The Heuristic Core: Understanding Subnetwork Generalization in Pretrained Language Models

Adithya Bhaskar **Dan Friedman** **Danqi Chen**
Princeton Language and Intelligence (PLI), Princeton University
adithyab@princeton.edu {dfriedman, danqic}@cs.princeton.edu

Abstract

Prior work has found that pretrained language models (LMs) fine-tuned with different random seeds can achieve similar in-domain performance but generalize differently on tests of syntactic generalization. In this work, we show that, even within a single model, we can find multiple subnetworks that perform similarly in-domain, but generalize vastly differently. To better understand these phenomena, we investigate if they can be understood in terms of "competing subnetworks": the model initially represents a variety of distinct algorithms, corresponding to different subnetworks, and generalization occurs when it ultimately converges to one. This explanation has been used to account for generalization in simple algorithmic tasks ("grokking"). Instead of finding competing subnetworks, we find that all subnetworks—whether they generalize or not—share a set of attention heads, which we refer to as the *heuristic core*. Further analysis suggests that these attention heads emerge early in training and compute shallow, non-generalizing features. The model learns to generalize by incorporating additional attention heads, which depend on the outputs of the "heuristic" heads to compute higher-level features. Overall, our results offer a more detailed picture of the mechanisms for syntactic generalization in pretrained LMs.[1]

1 Introduction

A central question in machine learning is to understand how models generalize from data that supports different possible solutions. McCoy et al. (2020) investigated this question in the context of pretrained language models (LMs) like BERT (Devlin et al., 2019) trained on natural language inference (NLI; Williams et al., 2018), showing that models trained with different random seeds perform very similarly on in-domain (ID) evaluation sets but generalize very differently to adversarial out-of-domain (OOD) evaluation sets. However, we have a limited understanding of how any individual model learns to generalize.

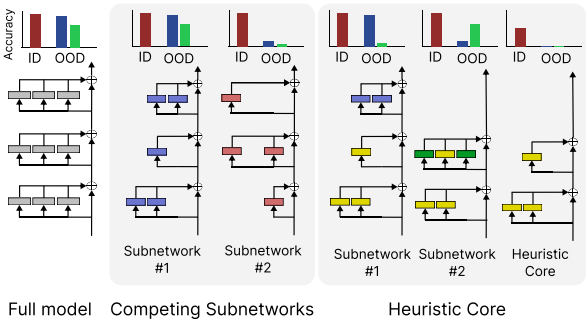


Figure 1: We find different subnetworks in a pretrained LM that achieve similar in-domain performance but generalize differently. Prior work has explained similar generalization phenomena in synthetic tasks in terms of distinct subnetworks that compete during training. We instead find evidence of a *heuristic core*: a set of attention heads that appear in all generalizing subnetworks but, on their own, do not generalize.

In this work, we use subnetwork analysis to better understand how pretrained LMs generalize on NLP tasks. Specifically, we use structured pruning (Wang et al., 2020; Xia et al., 2022) to isolate various subnetworks—subsets of attention heads and MLP layers—that approximate the behavior of the full model. Our main finding is, that even within a single model, there exist multiple subnetworks that all match the model's performance closely on in-domain evaluation sets, but exhibit vastly different generalization. This result has meaningful consequences for practitioners—for example, it underscores the importance of OOD evaluation of pruning methods. It also raises several questions about the mechanisms underlying generalization, which we proceed to investigate.

One possible explanation for our results is that the model initially consists of a variety of disjoint

[1]We release our code publicly at https://github.com/princeton-nlp/Heuristic-Core.

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 14351–14368

subnetworks, corresponding to distinct possible solutions for the task; the model ultimately generalizes if it converges to a subnetwork that generalizes. This model of generalization has been used to explain the "grokking" phenomenon on toy algorithmic tasks, whereby a neural network initially overfits a training set—but, after continued training, suddenly undergoes a phase shift to perfect generalization (Barak et al., 2022; Nanda et al., 2023; Varma et al., 2023). We refer to this as the *Competing Subnetworks* explanation, following Merrill et al. (2023). Our setting bears a similarity to grokking, originally documented by Tu et al. (2020): models converge early on in-domain evaluations and only start to generalize after training for additional epochs. Therefore, we test if generalization in this case also arises from competition between disjoint subnetworks.

First, we check if these behaviorally different subnetworks consist of disjoint subsets of model components (attention heads). To the contrary, we find a small set of nine attention heads that occur in *all* subnetworks—even the ones that do not generalize at all. Furthermore, we find that this same set of attention heads also appears when we prune earlier checkpoints of the model before it starts to generalize. In the grokking setting, generalization is marked by a decrease in *effective size*—defined as the size of the smallest subnetwork that matches the full network's performance on the generalization sets—as the model switches from a dense, memorizing subnetwork to a sparse, generalizing subnetwork. In contrast, we find that generalization in our case is accompanied by a sharp *increase* in effective size. Together with the set of common attention heads, our results suggest that, instead of selecting between competing subnetworks, the model initially learns a "core" of attention heads that implement simple heuristics. The model ultimately generalizes by learning additional attention heads that interact with these heads.

