouard Grave, Armand Joulin, Sainbayar Sukhbaatar

Facebook AI Research
†LORIA

https://arxiv.org/abs/2002.09402v3

Alex Fedorov

April 9, 2021

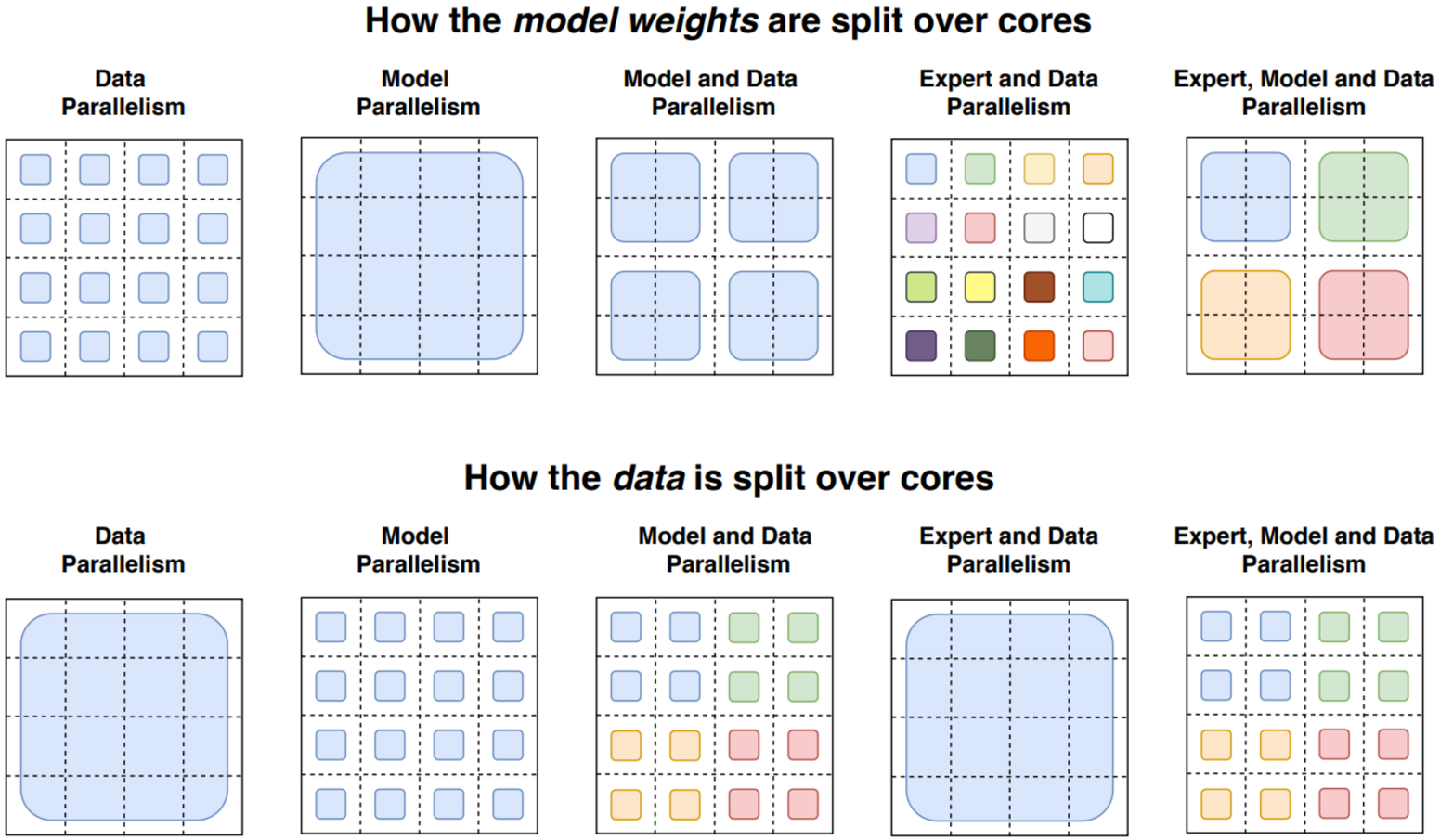
Vision

- Transformers are **efficient** on modern hardware
  - A sequence processed in parallel in an order-invariant way
- Transformers **do not take full advantage** of the input's **sequential property**
  - Limited Attention Span
  - Fail to generalize in copying strings and simple logical inference
    - Universal Transformers (https://arxiv.org/abs/1807.03819)
- Transformers **can not represent hierarchical structure**
  - https://arxiv.org/abs/1803.03585 and https://arxiv.org/abs/1906.06755

