The Impact of Depth on Compositional Generalization in Transformer Language Models

Anonymous ACL submission

Abstract

To process novel sentences, language models (LMs) must generalize compositionally—combine familiar elements in new ways. What aspects of a model's structure promote compositional generalization? Focusing on transformers, we test the hypothesis, motivated by theoretical and empirical work, that deeper transformers generalize more compositionally. Simply adding layers increases the total number of parameters; to address this confound between depth and size, we construct three classes of models which trade off depth for width such that the total number of parameters is kept constant (41M, 134M and 374M parameters). We pretrain all models as LMs and fine-tune them on tasks that test for compositional generalization. We report three main conclusions: (1) after fine-tuning, deeper models generalize more compositionally than shallower models do, but the benefit of additional layers diminishes rapidly; (2) within each family, deeper models show better language modeling performance, but returns are similarly diminishing; (3) the benefits of depth for compositional generalization cannot be attributed solely to better performance on language modeling or on in-distribution data.

1 Introduction

The number of possible sentences in natural language is enormous; regardless of the size of its training set, a language model (LM) will regularly encounter sentences it has never seen before. The ability to interpret such sentences relies on compositional generalization: the capacity to combine familiar words and syntactic structures in new ways (Montague, 1970; Fodor and Pylyshyn, 1988). Transformer LMs (Vaswani et al., 2017), while highly successful in many settings, often struggle when tested on benchmarks that require compositional generalization (Kim and Linzen, 2020). What architectural factors affect a transformer's ability to generalize compositionally?

In this paper, we test the hypothesis that increasing a transformer's depth—the number of layers it has—improves its performance on tasks that require compositional generalization. This hypothesis is motivated both by theoretical work, which has shown that adding layers increases the expressive capacity of neural networks in general (Raghu et al., 2017) and in transformers in particular (Merrill et al., 2021), and by experimental work suggesting that deeper models generalize more compositionally than shallower ones (Mueller et al., 2022; Murty et al., 2022).

While existing empirical work lends some credibility to this hypothesis, to directly confirm it we must address the confound between depth and size (number of parameters). As each additional layer introduces a new set of parameters, deeper models are also larger, all else being equal. LMs' performance on a wide variety of tasks is correlated with their size (Kaplan et al., 2020; Hoffmann et al., 2022; Muennighoff et al., 2023). To disentangle these two factors, we construct classes of models with equal total number of parameters but differing depths; we do so by reducing the model's feed-forward dimension to compensate for added depth. We pretrain all models on language modeling and fine-tune them on four compositional generalization tasks. COGS (Kim and Linzen, 2020), COGS-vf (Qiu et al., 2022a), GeoQuery (Zelle and Mooney, 1996), and the English passivization portion of Multilingual Transformations (Mueller et al., 2022).

In addition to any possible direct effect on compositional generalization, depth may also be correlated with other factors which may themselves predict compositional generalization, such as language modeling loss during pretraining or in-domain fine-tuning performance. This complicates the interpretation of any relationship we might find between depth and generalization performance. To address this concern, we also investigate and correct for the

effect of depth on language modeling performance and in-distribution loss.

We report the following findings, across three model size classes (41M, 134M, and 374M parameters): (1) In general, deeper models have lower perplexity (Section 3.1). The marginal increase in performance gained from additional layers diminishes rapidly as models get deeper, and performance begins to degrade when the feed-forward dimension approaches the dimensionality of the model's contextualized embeddings. (2) In general, deeper models display better compositional generalization (Section 3.2). Again, most of the benefit of depth accrues from the first few layers; for several of the compositional generalization benchmarks we use, performance saturates very quickly as models get deeper. (3) Deeper models generalize more compositionally even after correcting for the fact that their language modeling perplexity is lower and their in-distribution performance on the fine-tuning task is higher (Section 3.3).

2 Methodology

2.1 Constructing Families of Models with Equal Numbers of Parameters

To make a transformer LM deeper without increasing the total number of parameters, we need to also make it narrower. There are several ways to do so: we can reduce the size of the feed-forward dimension d_{ff}, reduce the size of the residual stream (the embedding size) d_{model}, or reduce the size of the attention outputs d_{attn} (see Appendix B for a diagram of a transformer layer annotated with dimensionality labels). Vaswani et al. (2017) coupled these variables at $d_{\text{model}} = d_{\text{attn}} = d_{\text{ff}}/4$. Most transformer LMs have adopted this ratio (Devlin et al., 2019; Kaplan et al., 2020; Hoffmann et al., 2022, *inter alia*), though Raffel et al. (2019) increased the size of d_{ff} relative to d_{model} and d_{attn} for their two largest models. By contrast, we vary d_{ff} with depth (while holding $d_{\text{model}} = d_{\text{attn}}$ constant). By keeping the attention mechanism identical across models of varying depths, we rule out the possibility that depth will be confounded with the capacity of the self-attention mechanism. We refer to $d_{\text{model}}/d_{\text{ff}}$, conventionally set to $1/4$, as the *feed-forward ratio*.

Deriving hyperparameter relations As a starting point for our size classes of models, we use hyperparameters taken from the T5-base and T5-large size classes (Raffel et al., 2019) as well as a smaller model from Kim and Linzen (2020) which has identical layer-internal hyperparameters to T5-small but fewer layers. We implement models using `t5x` (Roberts et al., 2022). We then calculate how much the size of the feed-forward dimension must change to accommodate adding or removing layers. Starting from the parameter formula in Kaplan et al. (2020), the number of parameters M in a layer is

$$M(d_{\text{ff}}) = 2d_{\text{model}}d_{\text{ff}} + 4d_{\text{model}}d_{\text{attn}} = \beta \cdot d_{\text{ff}} + A,$$

where the constant β represents the contribution of the parameters of the feed-forward block which project vectors from $\mathbb{R}^{d_{\text{model}}}$ into $\mathbb{R}^{d_{\text{ff}}}$ and back into $\mathbb{R}^{d_{\text{model}}}$; and the constant A represents the parameters of everything aside from the feed-forward block, including the attention mechanism. The total parameter count of a full model N in terms of d_{ff} and n_{layers} is then

$$\begin{aligned} N(n_{\text{layers}}, d_{\text{ff}}) &= n_{\text{layers}} \cdot M(d_{\text{ff}}) + 2d_{\text{model}}n_{\text{vocab}} \\ &= n_{\text{layers}} \cdot M(d_{\text{ff}}) + E, \end{aligned}$$

where E represents the parameters of the vocabulary embedding and unembedding transformations. Given initial values $(n^0_{\text{layers}}, d^0_{\text{ff}})$ which characterize the baseline model in each size class (e.g., T5-large), our goal is to find pairs $k, w(k)$ such that

$$N(n^0_{\text{layers}} + k, d^0_{\text{ff}} - w(k)) = N(n^0_{\text{layers}}, d^0_{\text{ff}}).$$