We refer to this core set of attention heads as the *heuristic core*, and conduct further analysis to better understand what roles they play in the model. We find that these attention heads are associated with entailment heuristics—namely, attending to words that are repeated across sentences—supporting our characterization of these components as heuristic core. On the other hand, ablating these attention heads from the original model leads to a dramatic *decrease* in performance on the generalization set. This result further supports the idea that generalization arises from additional components building off simple features extracted by heuristic components. At intermediate sparsity[2] levels, subnetworks contain different "counter-heuristic" components, leading to different degrees of partial generalization to different subcases.

These results have important practical implications. For instance, they suggest that we cannot make the model more robust by ablating the "heuristic" components. More broadly, our experiments paint a more detailed picture of the mechanisms underlying generalization in natural language understanding tasks. In doing so, they highlight interesting phenomena and open new avenues to future research into language models' internals.

2 Problem Setup

We focus on two sentence-pair classification tasks. MNLI (Williams et al., 2018) is a natural language inference (NLI) dataset, which involves predicting whether a *premise* sentence logically entails a *hypothesis* sentence. QQP (Iyer et al., 2017) is a paraphrase identification dataset. Prior work has noted that these datasets admit "heuristics": solutions that achieve high in-domain accuracy but do not generalize well. In particular, McCoy et al. (2019) introduced the HANS dataset, which tests whether NLI models use specific heuristics, such as lexical overlap (predicting that sentences entail each other if they share many words); see Appendix Table 1 for examples and the list of abbreviations used for HANS subcases. Each subcase of HANS has 1000 validation examples. Similarly, the QQP-PAWS dataset (Zhang et al., 2019) tests whether paraphrase identification models are susceptible to a word-overlap heuristic (Appendix Table 2). PAWS-QQP has $12,663$ examples in all, of which $8,696$ bear the verdict "not equivalent". We work with the latter subset in this paper.

Models. We mainly focus on BERT (Devlin et al., 2019) throughout the paper, but our findings also generalize to RoBERTa (Liu et al., 2019) and GPT-2 (Radford et al., 2019) (Section 6). We fine-tune BERT models on MNLI and QQP (see Appendix A for more training details). Our results reproduce the observations of Tu et al. (2020): in-domain accuracy saturates early, and out-of-domain accuracy increases much later. We focus on the four subcases in Table 1 for HANS, as the model gener-

[2] In this paper, sparsity refers to the fraction (or percentage) of attention heads and MLPs that have been pruned away.

Subcase	Name	Abbreviation	Example
Lexical Overlap (LO)			
Subject-Object Swap	`lexical_overlap_ln_subject-object_swap`	SO-Swap	The doctor advised the president. $\not\rightarrow$ The president advised the doctor.
Preposition	`lexical_overlap_ln_preposition`	Prep	The tourist by the manager saw the artists. $\not\rightarrow$The artists saw the manager.
Constituent (C)			
Embedded under if	`constituent_cn_embedded_under_if`	Embed-If	If the artist slept, the actor ran. $\not\rightarrow$ The actor ran.
Embedded under verb	`constituent_cn_embedded_under_verb`	Embed-Verb	The lawyers believed that the tourists. shouted $\not\rightarrow$ The tourists shouted.

Table 1: Example cases and adversarial subcases from HANS (McCoy et al., 2020). We focus mostly on these subcases in the main text as the model generalizes best to them. We use abbreviated names for ease of discussion.

Question #1	Question #2	Equivalent?
Is a contagious yawn also fake?	Is a fake yawn also contagious?	✗
How are noble gases stable?	How are stable gases noble?	✗
How is Perth better than Melbourne?	How is Melbourne better than Perth?	✗

Table 2: Example questions from the PAWS-QQP dataset (Zhang et al., 2019) dataset. We only work with questions whose label is "not equivalent", as the others are heuristic-friendly.

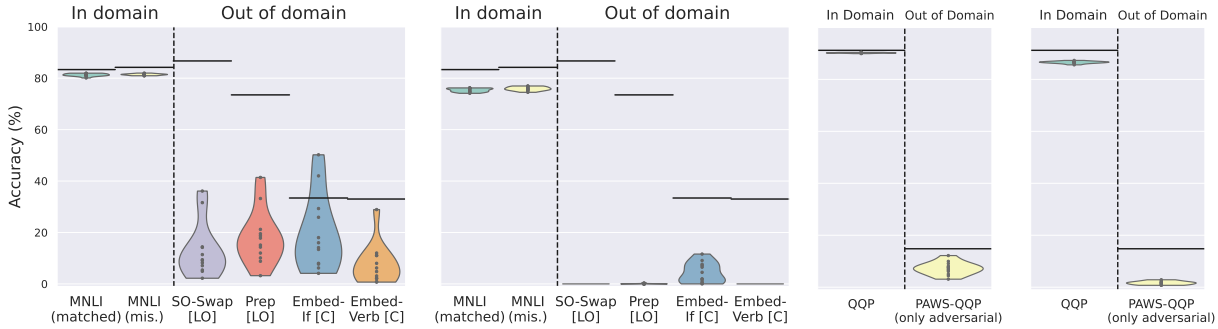
alizes well on them. Other subcases are discussed in Appendix C. The effect is smaller, but still consistent, on PAWS-QQP (Table 2).