Related Work

Parallelization

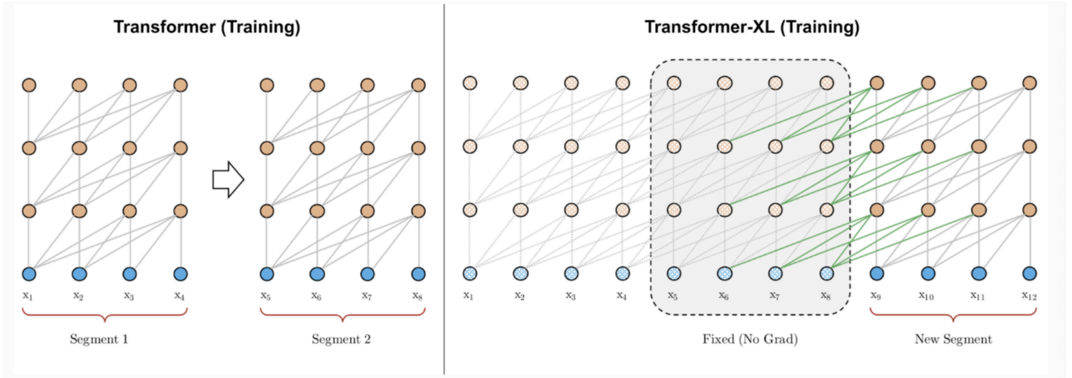
- Switch Transformers
  - Trillion+ parameters
  - Mixture of experts

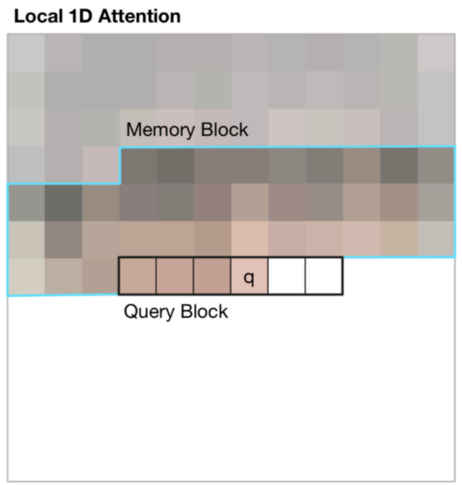


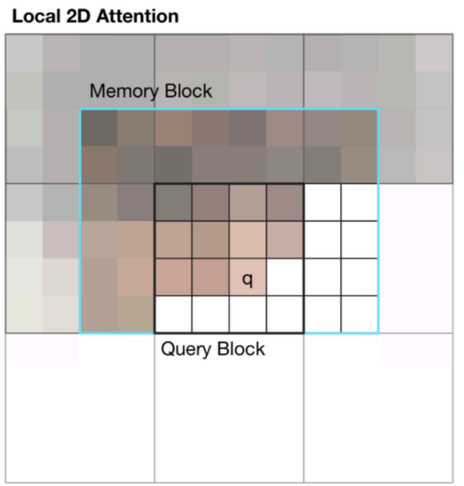
- https://arxiv.org/abs/2101.03961

Longer Attention Span

- Transformer-XL
  - Hidden State Reuse + Relative positional Encoding
  - https://arxiv.org/abs/1901.02860
- Adaptive Attention Span
  - Lower layers do not ned very long attention spans, while opposite for higher
  - https://arxiv.org/abs/1905.07799
- Localized Attention Span (Image Transformer)
  - https://arxiv.org/pdf/1802.05751.pdf







Improving temporal relations in Transformers

- Universal Transformers
  - Recurrent network in depth with Adaptive Computation Time (https://arxiv.org/abs/1603.08983)
  - DEQ https://arxiv.org/abs/1909.01377
  - ALBERT https://arxiv.org/pdf/1909.11942.pdf
- Adding second recurrent encoder
  - https://arxiv.org/abs/1904.03092 and https://arxiv.org/abs/1804.09849
- R-Transformer
  - Adding Recurrent sublayer to remove the need of positional encoding https://arxiv.org/abs/1907.05572
- Self-Attention on top of LSTM cell past output
  - https://arxiv.org/abs/1911.11423