Solving for w as a function of k tells us how much to increase (or decrease) d^0_{ff} if we remove (or add) k layers from an existing model:

$$w(k) = \left\lfloor \left(1 - \frac{n^0_{\text{layers}}}{n^0_{\text{layers}} + k}\right)\left(d^0_{\text{ff}} + \frac{A}{\beta}\right)\right\rceil. \quad (1)$$

Since adding or removing k layers might require changing d^0_{ff} by a fractional amount, we round $w(k)$ to the nearest integer; while this means that our models may not be exactly equal in total parameter count, the differences are very small relative to N. Table 1 reports the exact hyperparameter values we use for each of our three size classes, derived from Equation 1 above, and Figure 1 shows each size class plotted as $(n_{\text{layers}}, d_{\text{ff}})$ pairs.

2.2 Datasets and Training

2.2.1 Language Modeling

We use the Colossal Clean Crawled Corpus (C4; Raffel et al. 2019) as our pretraining corpus. C4 was created by filtering data from the Common

	41M							134M										374M								
n_{layers}	1	**2**	3	4	5	6	7	1	2	4	6	8	**12**	16	21	26	32	1	2	4	6	8	12	16	**24**	32
d_{ff}	4779	**2048**	1138	682	409	227	97	36k	17k	8193	5121	3584	**2048**	1280	731	393	128	99k	49k	24k	15k	11k	6998	4907	**2816**	1770
	$d_{\text{model}} = d_{\text{attn}} = 512,\ n_{\text{heads}} = 8$							$d_{\text{model}} = d_{\text{attn}} = 768,\ n_{\text{heads}} = 8$										$d_{\text{model}} = d_{\text{attn}} = 1024,\ n_{\text{heads}} = 64$								

Table 1: Models of varying depths across three size classes. Bolded variants are the baseline models whose hyperparameters were taken from Kim and Linzen (2020) and Raffel et al. (2019).

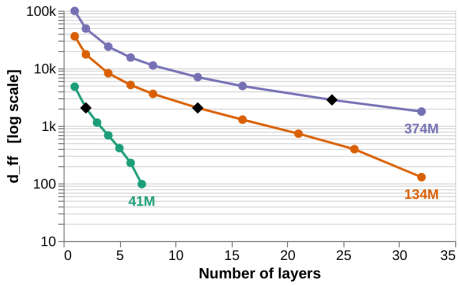


Figure 1: Models for the 41M-, 134M-, and 374M-parameter size classes. Points indicate models trained in this paper, and black diamonds represent the baseline models for each class whose hyperparameters were taken from Kim and Linzen (2020) and Raffel et al. (2019).

Crawl dataset of scraped English-language web files to serve as a language modeling corpus. We use a context size n_{ctx} of 1024 tokens and a batch size of 128 sequences ≈ 131k tokens. We pretrain each model for 1M steps, resulting in a total training dataset of roughly 131B tokens.

2.2.2 Compositional Generalization

In compositional generalization datasets, models are tested on a distribution that contains novel combinations of pieces, each of which has been previously seen independently during training. We fine-tune our pretrained models on the training portion of the dataset for 10,000 steps, measuring in-distribution generalization accuracy (validation accuracy) every 250 steps. Validation loss continued to decrease throughout training runs on each dataset, so we report values from the end of each fine-tuning run without early stopping. We use four compositional generalization datasets which are intended to be used as benchmarks for compositionality (for examples of instances of these tasks, see Table 2):

1. **COGS** (Kim and Linzen, 2020) is a semantic parsing dataset introduced to serve as a test for compositional generalization. It consists of natural-language sentences paired with formal semantic representations, and is constructed such that the out-of-domain generalization distribution contains two generalization types: new combinations of familiar words (*lexical generalization*, such as using the word 'hedgehog' as the object of a sentence when this word has only been seen during training as a subject); or using known words in new syntactic structures (*structural generalization*, such as relative clauses that are more deeply nested than seen in training).

2. **Variable-free COGS** (COGS-vf; Qiu et al. 2022a) is a simplified variant of COGS where the semantic representations are converted into a form which does not use numbered variables (see Table 2 for a comparison between COGS and COGS-vf). Removing variables from the representation has the benefit of lowering the associated computational cost of training by making sequences meaningfully shorter. This conversion has been previously shown to improve the performance of models by reducing the complexity of the output space (Qiu et al., 2022b), but comes at the cost of limiting the capacity of the formal language to represent many phenomena in natural language which require coordination of variable identity, such as control and anaphor binding.

3. **GeoQuery** (Zelle and Mooney, 1996) contains natural-language questions about US geography paired with SQL-style database queries representing those questions. We report results on the GeoQuery Standard split.

4. **English passivization** (Mueller et al., 2022) is a dataset of English active-voice sentences paired with their passive-voice counterparts (adapted from Mulligan et al. 2021). This benchmark is designed to test whether models use shallow, positional heuristics or syntactically-sensible ones. While Mueller et al. (2022) implemented a number of transformations in different languages, we focus on the English Passivization task.

COGS	x: A hedgehog ate the cake . y: *cake(x_4); hedgehog(x_1) AND eat.agent(x_2, x_1) AND eat.theme(x_2, x_4)
COGS-vf	x: A hedgehog ate the cake on the bed . y: eat(agent = hedgehog, theme = *cake(nmod.on = *bed))
GeoQuery	x: which states have cities named m0 y: answer(intersection(state, loc_1(intersection(city, m0))))
English Passivization	x: our vultures admired her walrus above some zebra . y: her walrus above some zebra was admired by our vultures .

