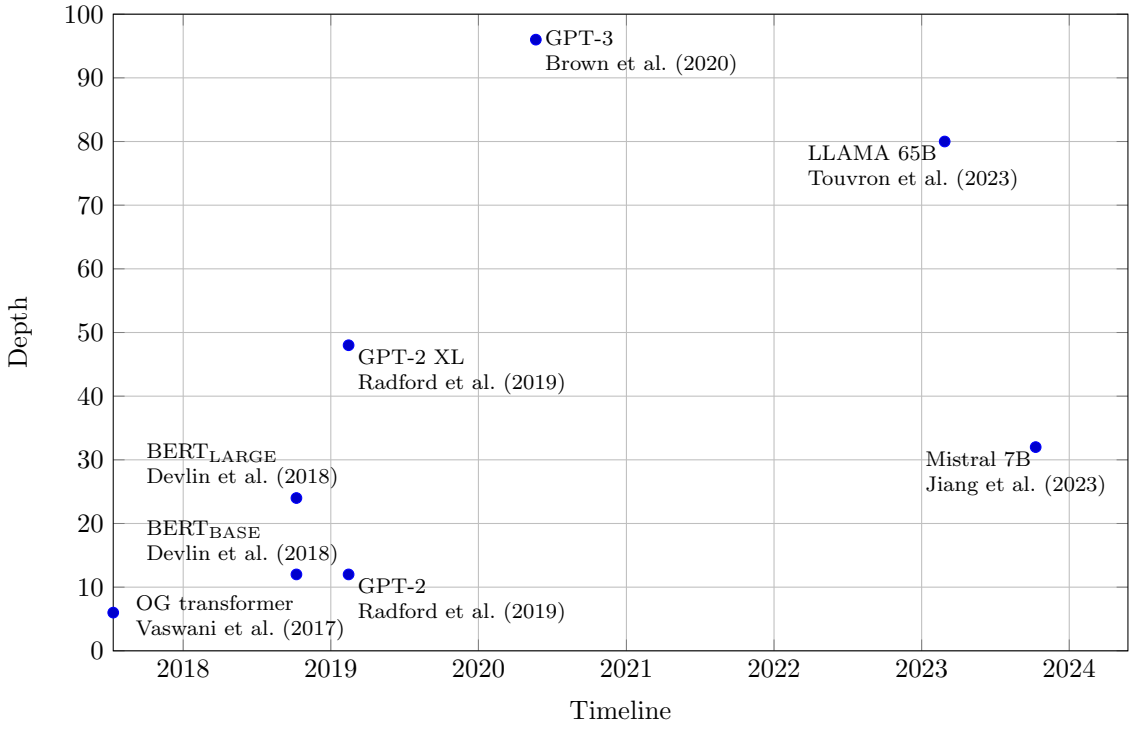
It's not that deep! (or is it?)

October 27, 2024



1 Depth

It seems commonly accepted that in "deep learning", we should be making our models deeper. Indeed, it seems fashionable to make bigger and bigger models. Is this necessary? To start, we can consider the depth of other kinds of networks

- In feed-forward neural networks, two layers are enough, as shown in the universal approximation theorem Hornik et al. (1990) (but they have to get very wide).
- Convolutional neural networks vary, but large ones often have over a hundred layers, like in ResNet-152 He et al. (2016).
- Transformers have been getting larger over time (some famous ones shown in the chart) but not as big as CNNs (yet?).

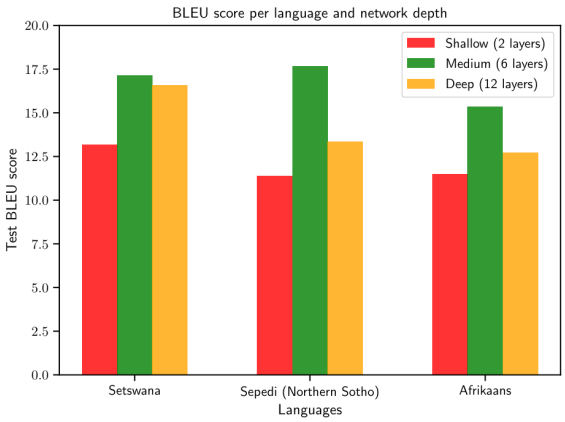
So one might wonder, what does depth give us in transformers? When do we need more depth? When should we not use more depth?