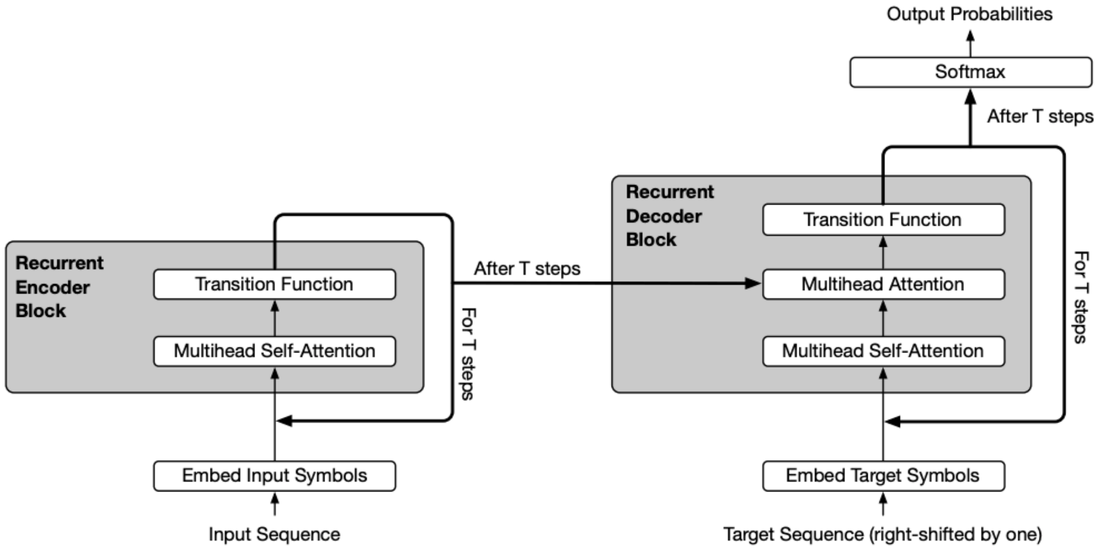


Figure 2: The recurrent blocks of the Universal Transformer encoder and decoder. This diagram omits position and time-step encodings as well as dropout, residual connections and layer normalization. A complete version can be found in Appendix A. The Universal Transformer with dynamic halting determines the number of steps T for each position individually using ACT (Graves, 2016).