Finding subnetworks via pruning. For a Transformer model, we define a subnetwork as a subset of the attention heads and MLP (feed-forward) layers. An important motif in our analysis will be the search for subnetworks that preserve model performance. We choose structured pruning towards this end. While circuit-finding methods (Conmy et al., 2023) would also be useful, pruning is faster and allows injecting randomness more readily. We use structured rather than unstructured pruning for two reasons. First, unstructured pruning can be expressive enough to find subnetworks with completely different behavior from the original model (Wen et al., 2023). Second, structured pruning lets us compare subnetworks in terms of semantically meaningful components: attention heads.

We prune via optimization of a binary *mask*, where 0 and 1 indicate dropping and retention, respectively. Our runs prune attention heads and entire MLP layers, each corresponding to one bit in this mask. We use *mean ablation* per recommendations from work in Mechanistic Interpretability (Zhang and Nanda, 2024)—pruned layers' (or heads') activations are replaced by the mean activation over the training data. Following CoFi Pruning (Xia et al., 2022), we relax binary masks to floats in $[0, 1]$ and then perform gradient descent to optimize them. The objective optimized is the KL loss between the predictions of the full model and those of the pruned subnetwork. Finally, the mask entries are discretized to $\{0, 1\}$ based on a threshold. L0 regularization is used to enforce a target sparsity. We adopt Louizos et al. (2018)'s recipe to model the masks and furnish the details in Appendix B. More details and hyperparameters, along with a more detailed discussion of the method, are provided in Appendix A. We freeze the model after fine-tuning and only optimize the pruning masks to preserve faithfulness to the full model.

3 Comparing Subnetwork Generalization

We start by examining how different subnetworks generalize on MNLI/HANS and PAWS/QQP. Specifically, we compare the generalization behavior of subnetworks with the same sparsity level, but pruned with different seeds; and we compare subnetworks pruned with different sparsity levels. We prune the BERT models fine-tuned on MNLI and QQP with 12 random seeds for different target sparsities (more details in Appendix A). The resulting subnetworks are evaluated on the ID (MNLI and QQP validation sets) and OOD (HANS subcases, and PAWS-QQP's adversarial subset) evaluation sets. In our runs, we observe that the MLP layers are never pruned (although they are allowed to be).



(a) MNLI, 50% (b) MNLI, 70% (c) QQP, 30% (d) QQP, 60%

Figure 2: Pruning a BERT model with different random seeds results in subnetworks that perform similarly in-domain but generalize differently. The dots refer to the accuracy of the pruned subnetworks, while solid lines indicate full model performance. MNLI/HANS: At 50% sparsity, the subnetworks perform within 3% of the model on MNLI but show varying generalization. At 70% sparsity, the subnetworks behave as pure heuristics despite respectable MNLI accuracy. The trend also holds for QQP/PAWS, with sparsities of 30% and 60%. Figure 12 in Appendix C shows the plot for all subcases of HANS.

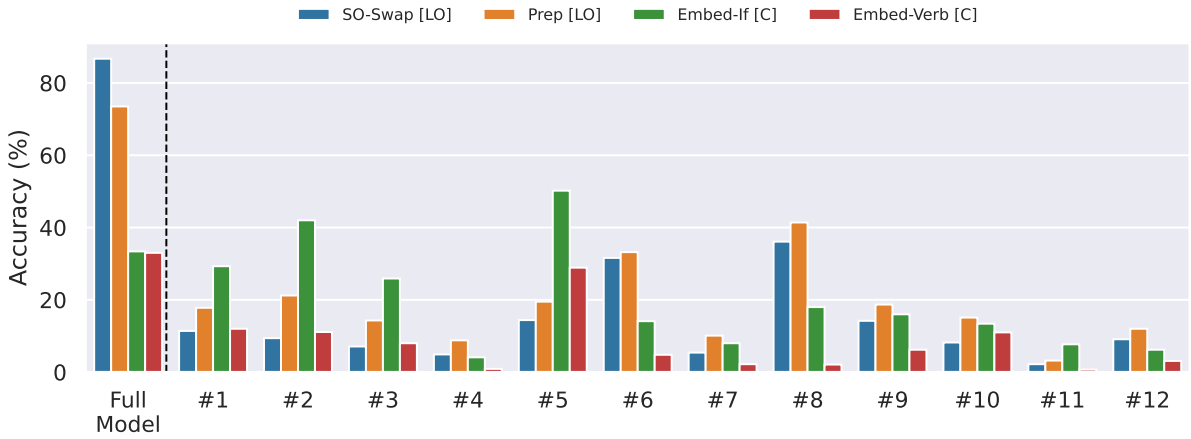


Figure 3: Different subnetworks at 50% sparsity (found by pruning with different random seeds) generalize partially to different subcases of HANS. Subnetwork #5 generalizes to the *Constituent* subcases, whereas #8 generalizes to the *Lexical Overlap* subcases. Subnetwork #2 does well on the *Embed-If* subcase of *Constituent*, but not *Embed-Verb*.