Table 2: Examples of inputs (x) & targets (y) from each compositional generalization dataset.

3 Results

3.1 Language Modeling

Deeper models have lower perplexity. We find that depth has a significant impact on model performance. At the shallow end of the spectrum, increasing model depth results in a dramatic improvement in perplexity (Figure 2). In Figure 3a we compare the perplexity of each model in a size class relative to that of the best-performing model of that size. In the extreme case, the perplexity of a single-layer model can be nearly twice that of the optimal model in the class. Moreover, as parameter count increases the disparity between the worse, shallower models and the better, deeper models increases as well: For 41M-parameter models the ratio between the perplexity of the single-layer model and that of the optimal (5-layer) model is 1.59; for the 134 M-parameter models, the ratio is 1.86; and for the 374M-parameter models, the ratio is 1.99.

Performance increases most rapidly within the first few layers. While deeper models do, in general, perform better than shallower ones, the increase in performance that comes from adding layers diminishes rapidly as models become deeper (Figure 3a). The performance difference between 1-layer and 2-layer models is dramatic across all size classes; moving from 2 to 4 layers results in a much more modest performance improvement. We also note that as models get larger in our setup, they are able to make productive use of increasingly more layers: the optimal 41M-parameter model in our setup has 5 layers, while the optimal 134M-parameter model has 12; among 374M-parameter models, the 24-layer model had the best performance. At the same time, the pattern of the diminishing utility of depth holds even for the largest models we study.

Performance starts degrading when models become too narrow. At the deeper end of our scale, adding layers is not only unhelpful for performance, but begins to harm it (see the right-hand sides of each size-class curve in Figure 3a). As previously noted, the point at which trading width for depth becomes harmful is not an absolute function of depth, since the optimal models from each size class have differing depths. However, comparing the relative performance of models within a size class to the feed-forward ratio $d_{\text{model}}/d_{\text{ff}}$ shows that model performance begins to worsen once d_{ff} becomes smaller than d_{model} (to the right of the red dashed line in Figure 3b); when this happens, the affine projection of the vectors from $\mathbb{R}^{d_{\text{model}}}$ into $\mathbb{R}^{d_{\text{ff}}}$ becomes a non-injective map. In Appendix D we analyze the weight matrices of the affine transforms in the feed-forward network of each layer and demonstrate that as $d_{\text{model}}/d_{\text{ff}}$ increases the transforms become increasingly rank-deficient.

Larger models are more robust to changes in the feed-forward ratio. Varying d_{ff} while keeping d_{model} constant results in feed-forward ratios $d_{\text{model}}/d_{\text{ff}}$ which deviate significantly from the standard ratio of $1/4$ (black vertical rule in Figure 3b). We find that smaller models are more sensitive to the particular value of the feed-forward ratio, and that for small models the standard ratio may not be optimal. Within the 41M-parameter size class there is a narrow range of feed-forward ratios in which model performance is within a few percentage points of the best-in-class model. As models get larger, this range expands leftward to include models which have increasingly wide feed-forward networks relative to the size of their contextual embeddings. This shows that larger models have more leeway to trade depth for width, becoming wider in proportion to their model dimension d_{model} without incurring large penalties for their perplexity. It also shows that when $d_{\text{model}}/d_{\text{ff}} < 1$ the feed-forward ratio no longer serves as a predictor of relative perplexity independent of size.

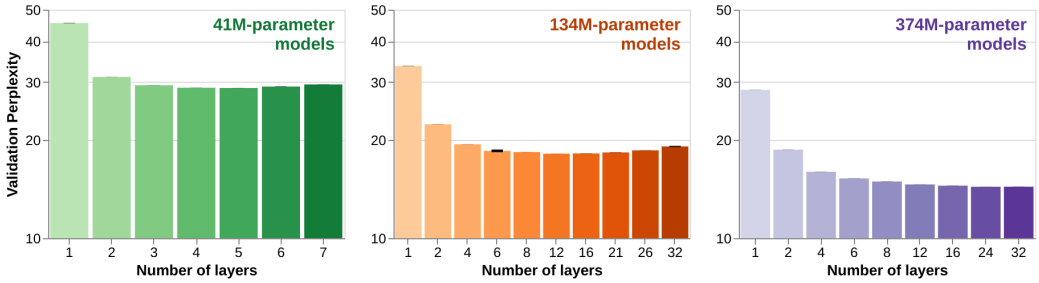
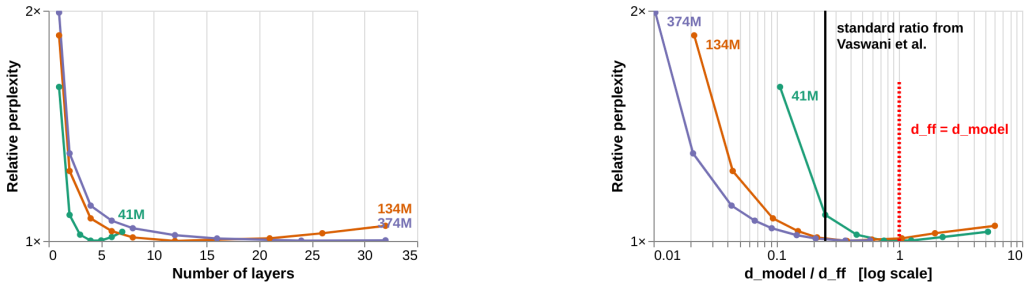


Figure 2: Deeper models achieve lower perplexities than shallower ones after equal amounts of training data regardless of size, but the benefits of adding layers diminish quickly with depth. Mean over 5 runs shown with error bars.



(a) (b)

Figure 3: Relative perplexity compared to the best model in each size class. **(left)** Perplexity goes down rapidly as models get deeper; only a few layers are needed to obtain most of the value of depth. **(right)** When $d_{\text{model}}/d_{\text{ff}} > 1$ (red dashed rule), perplexity slowly increases. As models get larger, the range of $d_{\text{model}}/d_{\text{ff}}$ ratios where performance is close-to-optimal expands leftward to include smaller and smaller values.