2 Shallower transformers

In fact, there are many reasons why we wouldn't want a deeper transformer. These include computational cost and difficulty of optimization, which affect the architecture you want to you in a particular setting.

2.1 Low resource settings

Murray et al. (2019) find that on low resource language translation, smaller models can be better. In fact, completely deleting the FFN from the baseline model still results in improved BLEU in some cases. Indeed, Van Biljon et al. (2020) find that with 3 layers is better than the standard 6 on low resource translation tasks. (they count both encoder and decoder layers so it's doubled)



They tested English to Setswana (123868), Sepedi (30777), and Afrikaans (53172). Their medium sized models outperform the previous BLEU baselines (and for Sepedi, doubles the previous best).

2.2 Trading width for depth

The paper "Wide Attention Is The Way Forward For Transformers?" by Brown et al. (2022) compared 8 layer transformers with 1 layer transformers with many heads on IMDb classfication and other tasks. The model sizes and tasks are very limited, but they do find marginal improvements in this setting. (The results are very limiting, and it appears after a round of critical reviews they appended a ? to their title)

Table 2: Test accuracy on all tasks for deepest and widest models.

Attention	**IMDb Token**		**IMDb Byte**		**Listops**		**Doc Matching**		**Average**	
Type	Deep	Wide	Deep	Wide	Deep	Wide	Deep	Wide	Deep	Wide
BigBird	86.0	85.3	62.7	62.4	36.7	37.3	63.9	64.0	62.3	62.3
Linear	86.7	88.0	64.5	64.7	37.1	37.4	64.2	63.9	63.1	63.5
Linformer	84.2	84.1	56.3	52.6	29.9	37.0	64.5	63.1	58.7	59.2
Local	74.2	73.9	55.5	57.0	36.8	37.7	58.0	58.1	56.1	56.7
Longformer	86.0	84.1	61.6	57.5	36.8	37.7	61.2	58.1	61.4	59.4
Performer	87.0	87.0	64.4	64.8	36.2	36.6	65.0	66.3	63.1	63.7
Sinkhorn	86.1	86.5	62.3	61.6	19.4	21.7	64.0	65.7	57.9	58.9
Sparse	85.7	85.7	61.2	62.9	37.0	36.9	63.2	63.6	61.8	62.3
Synthesizer	86.7	86.5	61.4	61.1	36.5	37.3	71.1	72.3	63.9	64.3
Transformer	85.8	85.5	62.6	62.4	35.7	37.2	64.0	64.4	**62.0**	**62.4**
Average	84.8	84.7	61.2	60.8	34.2	35.7	63.9	64.0	**61.0**	**61.3**

2.3 Rank Collapse/Oversmoothing/Token Uniformity Problem

Three different names for the same phenomena, that repeated application of self-attention layers gradually reduces the dimensionality of the word embedding space. Dong et al. (2021) show that pure self-attention networks lose rank doubly exponentially with depth. However, residual connections can prevent this problem, and FFN can also slow it down. This suggests that making your networks deeper introduces problems not found in shallower networks, which may or not be desirable.

3 Deeper Transformers

On the other hand, there are reasons to believe more layers are helpful. Here are a couple examples, but there are many, many more.

3.1 High resource settings

We can look at another instance of MT. In this case, Liu et al. (2020) train transformers for WMT'14 English-French (36M sentences) and English-German (4.5M), using 60 encoder layers and 12 decoder layers. Note that they had to use some special initialization trick, called ADMIN, to make the deep transformer trainable. There were improvements over the baseline.