Finetuning Pretrained Transformers into RNNs

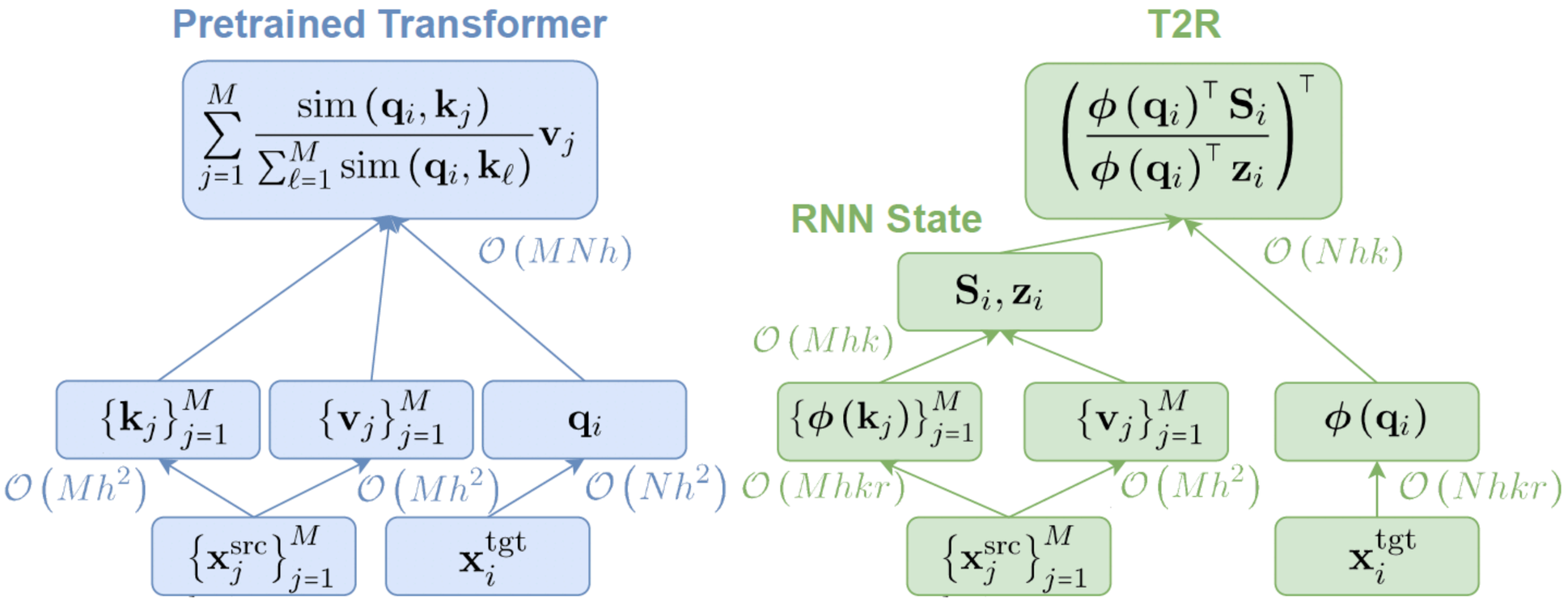


Figure 1: Attention computation steps and their time complexity in pretrained transformer and T2R models during inference generation. Features $\phi(\mathbf{q}_i)$ and $\phi(\mathbf{k}_j)$ are directly computed from input vectors, and $\mathbf{q}_i$ and $\mathbf{k}_j$ are never constructed. M: source length; N: target length; h: model dimensions; k: feature size; r: # heads.