3.2 Compositional Generalization

To test the impact of depth on compositional generalization, we fine-tune the models pretrained in the previous section on the training portions of each of the compositional generalization benchmark datasets. We measure the full-sequence (exact match) accuracy of the models on the out-of-distribution generalization set and note several findings:

Deeper models generalize better. As with language-modeling performance, depth has a significant impact on how well models generalize (Figure 4). On each of the datasets, deeper models tend to attain higher generalization accuracies than shallower models in the same size class. The effect of depth on compositional generalization is more variable than it is for language modeling: for COGS, COGS-vf, and GeoQuery we note some small non-monotonicity in the generalization accuracy across different depths. On English Passivization, the 41M- and 134M-parameter model classes show largely-consistent trends where deeper models perform better than shallower ones; the 374M-parameter models do show more significant non-monotonicity, though the deepest models do still outperform the shallowest ones.

The benefit of depth saturates quickly for some tasks. As with language modeling, most of the benefit of depth is gained by having only a few layers. For three of the tasks—COGS, COGS-vf, and GeoQuery—we see threshold depths after which generalization accuracy stays relatively constant as depth increases. These threshold depths are low and constant across model sizes, but vary by dataset: 4–6 layers for COGS, and 2–4 layers for COGS-vf and GeoQuery. Performance on COGS-vf appears to saturate with fewer layers than on COGS despite the fact that the two datasets are equivalent in expressive capacity;[1] this suggests that the saturation we observe on some datasets is closely linked to the complexity of the output representation independent from the complexity of the compositional generalization expressed in the data. On English Passivization, the impact of depth is

[1] As previously noted, COGS can represent phenomena that COGS-vf cannot, but both output representations are sufficiently rich to represent the examples studied here.

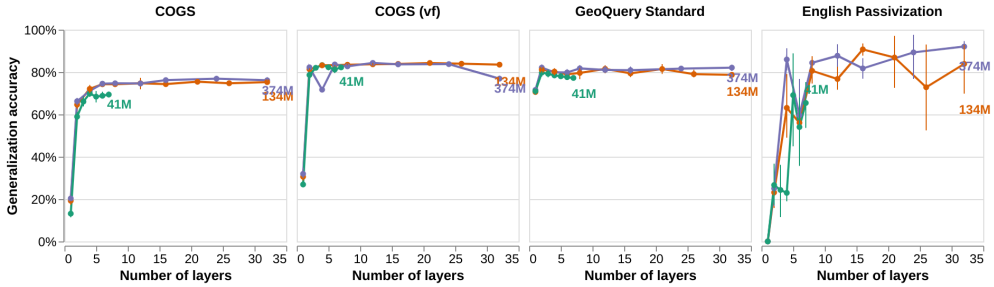


Figure 4: Deeper models generalize better than shallower models on compositional tasks across datasets and size classes. Error bars (easily visible only on the English Passivization data) report 95% confidence intervals in estimation of the mean, taken over 5 runs.

more variable, which makes it difficult to ascertain if a size-independent threshold exists.

The threshold effects suggest that some subsets of the datasets can be addressed with relatively simple models. We investigate this hypothesis using the fact that COGS and COGS-vf include two types of generalization cases: lexical generalization, where a familiar word needs to be interpreted in a familiar syntactic context in which it has not been observed; and structural generalization, where the syntactic structure is novel and needs to be constructed from familiar syntactic pieces. Breaking performance down by the type of generalization required, we find that even deep models at the largest model size systematically fail to generalize structurally (Figure 5); the benefit of depth is largely limited to the easier lexical generalization. This supports the hypothesis that the saturated effect of depth is due to the existence of easier subsets of the datasets, and shows that increasing depth alone does substantially improve the models' ability to learn the correct inductive bias for these structural tasks.

3.3 The Effect of Depth on Generalization is not Solely Attributable to Better Pretraining Loss or In-distribution Performance

Although deeper models generalize better than shallower models do, our pretraining analysis in Section 3.1 shows that deeper models also attain lower validation perplexities on their pretraining corpus than shallower models. Additionally, we observe that deeper models achieve lower in-distribution loss on the fine-tuning tasks than shallower models (Figure 7a). Both of these observations are potential confounds for the interpretation of the previous section: perhaps depth does not *directly* improve generalization accuracy, but only does so indirectly by allowing models to either become better LMs or else to better learn the in-distribution fine-tuning data. To determine whether that this is the case, or whether depth does in fact directly improve generalization, we correct for both of these potential confounds.

First, to correct for the fact that deeper models attain lower pretraining losses, we repeat our fine-tuning experiments using checkpoints of models that have equal validation perplexities within a size class. We pick the least-performant (i.e., shallowest) model within a size class as the "reference model" and note its validation perplexity at the end of pretraining. We then pick the checkpoints of all deeper[2] models at the point when they achieved this reference perplexity (Figure 6a). Finally, we fine-tune each of these checkpoints on the compositional generalization tasks. We repeat this process for successively deeper reference models. We find that even when fine-tuning from checkpoints of equal validation perplexity, deeper models still generalize better than shallower models (Figure 6b). For compositional datasets where we observe thresholding behavior, the benefits of depth continue to hold up through that threshold depth.

Next, we correct for the potentially confounding fact that deeper models learn the in-distribution split of the compositional generalization tasks better than the shallower models do. To do this, we compare the generalization accuracies of models at points during fine-tuning when they have equal in-distribution loss. Figure 7b shows that even after

[2]We only consider models deeper than the reference model since, in general, shallower models will never attain the perplexity of the reference model at the end of its pretraining. This assumption breaks down when considering the deepest models in each size class, but these are far deeper than the points at which depth seems to saturate performance on our compositional datasets so we do not extensively explore this regime.