Model	WMT'14 English-French (FR) #param	T↓	M↑	BLEU↑	Δ	WMT'14 English-German (DE) #param	T↓	M↑	BLEU↑	Δ
6L-6L Default	67M	42.2	60.5	41.3	-	61M	54.4	46.6	27.6	-
6L-6L ADMIN	67M	41.8	60.7	41.5	0.2	61M	54.1	46.7	27.7	0.1
60L-12L Default	262M	diverge				256M	diverge			
60L-12L ADMIN	262M	**40.3**	**62.4**	**43.8**	2.5	256M	**51.8**	**48.3**	**30.1**	2.5

Table 1: Test results on WMT'14 benchmarks, in terms of TER (**T**↓), METEOR (**M**↑), and BLEU. Δ shows difference in BLEU score against baseline 6L-6L Default. Best results are boldfaced. 60L-12L ADMIN outperforms 6L-6L in all metrics with statistical significance ($p < 0.05$). Following convention, BLEU is computed by `multi-bleu.perl` via the standardized tokenization of the publicly-accessible dataset.

BLEU via `multi-bleu.perl`	FR
36L-12L-768D ADMIN + BT	**46.4**
60L-12L ADMIN + BT	46.0
BT (Edunov et al., 2018)	45.6
60L-12L ADMIN	43.8
BLEU via `sacreBLEU.py`	**FR**
36L-12L-768D ADMIN + BT	**44.4**
60L-12L ADMIN + BT	44.1
60L-12L ADMIN	41.8
BT (Edunov et al., 2018)	-

Table 4: Back-translation results on WMT'14 EN-FR.

They hypothesize that deeper models can better exploit on big and noisy datasets. For instance, back-translation on French WMT added 21.8M more examples. However it appears

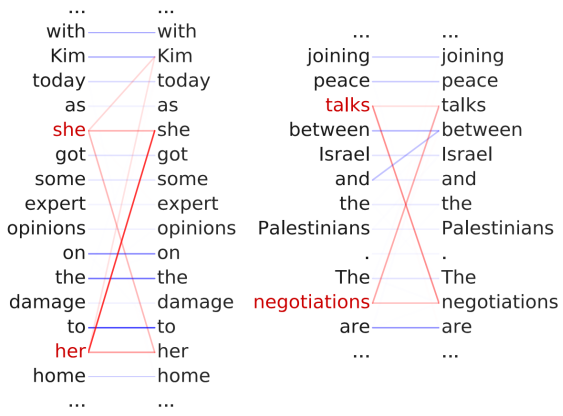
3.2 What are later layers doing

If more layers are helpful, then what are the later layers doing? Here, Clark et al. (2019) train 12 layer BERT encoders and probe to see what the attention heads are doing. Many interesting observations were found in the paper, that different layers had qualitatively different purposes. One note is that they find many heads which attend to the directly following token in the earlier layers 1-6, but not later. For a few more examples, see this list, where they notate each head with its (layer)-(head index).

Head 8-10 - Direct objects attend to their verbs
Head 8-11 - Noun modifiers (e.g., determiners) attend to their noun
Head 7-6 - Possessive pronouns and apostrophes attend to the head of the corresponding NP
Head 4-10 - Passive auxiliary verbs attend to the verb they modify
Head 9-6 - Prepositions attend to their objects
Head 5-4 - Coreferent mentions attend to their antecedents

Head 5-4

- **Coreferent** mentions attend to their antecedents
- 65.1% accuracy at linking the head of a coreferent mention to the head of an antecedent



3.3 Induction Heads

Another neat construction is an "induction head"m which Elhage et al. (2021) find emerge in two-layer transformers, but not one-layer ones. At a token a, the induction head looks at the previously occurring a, and checks the token b that comes after that a. Then, it predicts b as the next token after the current position. This really is a two-layer construction.

The first layer has heads which attend to the previous token. Then when on token a, the second layer looks at the previous attention layer to find tokens b preceded by an a. Then it predicts b.

4 Some Theoretical Results

It would be cool to have something like the result for feed-forward networks, that we can always trade depth for width in terms of expressivity, but we don't really have those. Here are some relevant results in this direction, though.