https://arxiv.org/abs/2103.13076v1

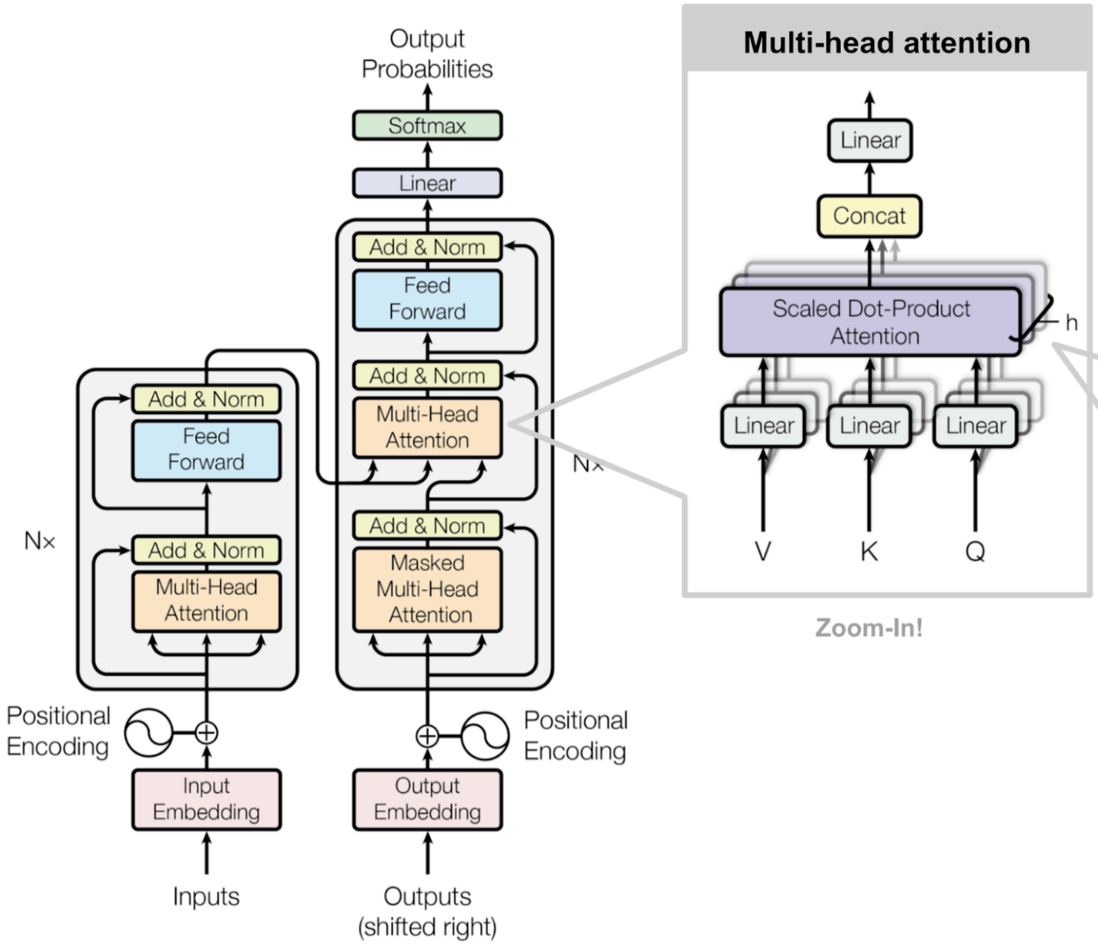
This work

Limitations of Transformers

- Limited Access to Higher Level Representations
- Maintaining a belief state
  - Memory of past inputs
  - World State that not observable in input

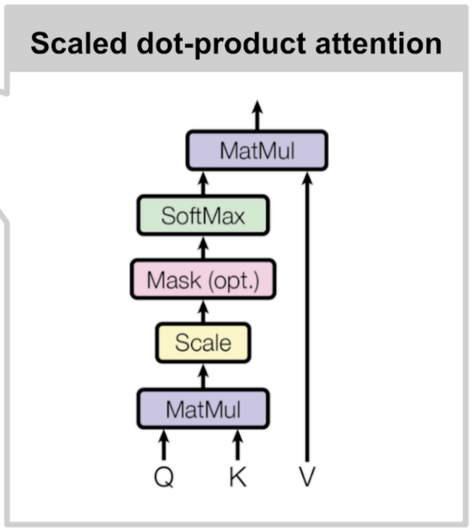
Transformer (T)

$$\text{MultiHeadAttention}\left(\mathbf{X}_q, \mathbf{X}_k, \mathbf{X}_v\right) = [\text{ head }_1; \ldots; \text{ head }_h]\mathbf{W}^o$$
$$\text{where head }_i = \text{Attention}\left(\mathbf{X}_q\mathbf{W}_i^q, \mathbf{X}_k\mathbf{W}_i^k, \mathbf{X}_v\mathbf{W}_i^v\right)$$



Zoom-In!

$$\text{Attention}\left(\mathbf{Q}, \mathbf{K}, \mathbf{V}\right) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}$$



Zoom-In!

$$a_{ij} = \text{softmax}\left(\frac{\mathbf{q}_i\mathbf{k}_j^\top}{\sqrt{d_k}}\right) = \frac{\exp\left(\mathbf{q}_i\mathbf{k}_j^\top\right)}{\sqrt{d_k}\sum_{r\in S_i}\exp\left(\mathbf{q}_i\mathbf{k}_r^\top\right)}$$

https://lilianweng.github.io/lil-log/2020/04/07/the-transformer-family.html

Transformer (T)

$$\mathbf{z}_t^l = \mathrm{Attn}\left(\mathbf{x}_t^l, \{\mathbf{x}_{t-\tau}^l, \ldots, \mathbf{x}_{t-1}^l\}\right)$$

$$\mathbf{X}^{l+1} = \mathrm{FF}\left(\mathrm{Attn}\left(\mathbf{X}^l\right)\right)$$

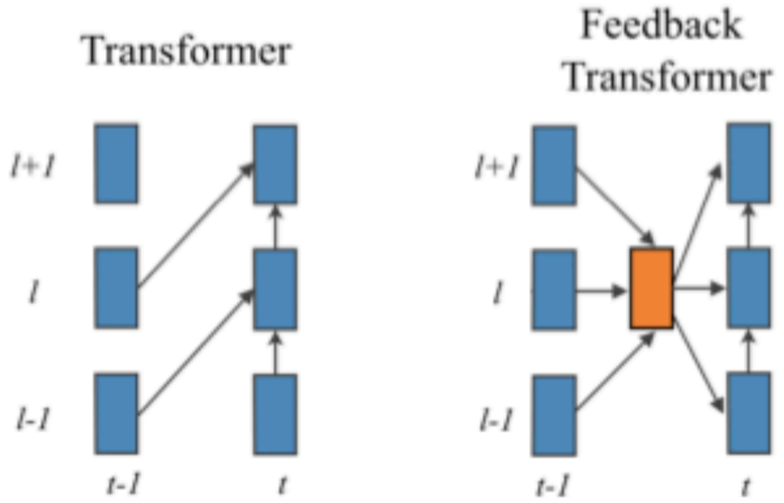


Figure 2: **Difference between Feedback and Transformer.** t indicates the timestep and l indicates the layer.