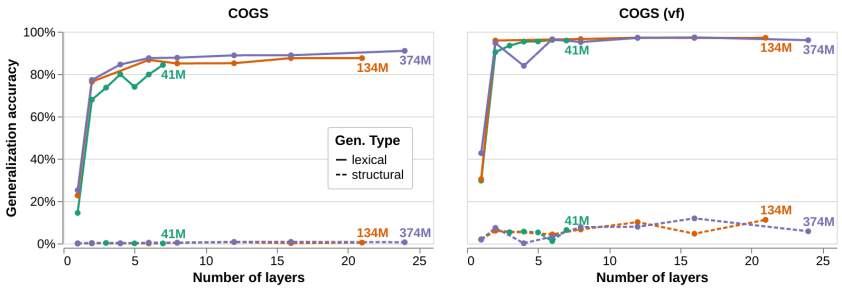
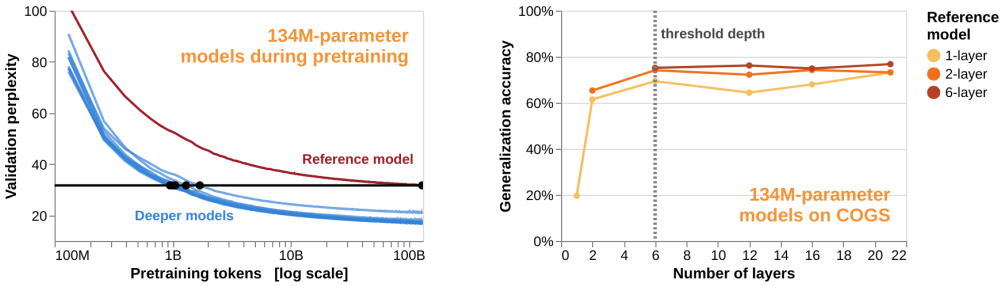


Figure 5: Increasing depth improves lexical generalization (solid lines) in both COGS and COGS-vf, but does not meaningfully improve structural generalization performance (dashed lines). Data shown is from a single run per condition.



(a) (b)

Figure 6: **(left)** To correct for the potential effect of deeper models' lower pretraining loss on their generalization accuracy, we pick a reference model depth (red) and use checkpoints (black) from deeper models (blue) which have equal validation perplexity as the reference model does at the end of its pretraining. We then fine-tune these 'pretraining-corrected' checkpoints on the compositional tasks. **(right)** Even when fine-tuning checkpoints with equal validation perplexity, deeper models still generalize better than shallower models do up through six layers. The figure shows generalization accuracies from 134M-parameter models on COGS (single run per condition).

adjusting for in-distribution performance, deeper models still achieve higher accuracies on the out-of-distribution generalization set than shallower models do.

4 Related Work

Compositionality Previous work has explored the degree to which neural models exhibit compositional behavior by training or fine-tuning models on compositional tasks such as simple command sequences (Lake and Baroni, 2018) or semantic parsing (Kim and Linzen, 2020; Keysers et al., 2020). Other work has explored methods to improve the compositional behavior of models, including through data augmentation (Qiu et al., 2022a), larger models (Qiu et al., 2022b), and architectural changes (Gordon et al., 2019; Csordás et al., 2021; Ontanon et al., 2022). Our work complements these approaches by exploring a specific architecture change: increasing depth without changing total model size.

Comparison to standard architectures We primarily focus on models that are shallower and wider than standard convention. Since d_{model} is fixed within each class this means that most of our models have increasingly small feed-forward ratios $d_{\text{model}}/d_{\text{ff}}$; moreover, since n_{layers}, d_{model}, and d_{ff} tend to increase in standard architectures as parameter count grows, this means that the disparities between our shallowest models and the conventional ones grows as the size class gets bigger. Exact parameter counts differ from the corresponding models in Raffel et al. (2019) and Kim and Linzen (2020) owing to differences in the size of the vocabulary/embedding layers and the fact that we use decoder-only models rather than encoder-decoder models, though the layer-internal hyperparameters of our base models are consistent with theirs. Qiu et al. (2022b) found that decoder-only models performed similarly to encoder-decoder models of comparable size; following Wang et al. (2022) we consider decoder-only models with half as many total layers as their encoder-decoder variants.

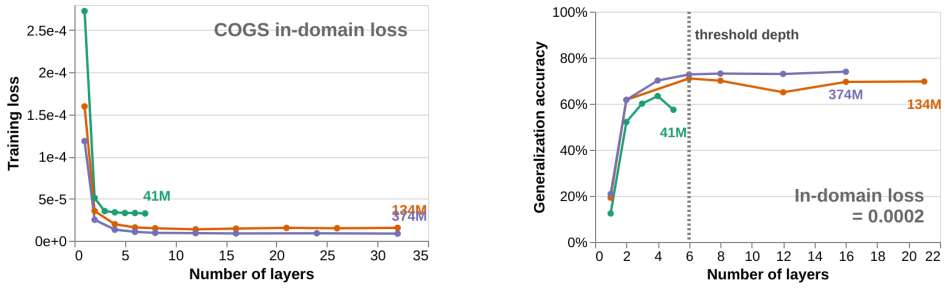


Figure 7: **(left)** Deeper models attain lower (better) in-domain loss values on compositional tasks. **(right)** Deeper models generalize better than shallower ones on COGS, even at points during fine-tuning when models have equal loss (0.0002) on the in-distribution portion of the dataset. Data shown is from a single run per condition.

Impacts of Depth Theoretical work has shown that the expressive capacity of neural networks in general (Raghu et al., 2017) and transformer models in particular (Merrill et al., 2021) grows exponentially in depth. Empirical work also points to the role of depth in model performance. In a more general setting, Tay et al. (2021) found that scaling by depth is generally more helpful than scaling by width on downstream tasks. For compositional generalization in particular, Mueller et al. (2022) found that reducing depth was more harmful than reducing with for pretrained encoder-decoder models. Murty et al. (2022) observed that deeper transformer encoders often have more tree-like representations and parsing accuracies on some compositional tasks. Tempering these positive results, Veit et al. (2016) noted that in models with residual connections, even very deep networks leveraged only shallow subnetworks of roughly constant depth. Brown et al. (2022) also concluded that wide, shallow transformer models can attain roughly-equal performance to deeper ones. Both sets of results, however, are confounded by a lack of control for total parameter count.