4.1 Logarithmic Depth

In this week's FLaNN talk, Daniel Hsu presented Sanford et al. (2024), which used some ideas from Massively Parallel Computation (MPC) theory to talk about transformer sizes.

Here's one result. tl;dr, assuming a well accepted conjecture in MPC theory, they find a lower bound on the depth of transformers needed to solve graph connectivity, assuming a bound on the width of the transformer.

Corollary 4.1. *Let $\epsilon \in (0,1)$ be any constant, and let $D \geq N^{\epsilon}$. Assume Conjecture 2.4, and suppose there exists $T \in Transformer^{N}_{m,L,H}$ with $mH = O(D^{1-\epsilon})$ that decides connectivity of any input graph with connected components having diameter $\leq D$. Then $L = \Omega(\log D)$.*

However, this does not suggest that a smaller transformer exists. In particular, Daniel suggested in the talk that mH might have to be linear in D.

4.2 Depth-Width Tradeoffs

Here Levine et al. (2020) prove and experimentally verify some ideas about depth-efficiency. tl;dr a network has to be very wide in order for making it deeper to help.

When depth L is below some threshold $\log(d)$ based on the width d, then it is more efficient to add depth than to make it wide. That is, any shallower network simulating a network of depth L would need an exponential increase in width.

Their results rely on the notion of "separation rank", which roughly means how hard it is to model different kinds of dependencies between different parts of the input. For instance, convolutional neural networks have high separation rank, because passing info from one side to another takes many layers, depending on kernel size. But transformers have low separation rank, because self attention layers allow positions to all look at each other.

Corollary 4.2. *With probability 1, the function realized upon randomization of the weights of a deep self-attention network with depth L^{deep} and with $d_x^{deep} > 3^{L^{deeP}}$ may only be realized by a shallower network with depth $L^{shallow} = \frac{L^{deep}}{d}$ and width $d_x^{shallow} = wd_x^{shallow}$, where $d > 1, w > 1$ (i.e., the deep network is deeper by a factor of d and the shallow network is wider by a factor of w), if the following holds:*

$$w \propto \exp(\exp(d))$$

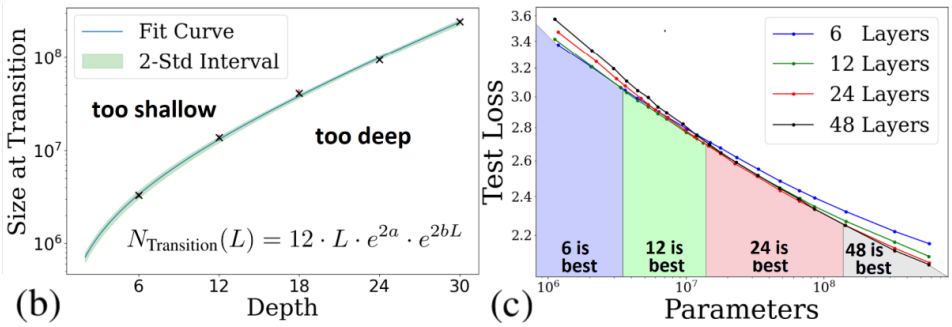
Again this is a bound, doesn't say that a shallower network actually exists. It is necessary for the width to get this big in order to simulate the separation rank of a deeper network, but this doesn't mean it's sufficient.

When depth is large, $D > \log_3(d_x)$, they say only an upper bound of polynomial width is needed to simulate the separation rank of large depth with shallower. Again, they only conjecture that such a transformer exists.