Feedback-T

$$\mathbf{z}_t^l = \mathrm{Attn}\left(\mathbf{x}_t^l, \{\mathbf{m}_{t-\tau}, \ldots, \mathbf{m}_{t-1}\}\right)$$

$$\mathbf{m}_t = \sum_{l=0}^{L} \mathrm{Softmax}\left(w^l\right)\mathbf{x}_t^l$$

$$\mathbf{k}_t^l = \mathbf{k}_t = W_k\mathbf{m}_t \quad \mathbf{v}_t^l = \mathbf{v}_t = W_v\mathbf{m}_t$$

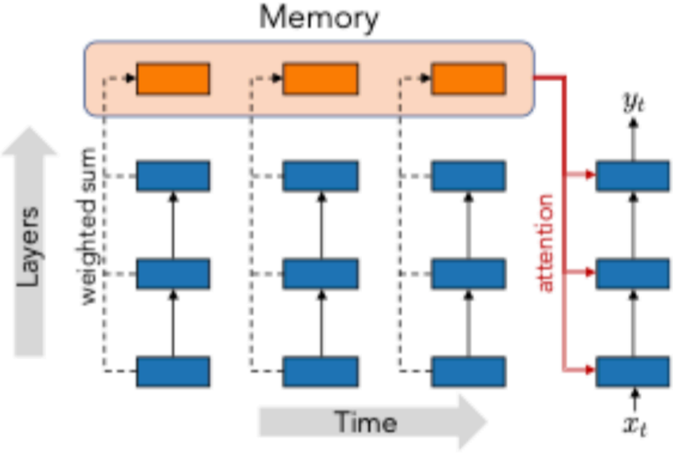


Figure 1: The **Feedback Transformer** merges past hidden representations from all layers into a single vector and stores it in memory.

LIMITED ACCESS TO LONG MEMORY
LIMITED STATE UPDATE

Task		Trans-former	Feedback Trans.
Copy	Char	59.1	76.2
	Seq	6.2	23.6
Reverse	Char	50.2	74.8
	Seq	5.9	29.2
Counting	Len 50	99.6	99.7
	Len 1K	82.4	95.3
Random Walk		68	100
Algorithmic	3 vars	33.7	99.1
	5 vars	37.5	92.6

Table 1: **Accuracy on toy tasks.** Char is character accuracy, Seq is sequence accuracy.