Controlling for model size There are various choices to be made when studying the impact of hyperparameter choices without affecting the net model size, i.e constructing size classes of models. Kaplan et al. (2020) covaried the number of layers n_{layers} with the contextual embedding dimension d_{model}, which they coupled to the attention-internal d_{attn} and feed-forward dimension at the standard ratio of $d_{\text{model}} = d_{\text{attn}} = d_{\text{ff}}/4$. Among models of an equal size, they concluded that performance increases are largely driven by increasing the total parameter count of models, and that within "reasonable limits" language modeling perplexity is only weakly dependent on shape (though Tay et al. 2021 concluded that the same was not true for performance on downstream tasks, but did so without controlling for the impact of size).

5 Conclusion

Compositional generalization is essential for interpreting novel sentences. What aspects of the transformer LM architecture contribute to an inductive bias favoring compositional generalization? In a controlled experiment that teases apart depth from total number of parameters, we find that deeper transformers show better compositional generalization (and better language modeling performance) independent of their total number of parameters, though in most cases the usefulness of adding layers decreases rapidly as models get deeper. Most of the benefits of depth come from having just a few layers, allowing comparatively shallow models to achieve levels of generalization accuracy on compositional tasks comparable to much deeper models, and to reach language modeling perplexity within a few percentage points of the best-in-class model. We also show the benefits of depth for compositional generalization are not merely a consequence of the fact that deeper models learn the in-distribution data or pretraining corpus better; rather, depth affects generalization over and above these other factors. Our results are robust across nearly an order of magnitude in model size (41M, 134M and 374M parameters).

6 Limitations & Future Work

Alternative approaches to controlling for total size Our approach to controlling for total parameter count necessitates making depth-width tradeoffs. An alternative approach would be to construct

Universal Transformers (Dehghani et al., 2018), where each model in a size class has a transformer layer with the same parameters repeated n_{layers} times. Such a weight-sharing approach would allow for deeper models to have arbitrarily-wide feed-forward networks, mitigating the impact of making models too narrow. While such weight sharing prevents models from performing different computation in different layers, such restriction may in fact be beneficial for compositional generalization where similar computations (e.g., combining two syntactic phrases to a larger phrase) may need to apply recursively at different scales.

Pretraining corpus effects We consider models pretrained on natural-language data. For our particular choice of compositional generalization experiments, the presence of lexical items in both the pretraining corpus and the generalization datasets represents a potential confounder of generalization performance which could be mitigated by modifying compositional datasets (Kim et al., 2022). More generally, the distribution of pretraining data affects the inductive biases conferred to LMs (Papadimitriou and Jurafsky, 2023). As a particular area of interest for future work, we point out the hypothesis that including source code in the pretraining corpus (OpenAI, 2023; Google et al., 2023) will improve compositional generalization.

Fine-tuning vs. in-context learning We use fine-tuning to adapt our pretrained models to the compositional tasks. Due to its computational cost and task-specificity, fine-tuning is less useful in practice than in-context learning as model size grows (Brown et al., 2020). Because in-context learning only becomes reliable at scales far larger than we are able to train, we did not explore the effect of depth on compositional generalization accuracy in in-context learning (Si et al., 2022); we point this out as an avenue for future research.

7 Ethics Statement

Throughout our experimental process, we sought to comply with best practices to mitigate any risks associated with LLM research. We use open-source datasets which are inspectable by third-parties for issues such as bias, and toxicity. We do not release any public checkpoints for the models we train, so there is no risk to misuse of any created artifacts, though we note that we derive our implemented models from existing publicly-available T5 models. We train models on English-only natural-language data, and fuller exploration should be done to explore how language impacts the results found here.

References

Jason Ross Brown, Yiren Zhao, Ilia Shumailov, and Robert D Mullins. 2022. Wide attention is the way forward for Transformers?

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33*, volume 33, page 1877–1901. Curran Associates, Inc.

Róbert Csordás, Kazuki Irie, and Juergen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 619–634, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2018. Universal Transformers. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, page 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jerry A Fodor and Zenon W Pylyshyn. 1988. Connectionism and cognitive architecture: a critical analysis. *Cognition*, 28(1-2):3–71.

Google, Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele

Catasta, Yong Cheng, Colin Cherry, Christopher A Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Li, Music, Wei Li, Yaguang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. 2023. PaLM 2 Technical Report. Technical report, Google.

Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. 2019. Permutation Equivariant Models for Compositional Generalization in Language. In *ICLR 2020 (OpenReview)*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training Compute-Optimal Large Language Models.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring Compositional Generalization: A Comprehensive Method on Realistic Data.

Najoung Kim and Tal Linzen. 2020. COGS: A Compositional Generalization Challenge Based on Semantic Interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 9087–9105, Online. Association for Computational Linguistics.

Najoung Kim, Tal Linzen, and Paul Smolensky. 2022. Uncontrolled lexical exposure leads to overestimation of compositional generalization in pretrained models.

Brenden Lake and Marco Baroni. 2018. Generalization without Systematicity: On the Compositional Skills of Sequence-to-Sequence Recurrent Networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2873–2882. PMLR.

William Merrill, Ashish Sabharwal, and Noah A Smith. 2021. Saturated Transformers are Constant-Depth Threshold Circuits. *Transactions of the Association for Computational Linguistics*, pages 843–856.

Richard Montague. 1970. Universal grammar. *Theoria*, 36(3):373–398.

Aaron Mueller, Robert Frank, Tal Linzen, Luheng Wang, and Sebastian Schuster. 2022. Coloring the Blank Slate: Pre-training Imparts a Hierarchical Inductive Bias to Sequence-to-sequence Models. In *Findings of the Association for Computational Linguistics: ACL 2022*, page 1352–1368, Dublin, Ireland. Association for Computational Linguistics.

Niklas Muennighoff, Alexander M Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling data-constrained language models.

Karl Mulligan, Robert Frank, and Tal Linzen. 2021. Structure Here, Bias There: Hierarchical Generalization by Jointly Learning Syntactic Transformations. In *Proceedings of the Society for Computation in Linguistics 2021*, pages 125–135.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. 2022. Characterizing Intrinsic Compositionality in Transformers with Tree Projections.