Different subnetworks generalize differently. Figures 2a and 2c plot the results for a target sparsity of 50% and 30% in the case of MNLI and QQP, respectively. The pruned models perform close to each other in-domain, and are all within 2-3% of the full model's accuracy on both the MNLI and QQP validation splits. However, their accuracies on the out-of-domain evaluation splits varies widely. In the most extreme case, the accuracies on the *Embed-If (Constituent)* subcase of HANS vary from just 4.1% to 50.2%. Moreover, different subnetworks generalize to different subcases, as Figure 3 illustrates. Subnetwork #8 achieves 41.4% at *SO-Swap (Lexical Overlap)* but only 18.0% at *Embed-If (Constituent)*. Subnetwork #5, on the other hand, achieves only 19.5% at the former but 50.2% at the latter. Thus, pruning different parts of the model seems to sacrifice generalization on the OOD subcases to varying degrees. The results in the case of QQP and PAWS-QQP are similar.

Sparser subnetworks generalize worse. We also observe that sparser subnetworks consistently generalize worse. First, we prune the model to a higher sparsity (70% for MNLI and 60% for QQP) with 12 random seeds. As Figures 2b and 2d show, virtually *all* subnetworks at this sparsity show a complete lack of generalization on almost every OOD subcase. The ID accuracy stays at a respectable 74-76%, indicating that a large portion of model performance is explained by these sparse subnetworks. In Figure 4, we plot the generalization accuracy of subnetworks pruned at different sparsity levels,

(a) MNLI/HANS (b) QQP/PAWS-QQP

Figure 4: The OOD accuracy decreases fairly smoothly with sparsity (3 seeds). The drop in ID accuracy is slow and has low variance. We find no subnetworks sparser than 30% generalizing as well as the full model on either dataset.

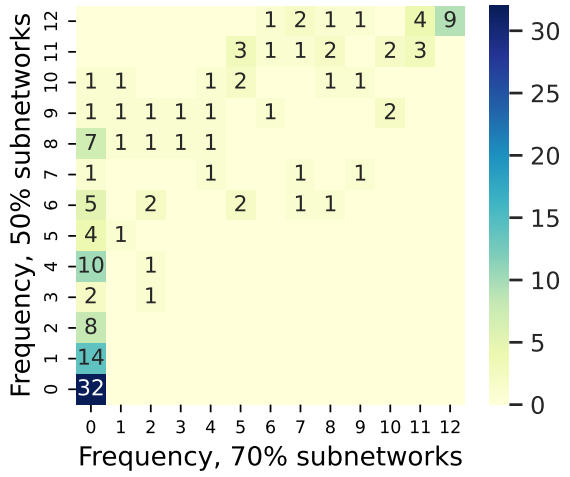


Figure 5: This heatmap quantifies the frequencies of attention heads in the 50% and 70% sparsity subnetworks (MNLI). Entry (i, j) corresponds to the number of heads appearing in i/12 50% subnetworks and j/12 70% subnetworks. In particular, we note that *nine* attention heads appear in all of the subnetworks.

for 3 random seeds. The generalization accuracy generally drops steadily at higher sparsity levels and is devoid of any drastic phase transitions.