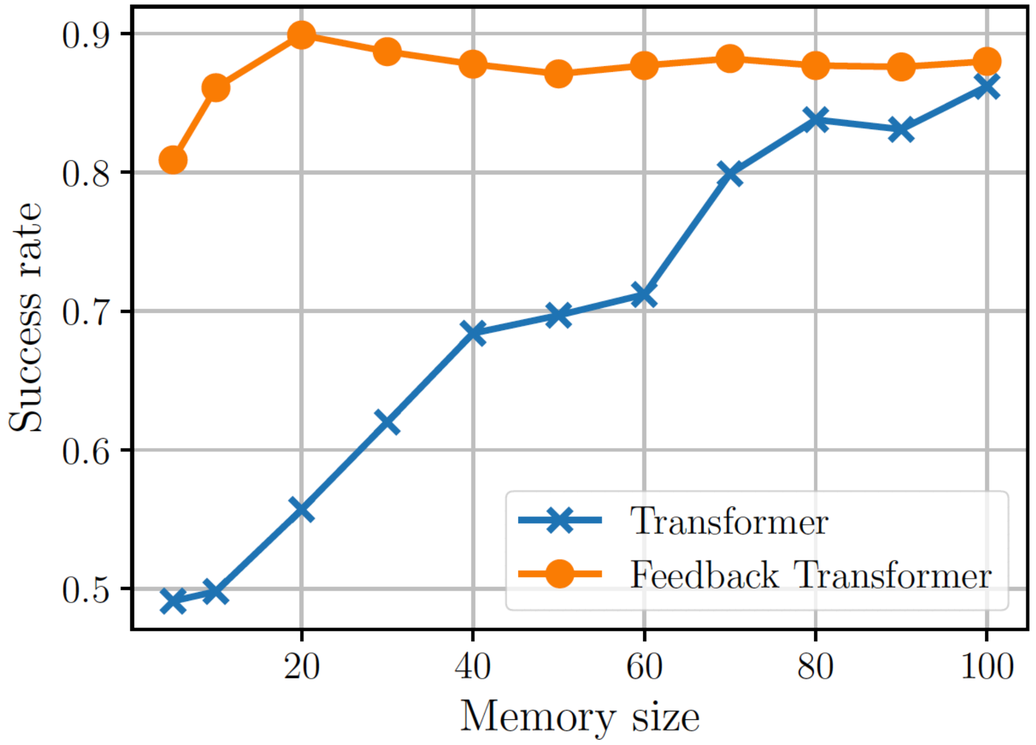


Figure 3: **Results on the Corridor task.** The Transformer degrades as the memory size decreases, but the Feedback Transformer maintains performance.

STRONG PERFORMANCE WITH SMALL, SHALLOW MODELS

LONG MEMORY TRACKS STATE

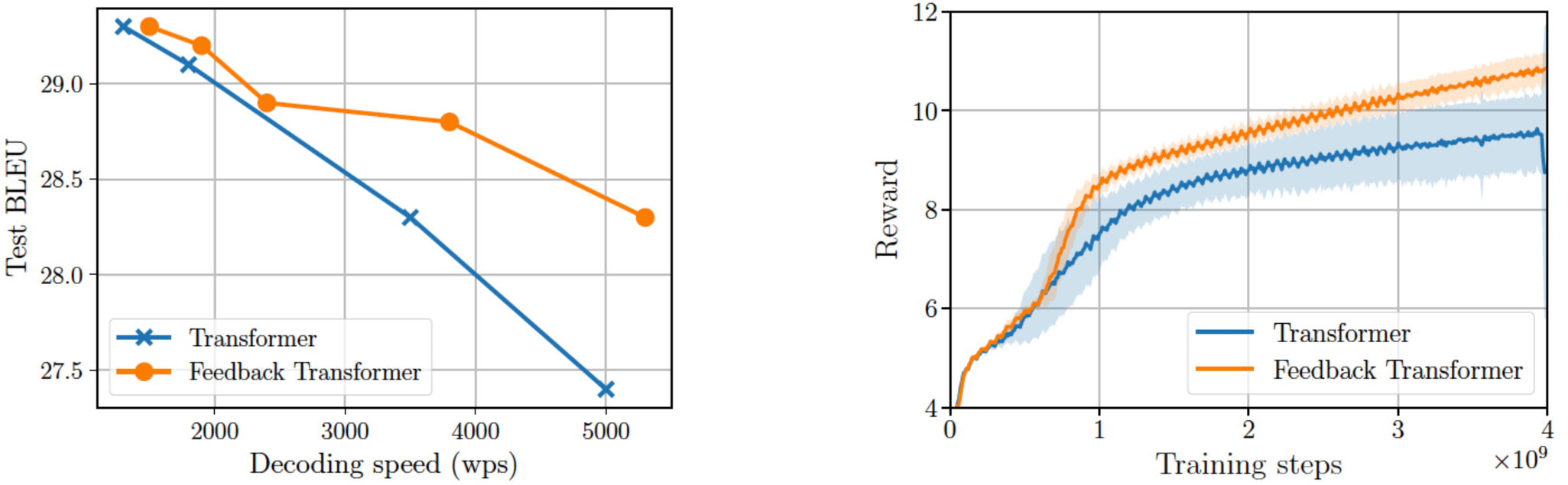


Figure 4: **(left) Machine Translation on `WMT14 En-De`,** test set BLEU and decoding speed in words-per-second for varying decoder depths. **(right) Maze Navigation in Gridworld.** We display average reward comparing Feedback Transformer to standard Transformers.

COMPARISON TO RECURRENT ARCHITECTURES

Model	Test
Recurrent Architectures	
DenseNMT Shen et al. (2018)	25.5
RNMT+ (Chen et al., 2018)	28.5
Hybrid Architectures	
BiARN (Hao et al., 2019)	28.9
SRU (Lei et al., 2017)	28.4
Transformer Architectures	
Transformer (Vaswani et al., 2017)	28.4
Transformer (Ott et al., 2018)	29.3
Feedback Transformer	29.5

Table 2: **Results on** `WMT En-De` comparing the Feedback Transformer to Recurrent architectures, hybrid Recurrent-Transformer models, and standard Transformers.