Corollary 4.3. *Let y^{deep} denote the function realized by a deep self-attention network at any output location $i \in [N]$, with depth and width denoted L^{deep}, d_x^{deep} such that $L^{deep} > \log_3 d_x^{deep}$. Denote $\beta_1 := \frac{\log_3 d_x^{deep}}{L^{deep}} < 1$. Then, there exists $\beta_2 = O(\log(H) \cdot \log(d_x^{deep}) \cdot \log(L^{deep}))$ such that the function realized by a network of depth: $L^{shallow} = \beta_1 \cdot L^{deep} + \beta_2$, and width: $d_x^{shallow} = 3^{\beta_2 d_x^{deep}}$, denoted $y^{shallow}$, has higher separation rank, i.e.:*

$$sep(y_p^{shallow}) > sep(y_{p'}^{deep}) \quad ; \quad where\ p, p' \in [d_x]$$

The above corollary, which follows from theorems 1 and 2, shows that the separation rank of a function realized by a self-attention network of arbitrary depth $L > \log_3(d_x)$ can be surpassed by a shallower network of polynomial width, contrarily to the established behavior for networks of depth $L < \log_3(d_x)$.



It is worth noting that their experiments are decoder language modeling on English Wikipedia, BookCorpus and OpenWebText, with a small vocab size of 2000 in order to control the embedding dimensions. Also, their theoretical analysis ignores all nonlinear activations and layernorm.

5 Ideas and Questions

5.1 Masked Hard Attention

In a recent revision, (Yang et al., 2024) showed for unique hard-attention transformers, more layers always gains more expressivity. This relies on the proof from Etessami and Wilke (2000) that linear temporal logic formulas get more expressive the deeper you make them.

5.2 Soft Attention

A naive extension of that result for log-precision softmax transformers would require showing that $\mathbf{TC}^0$ get more expressive with more depth, and doing so would resolve on of the main open questions in circuit complexity (whether $\mathbf{TC}^0 = \mathbf{NC}^1$), so it is likely not easy.

It appears that Keisler and Lotfallah (2011) suggest an affirmative answer to this question, but only when the structures are graphs. That is, I played around with the proof for a while, but could not get it to work for strings. I have been working with some restrictions $\mathbf{TC}^0$ that might be easier to prove something about. Will write up sometime later.

References

Jason Ross Brown, Yiren Zhao, Ilia Shumailov, and Robert D Mullins. 2022. Wide attention is the way forward for transformers? *arXiv preprint arXiv:2210.00640* (2022).

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341* (2019).

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

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. 2021. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International Conference on Machine Learning*. PMLR, 2793–2803.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread* 1 (2021), 1.

Kousha Etessami and Thomas Wilke. 2000. An until hierarchy and other applications of an Ehrenfeucht–Fraıssé game for temporal logic. *Information and Computation* 160, 1-2 (2000), 88–108.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. 1990. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural networks* 3, 5 (1990), 551–560.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

H Jerome Keisler and Wafik Boulos Lotfallah. 2011. Rank hierarchies for generalized quantifiers. *Journal of Logic and Computation* 21, 2 (2011), 287–306.

Yoav Levine, Noam Wies, Or Sharir, Hofit Bata, and Amnon Shashua. 2020. The depth-to-width interplay in self-attention. *arXiv preprint arXiv:2006.12467* (2020).

Xiaodong Liu, Kevin Duh, Liyuan Liu, and Jianfeng Gao. 2020. Very deep transformers for neural machine translation. *arXiv preprint arXiv:2008.07772* (2020).

Kenton Murray, Jeffery Kinnison, Toan Q Nguyen, Walter Scheirer, and David Chiang. 2019. Auto-sizing the transformer network: Improving speed, efficiency, and performance for low-resource machine translation. *arXiv preprint arXiv:1910.06717* (2019).

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

Clayton Sanford, Daniel Hsu, and Matus Telgarsky. 2024. Transformers, parallel computation, and logarithmic depth. *arXiv preprint arXiv:2402.09268* (2024).

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

Elan Van Biljon, Arnu Pretorius, and Julia Kreutzer. 2020. On optimal transformer depth for low-resource language translation. *arXiv preprint arXiv:2004.04418* (2020).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

Andy Yang, David Chiang, and Dana Angluin. 2024. Masked Hard-Attention Transformers Recognize Exactly the Star-Free Languages. In *Advances in Neural Information Processing Systems (NeurIPS)*. `https://arxiv.org/abs/2310.13897` To appear.