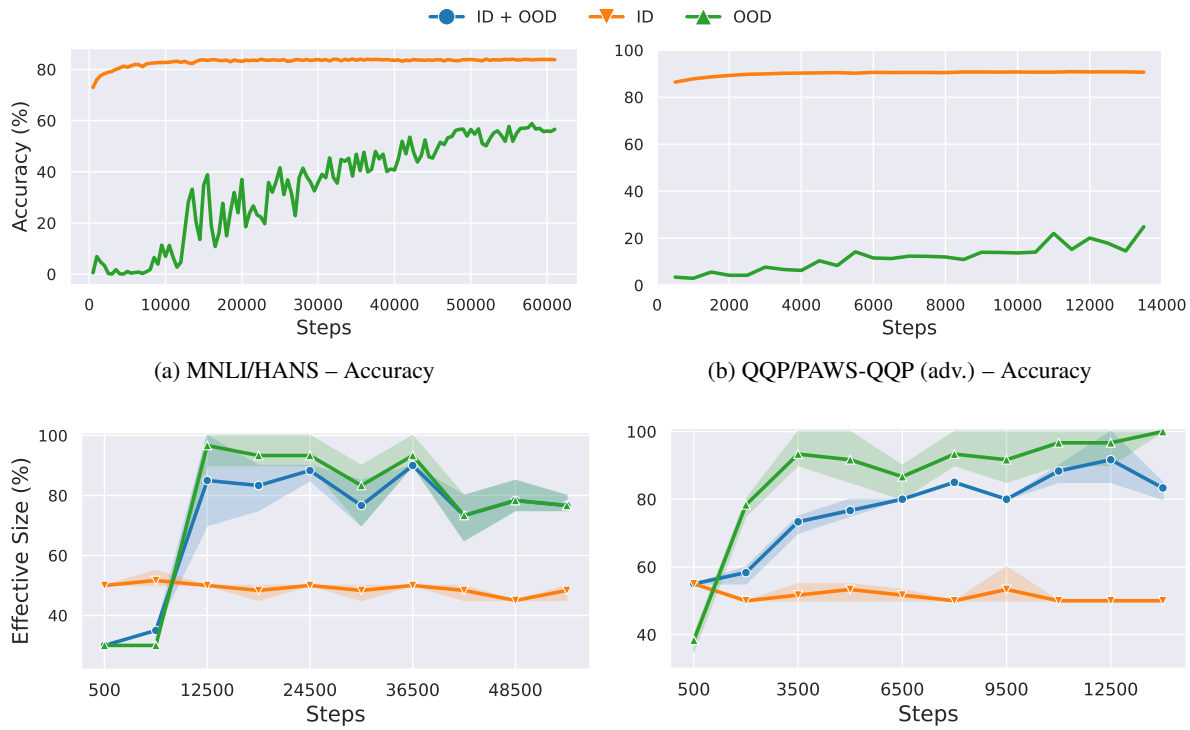
4 Competing Subnetworks Hypothesis

In the previous section, we observed that different subnetworks perform similarly to the full model in-domain but generalize differently. Moreover, no sparse subnetwork generalizes as well as the full model. We now seek to understand how these subnetworks emerge during training and interact to give rise to generalization.

Specifically, we explore if our findings can be explained in terms of *competing subnetworks*: the model initially represents multiple solutions as different subnetworks, and it ultimately converges to one. Whether it generalizes is dependent on which of the subnetworks it converges to. This hypothesis has been found to explain the curious phenomenon of grokking (Merrill et al., 2023). Additionally, as Tu et al. (2020) note, the ID performance of our BERT models saturates early in training (Figure 6) but the OOD accuracy starts to rise much later. Motivated by the similarity to grokking, we test if generalization in our case also arises from competition between disjoint subnetworks.

We consider the Competing Subnetwork Hypothesis to consist of two main predictions, mirroring Merrill et al. (2023). First, the model can be composed into subnetworks that are *disjoint*, representing distinct algorithms consistent with the training data. Second, a rise in generalization is accompanied by a reduction in the model's *effective size*—the smallest subnetwork whose accuracy matches the model's within a given threshold (e.g., 3%) on a target dataset—which occurs when the model switches from a dense, non-generalizing subnetwork to a sparse, generalizing subnetwork.

All subnetworks share a common set of components. We start by measuring how many model components are shared between different subnetworks. In particular, we compute the frequencies of each attention head in the 50% (partially generalizing) subnetworks and in the 70% (non-generalizing) subnetworks of the MNLI model. These frequencies are visualized in Figure 5 (a corresponding plot for QQP is presented in Figure 11 of Appendix C). The Spearman's Rho correlation between the two frequencies if 0.82 (p-value = $1.6 \cdot 10^{-36}$), i.e., very strong agreement. (The corresponding value for QQP is 0.74, corresponding



(a) MNLI/HANS – Accuracy

(b) QQP/PAWS-QQP (adv.) – Accuracy

(c) MNLI/HANS – Effective size

(d) QQP/PAWS-QQP (adv.) – Effective size

Figure 6: The ID and OOD accuracies, and effective size of the model during fine-tuning, computed by pruning with 3 seeds. Generalization is associated with a steep *increase* in effective size for OOD/mixed datasets. The effective size for ID datasets remains constant, showing that these extra effective parameters go towards generalization.