Santiago Ontanon, Joshua Ainslie, Zachary Fisher, and Vaclav Cvicek. 2022. Making transformers solve compositional tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3591–3607, Stroudsburg, PA, USA. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 Technical Report.

Isabel Papadimitriou and Dan Jurafsky. 2023. Pretrain on just structure: Understanding linguistic inductive biases using transfer learning.

Linlu Qiu, Peter Shaw, Panupong Pasupat, Paweł Krzysztof Nowak, Tal Linzen, Fei Sha, and Kristina Toutanova. 2022a. Improving Compositional Generalization with Latent Structure and Data Augmentation.

Linlu Qiu, Peter Shaw, Panupong Pasupat, Tianze Shi, Jonathan Herzig, Emily Pitler, Fei Sha, and Kristina Toutanova. 2022b. Evaluating the Impact of Model

Scale for Compositional Generalization in Semantic Parsing.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research: JMLR*, 21(2020):1–67.

Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. 2017. On the Expressive Power of Deep Neural Networks. In *International Conference on Machine Learning*, pages 2847–2854. PMLR.

Adam Roberts, Hyung Won Chung, Anselm Levskaya, Gaurav Mishra, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, Curtis Hawthorne, Aitor Lewkowycz, Alex Salcianu, Marc van Zee, Jacob Austin, Sebastian Goodman, Livio Baldini Soares, Haitang Hu, Sasha Tsvyashchenko, Aakanksha Chowdhery, Jasmijn Bastings, Jannis Bulian, Xavier Garcia, Jianmo Ni, Andrew Chen, Kathleen Kenealy, Jonathan H. Clark, Stephan Lee, Dan Garrette, James Lee-Thorp, Colin Raffel, Noam Shazeer, Marvin Ritter, Maarten Bosma, Alexandre Passos, Jeremy Maitin-Shepard, Noah Fiedel, Mark Omernick, Brennan Saeta, Ryan Sepassi, Alexander Spiridonov, Joshua Newlan, and Andrea Gesmundo. 2022. Scaling up models and data with `t5x` and `seqio`.

Olivier Roy and Martin Vetterli. 2007. The effective rank: A measure of effective dimensionality. In *2007 15th European Signal Processing Conference*, pages 606–610.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. 2022. Prompting GPT-3 to be reliable. In *The Eleventh International Conference on Learning Representations*.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. 2021. Scale efficiently: Insights from pre-training and fine-tuning Transformers. In *International Conference on Learning Representations*.

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 *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Andreas Veit, Michael Wilber, and Serge Belongie. 2016. Residual networks behave like ensembles of relatively shallow networks.

Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. 2022. What language model architecture and pretraining objective work best for zero-shot generalization?

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the thirteenth national conference on Artificial intelligence - Volume 2*, AAAI'96, pages 1050–1055. AAAI Press.

A Full table of results

Table 3 displays pretraining and compositional generalization accuracy on all model sizes and tasks.

B Annotated transformer layer

Figure 8 shows the schematic for a single transformer layer. The layers input enters on the left and passes through the various model components (grey boxes), being combined with the residual connections before exiting right to subsequent layers. Blue boxes show the dimensionality of the vectors after transformation; we are primarily concerned with the size of the embedding vectors d_{model} and the internal dimension of the feed-forward block d_{ff}. The size of the vectors internal to the attention mechanism, d_{attn}, is not shown here but is usually set to be equal with d_{model}; we follow this convention here. Non-learned operations like addition, layer normalization, and the feed-forward network's nonlinearity are shown in grey circles.

C Compute Analysis

To better understand the implications of the diminishing marginal utility of depth on perplexity and generalization performance, we additionally analyze how depth impacts the compute performance of models by measuring the latency of models (measured in seconds per step) during pretraining. Figure 9 shows that depth strongly influences latency when controlling for model size: 32-layer 374M-parameter models are twice as slow as the 4-layer 374M-parameter models when running on the same hardware.

We note two implications of this finding relevant to the practical use of deep models in the context of our main results on perplexity and accuracy:

1. Since models with relatively few layers can attain withing 1% of the performance (i.e., loss for language modeling or generalization accuracy for compositional generalization tasks) of the better, deeper models when trained on equal volumes of data, shallow models

size	n_{layers}	C4 val. PPL (↓)	COGS (↑)	COGS-vf (↑)	GeoQuery Standard (↑)	English Passivization (↑)
41M	1	45.7	12.4	25.7	68.2	0.00
	2	31.1	58.2	78.3	76.4	9.88
	3	29.3	63.1	80.8	**79.6**	26.2
	4	28.8	68.5	82.5	78.6	28.0
	5	**28.8**	63.4	82.5	76.8	**89.9**
	6	29.1	68.4	82.6	77.5	74.1
	7	29.6	**72.3**	**83.0**	77.1	78.3
134M	1	33.6	19.4	26.3	72.5	0.00
	2	22.3	65.5	83.0	81.4	29.9
	4	19.4	71.1	83.6	78.2	59.3
	6	18.7	74.3	83.2	80.0	49.4
	8	18.3	72.9	83.7	73.6	91.9
	12	**18.1**	73.0	84.7	**82.9**	87.1
	16	18.2	75.0	83.8	81.1	93.2
	21	18.3	75.1	**84.8**	80.0	88.1
	26	18.6	75.4	84.1	82.1	**98.4**
	32	19.2	**75.7**	84.0	78.9	94.8
374M	1	28.4	21.5	36.8	72.9	0.00
	2	18.6	66.2	82.2	80.7	13.6
	4	15.9	72.4	71.9	80.0	89.8
	6	15.2	75.1	83.1	78.2	18.8
	8	14.9	75.2	82.6	80.7	84.3
	12	14.6	76.3	84.3	80.0	81.0
	16	14.5	76.3	**85.1**	81.1	87.2
	24	**14.4**	78.0	83.1	83.2	89.6
	32	14.7	**78.8**	79.7	**84.6**	**90.2**

Table 3: Validation perplexity (↓, lower is better) on C4 after pretraining & generalization accuracy (%; ↑, higher is better) on compositional datasets after 10 k steps of fine-tuning. Bold values indicate best-in-size-class performance. Data is from a single run per condition.