MEMORY COMPOSITION

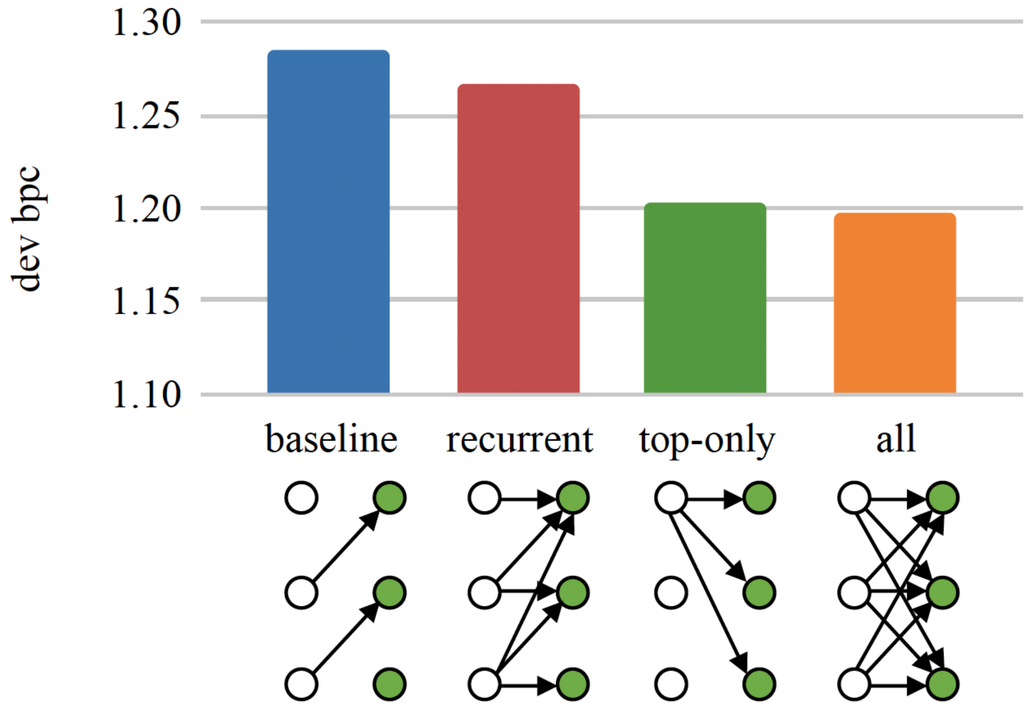


Figure 5: Comparison of different memory composition strategies on `char-PTB`. The recurrent connection alone is not as effective as feedback connections from a higher layer.

COMPARISON TO OTHER TRANSFORMER ARCHITECTURES

Model	Params	Test
Best Existing (Roy et al., 2020)	—	15.8
Trans-XL (Dai et al., 2019)	257M	18.3
Our Transformer	140M	19.9
Feedback Transformer	126M	18.3

Table 3: **Results on `WikiText-103`.** We report perplexity on test.

Model	Params	Test
Best Existing (Rae et al., 2020)	277M	0.97
Trans-XL (Dai et al., 2019)	277M	0.99
Feedback Transformer	77M	0.96

Table 4: **Results on `Enwiki8`.** We report bit-per-byte on test.

Task	Model	Training Speed	Inference Speed
Language Modeling	Transformer	296K	592
	Feedback Transformer	84.4K	2176
Translation	Transformer	280K	3190
	Feedback Transformer	126K	5410
Reinforcement Learning	Transformer	22.3K	—
	Feedback Transformer	22.3K	—

Table 5: **Results comparing Training and Inference Speed** for three different tasks. For language modeling, we measure words-per-second on Wikitext-103 fixing model size and attention span. For translation, we measure words-per-second on WMT En-De, both models with a 6 layer encoder and 2 layer decoder. For RL, we measure the training frame-per-second on maze navigation (with 20 CPU cores and 1 GPU). All inference speed is reported on 1 GPU.

Contributions

- For each timestep the model merges hidden representations of all layers into a high-level single vector as a memory. For the current timestep it attends memory.
- Model can speedup autoregressive generation
- Model can directly utilize previous high-level representations thus requires smaller size and shallow layers to achieve comparable performance