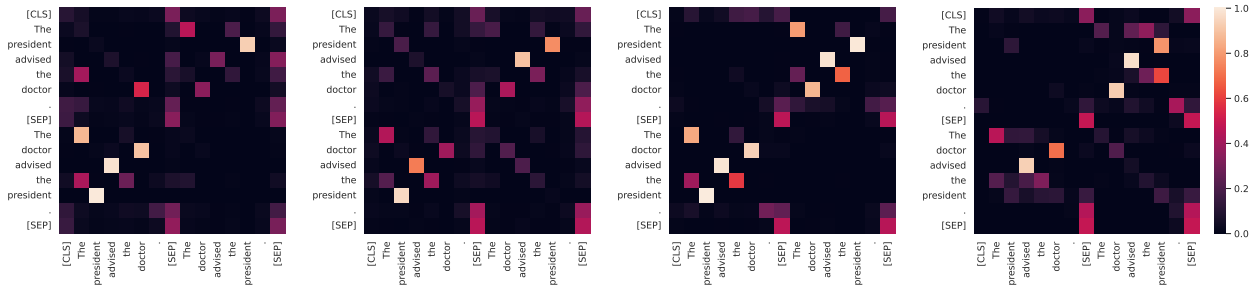
to strong agreement; see Figure 14 in Appendix C.) In particular, we observe that **nine attention heads** appear in *all* subnetworks, even the sparsest ones—that is, the subnetworks that behave most consistently with the heuristics. In other words, instead of finding disjoint subnetworks, we find a high degree of overlap (the expected intersection of 12 random 70% sparsity subnetworks is $7.7 \cdot 10^{-5}$ heads) between the subnetworks that partially generalize and the subnetworks that do not generalize at all.

Effective size increases as the model generalizes. Next, we investigate how these subnetworks emerge over training. We conduct a similar analysis to Merrill et al. (2023) and plot the *effective size* of the model, defined as the size of the smallest subnetwork that is *faithful* to the model on a particular evaluation dataset D. We say that a subnetwork is *faithful* to a model on dataset D if their accuracies are within 3%. We report faithfulness using different choices of D—either the in-domain validation set, the out-of-domain validation sets, or the equally weighted combination of the two (mixed data). We prune each checkpoint to sparsities of multiples of 5% and pick the highest one that yields a faithful subnetwork. The reported accuracies at each sparsity are averages over three seeds.

The results are in Figure 6. The effective size needed to approximate the model on the in-domain evaluation remains relatively flat over the course of training. However, on out-of-domain and mixed data, generalization is accompanied by an *increase* in effective size. We also examine the set of attention heads that appear in the MNLI subnetworks at the checkpoint immediately before the model starts to generalize. We find that the same nine attention heads that appear in all subnetworks at the end of training appear in these subnetworks as well. This suggests that, rather than switching between competing subnetworks, the model first learns a set of attention heads that compute simple, non-generalizing features, and then generalizes by incorporating more components into this core.

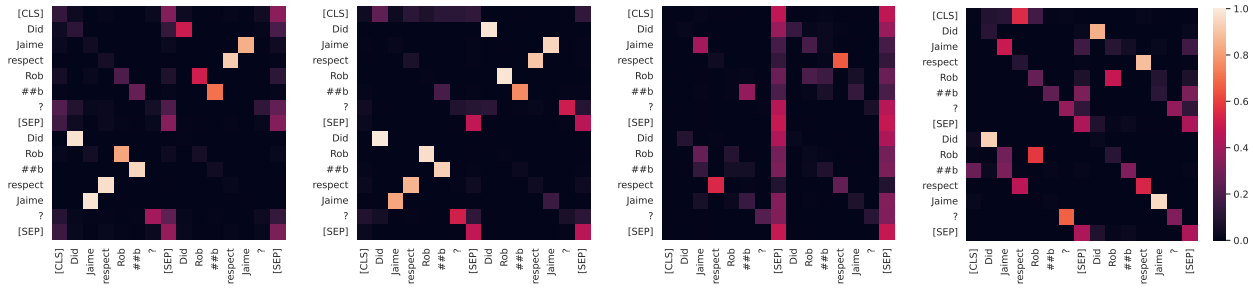
5 Understanding the Heuristic Core

Instead of finding evidence for distinct, competing subnetworks, we have found that all subnetworks share a set of common components. In particular, nine attention heads appear in all MNLI subnetworks, including subnetworks that do not generalize at all. We find the same nine attention heads



(a) Layer 2, Head 5 (b) Layer 3, Head 7 (c) Layer 4, Head 0 (d) Layer 8, Head 11

Figure 7: Many attention heads common to the MNLI subnetworks attend to tokens co-occurring in the premise and context. Since the order of, *e.g.*, "doctor" and "president" is reversed, the heads seem to identify lexical overlap.



(a) Layer 2, Head 5 (b) Layer 4, Head 0 (c) Layer 8, Head 11 (d) Layer 9, Head 1

Figure 8: Many attention heads in the QQP heuristic core also focus their attention between tokens repeated across the premise and context. Interestingly, there is a significant overlap between these heads and those for MNLI.

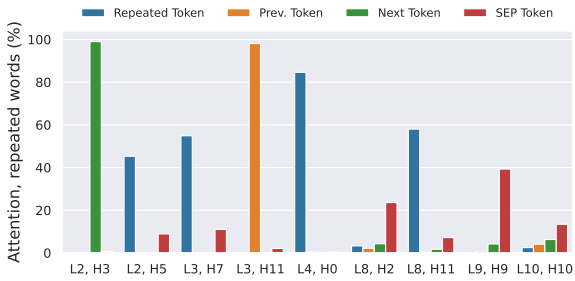


Figure 9: Of the 9 heads occurring in all MNLI subnetworks, several assign their attention weight between words repeated across the premise and hypothesis. Others either attend to the previous, next, or separator token. Here, "Lx, Hy" stands for "Head y of Layer x".

in subnetworks when we prune checkpoints early in training, before the model generalizes. We refer to these nine attention heads as the *heuristic core* of the model. In this section, we inspect these attention heads in more detail, to understand what features they compute, and how they interact with the rest of the model to enable generalization.