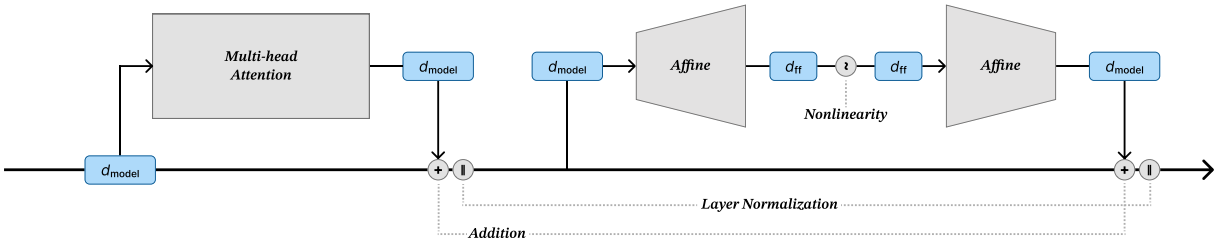


Figure 8: Diagram of a single transformer layer, annotated with the dimensions (blue) of each vector. Information is passed from left to right, through each component (grey box), and added back to the residual embeddings before normalization.

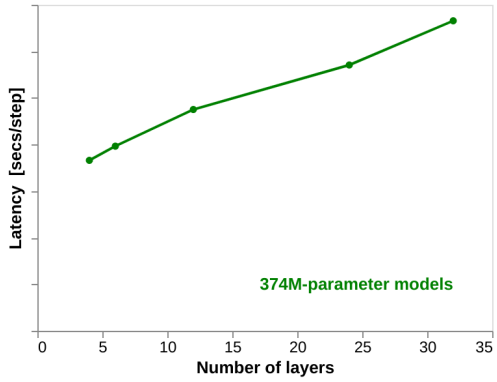


Figure 9: Deeper models are slower than shallower models when controlling for total parameter count. We report relative latency [seconds per step] for 374M-parameter models, showing a strongly-linear relationship between depth and latency. Similar relative trends are observed for other model sizes classes.

can be used at inference time at substantially lower compute costs without sacrificing performance.

2. Since shallow models have lower latency during training as well as during inference, one can either (a) train a shallower model in much less time than a deeper model on equal volumes of data, or (b) train a shallower model on much more data than a deeper model can be trained on given a fixed compute budget (e.g., access to a particular GPU platform for a given amount of time). Depending on the depths, sizes, volumes of data, and training times involved, this can result in shallower models attaining better performance than deeper models can.

D Feed-forward rank analysis

To investigate the role that the feed-forward block plays in the poor performance of models with extreme $d_{\text{model}}/d_{\text{ff}}$ ratios, we conduct rank anal-

ysis on the two transformations $\mathbb{R}^{d_{\text{model}}} \to \mathbb{R}^{d_{\text{ff}}}$ and $\mathbb{R}^{d_{\text{ff}}} \to \mathbb{R}^{d_{\text{model}}}$ which make up the feed-forward block. Our first approach is to conduct singular-value decomposition on each transform. For a given affine transform T, we compute the ordered singular values $\{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ where $k = \min(d_{\text{model}}, d_{\text{ff}})$ is the rank of T and $\sigma_i \geq \sigma_{i+1}$. We then normalize each singular value by dividing by the ℓ_1 norm of $\{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ to calculate how much of the T's image is accounted for by the best i-rank approximation of T for $i \leq k$. We note that as models get deeper (and consequently, d_{ff} and gets smaller and the feed-forward ratio $d_{\text{model}}/d_{\text{ff}}$ gets larger), the two transforms in the feed-forward block become increasingly skewed away from making full use of their available ranks (Figure 10).

We also measure the *effective rank* of each transform, defined by Roy and Vetterli (2007) a real-valued extension of rank to measure the effective dimensionality of transforms which are close to being rank-deficient:

$$\text{erank}(T) = \exp\left(-\sum \frac{\sigma_i}{\|\sigma\|_1} \log\left(\frac{\sigma_i}{\|\sigma\|_1}\right)\right).$$

We similarly note that the effective rank of the feed-forward transforms decreases as models get deeper and d_{ff} gets smaller relative to fixed d_{model}, suggesting that our deeper models are increasingly rank-deficient (Figure 11).

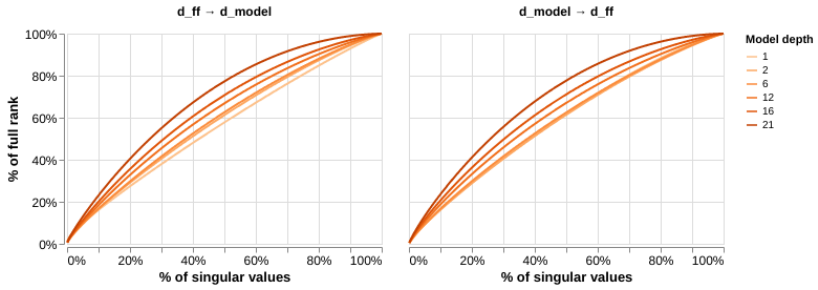


Figure 10: As models get deeper and d_{ff} gets smaller, the input **(left)** and output **(right)** projections in the feed-forward block become increasingly close to rank-deficient transforms. A graph of $y = x$ here would indicate that models spread their rank equally across all singular values.

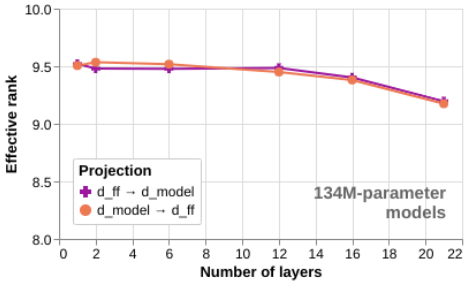


Figure 11: The effective rank of each feed-forward projection, averaged over all layers, decreases as models get deeper and d_{ff} gets smaller in proportion to d_{model}. A small amount of vertical jitter has been added to help distinguish lines.