How do the heuristic core attention heads behave? First, we look at the attention patterns exhibited by these attention heads. We inspect example attention patterns and observe that most heads exhibit a simple attention pattern, and we quantify these in Figure 9. Four out of nine heads (MNLI) attend to tokens that repeat across the premise and hypothesis, suggesting that they extract token overlap features. Figure 7 shows example attention patterns. Notably, we find that eight attention heads also appear in all QQP subnetworks. Figure 8 presents an example indicating that these heads play a similar role for QQP, attending between repeated words.

Other low-layer heads attend to the previous or subsequent token. The heads in the higher layers are more difficult to characterize but often attend to the special separator token, perhaps aggregating information from across the sequence. Overall, these findings are consistent with the notion that these attention heads calculate simple, shallow features.

How do the heuristic core attention heads interact with the rest of the model? Next, we ablate each heuristic core attention head from the full model and observe how the in-domain and out-of-domain accuracies change, in Table 3. None of the heads' ablation affects in-domain accuracy greatly.

Ablated Head	Accuracy (%) ↑				
	MNLI (matched)	SO-Swap [LO]	Prep [LO]	Embed-If [C]	Embed-Verb [C]
None	83.3	86.7	73.5	33.0	33.4
Avg. (Non-HC)	$83.2_{\downarrow 0.1}$	$86.2_{\downarrow}0.5$	$73.1_{\downarrow 0.4}$	$33.5_{\uparrow 0.5}$	$34.1_{\uparrow 0.7}$
Layer 2, Head 3	$83.2_{\downarrow 0.1}$	$79.3_{\downarrow 7.4}$	$72.3_{\downarrow 1.2}$	$28.8_{\downarrow 4.2}$	$28.9_{\downarrow 4.5}$
Layer 2, Head 5	$83.4_{\uparrow 0.1}$	$70.3_{\downarrow 16.4}$	$58.6_{\downarrow 14.9}$	$28.5_{\downarrow 4.5}$	$23.6_{\downarrow 9.8}$
Layer 3, Head 7	$83.5_{\uparrow 0.2}$	$68.6_{\downarrow 18.1}$	$60.2_{\downarrow 13.3}$	$28.2_{\downarrow 4.8}$	$23.0_{\downarrow 10.4}$
Layer 3, Head 11	$83.5_{\uparrow 0.2}$	$81.8_{\downarrow 4.9}$	$70.1_{\downarrow 3.4}$	$31.0_{\downarrow 2.0}$	$32.9_{\downarrow 0.5}$
Layer 4, Head 0	$82.9_{\downarrow 0.4}$	$45.4_{\downarrow 41.3}$	$41.8_{\downarrow 31.7}$	$21.2_{\downarrow 11.8}$	$12.7_{\downarrow 20.7}$
Layer 8, Head 2	$83.7_{\uparrow 0.4}$	$84.5_{\downarrow 2.2}$	$69.5_{\downarrow 4.0}$	$31.6_{\downarrow 1.4}$	$27.5_{\downarrow 5.9}$
Layer 8, Head 11	$81.0_{\downarrow 2.3}$	$88.3_{\uparrow 1.6}$	$76.5_{\uparrow 3.0}$	$37.6_{\uparrow 4.6}$	$34.1_{\uparrow 0.7}$
Layer 9, Head 9	$83.4_{\uparrow 0.1}$	$87.7_{\uparrow 1.0}$	$74.2_{\uparrow 0.7}$	$36.1_{\uparrow 3.1}$	$32.4_{\downarrow 1.0}$
Layer 10, Head 10	$81.5_{\downarrow 1.8}$	$84.7_{\downarrow 2.0}$	$72.2_{\downarrow 1.3}$	$41.1_{\uparrow 8.1}$	$38.7_{\uparrow 5.4}$
L2.H5 & L3.H7 & L4.H0	$80.4_{\downarrow 2.9}$	$3.6_{\downarrow 83.1}$	$11.1_{\downarrow 62.4}$	$20.1_{\downarrow 12.9}$	$4.7_{\downarrow 28.7}$

Table 3: The effects of ablating the 9 heads occurring in all MNLI subnetworks individually from the full model. It is noteworthy that some of these are also important for counter-heuristic behavior, as their ablation leads to reduced generalization. In the final row, Lx.Hy refers to Head y in Layer x.

Ablated Head	Accuracy (%) ↑	
	QQP	QQP-PAWS (adv.)
None	91.2	14.8
Layer 2, Head 3	$91.1_{\downarrow 0.1}$	$14.1_{\downarrow 0.7}$
Layer 2, Head 5	$91.1_{\downarrow 0.1}$	$9.9_{\downarrow 4.9}$
Layer 3, Head 3	$91.2_{\downarrow 0.0}$	$13.9_{\downarrow 0.9}$
Layer 4, Head 0	$90.2_{\downarrow 1.0}$	$3.3_{\downarrow 11.6}$
Layer 8, Head 11	$91.0_{\downarrow 0.2}$	$14.6_{\downarrow 0.2}$
Layer 9, Head 1	$91.2_{\downarrow 0.0}$	$16.6_{\uparrow 1.8}$
Layer 9, Head 9	$91.1_{\downarrow 0.1}$	$16.2_{\uparrow 1.4}$
Layer 11, Head 6	$91.1_{\downarrow 0.1}$	$17.1_{\uparrow 2.3}$

Table 4: The effects of ablating the 8 heads occurring in all QQP subnetworks individually from the full model. The heads with the strongest reduction are the same ones that had the largest effect on the MNLI model.

Surprisingly, however, many heads seem to be critical to performing well on HANS: for example, ablating a single head (L4.H0) reduces performance on *SO-Swap* by more than 40%. Moreover, the drops observed on ablating these heads is *super-additive*: ablating just three of them leads to the OOD accuracy plummeting. This indicates that heads outside the heuristic core rely on the heuristic core to implement more complex behavior.

Table 4 paints a similar picture for QQP. Perhaps most interestingly, the heads 2.5 and 4.0 that had the maximum ablation effect for MNLI also stand out here. Along with the fact that 5 of these 8 heads also exist in the MNLI model's core, this highlights the possibility that these heads already extract word overlap features in the BERT model prior to fine-tuning. We leave this question for future work.

6 Findings Generalize to RoBERTa and GPT-2

We show in Appendix D that our results extend to RoBERTa (Liu et al., 2019) models. Appendix E finds that they also transfer to GPT-2 (Radford et al., 2019) models, albeit with some novel behavior. Specifically, we observe that the accuracy on the *non-adversarial* OOD ('entailment') cases slightly drops late into training, and is accompanied by a mild increase in effective size. The performance on these cases also varies across different subnetworks of the same sparsity. Explaining these additional features requires further research and makes for future work. We also note that it is possible that models much larger (such as those with 7B+ parameters) take a different route to generalization. They might thus show qualitatively different behavior, and we urge caution in applying the results of this paper to them.

7 Related Work

Pruning. Pruning (LeCun et al., 1989) removes layers or parameters from a network for space efficiency and speed-ups. Structured Pruning (Wang et al., 2020; Ma et al., 2023; Xia et al., 2022, 2024) preserves some structure in the model, e.g. equal fraction of surviving parameters in each attention head, accepting lower compression in exchange for higher speed-ups. Unstructured pruning (Sun et al., 2024) approaches like Magnitude Pruning (LeCun et al., 1989; Hassibi and Stork, 1992) achieve better compression at the cost of lower potential speedup. Proposals such as Layer dropping (Zhang and He,

2020) and Block Pruning (Lagunas et al., 2021) have explored pruning at coarser granularities.

Pruning for interpretability. Mechanistic Interpretability seeks to understand models through their components, e.g., neurons, attention heads, and MLPs. Such analysis uncovers intricate interactions between different atomic components, which form *circuits* (Elhage et al., 2021; Olah et al., 2020) implementing various tasks. Pruning has occasionally been used as part of a larger interpretability effort (Lepori et al., 2023a), though mostly in an ad-hoc manner. Jain et al. (2023) use pruning and probing techniques to show that fine-tuning learns a thin "wrapper" around model capabilities. Lepori et al. (2023b) use pruning to identify subnetworks which they then show form modular building blocks of model behavior. In a similar vein, (De Cao et al., 2020) learn differentiable masks over the input to investigate model behavior. While most of these approaches replace missing layers with an identity function, work in Mechanistic Interpetability (Zhang and Nanda, 2024; Conmy et al., 2023) has shown that adding their mean activations might preserve faithfulness better. Prior work has examined attention patterns to decipher model behavior (e.g. Clark et al., 2019), although attention patterns have also been shown to be misleading in some cases (Jain and Wallace, 2019).

Generalization. The question of how a neural network generalizes has remained a compelling research direction for the NLP community (Yang et al., 2023). McCoy et al. (2020) find that fine-tuning BERT with different random seeds leads to disparate generalization despite similar in-domain accuracies. Other works have identified tasks where OOD accuracies increase long after in-domain accuracy saturates and attributed them to hard minority examples (Tu et al., 2020; Bhargava et al., 2021). Recently, a phenomenon called Grokking (Power et al., 2022) was discovered, where test accuracy on a toy algorithmic task increases abruptly long after overfitting to the training data. It has been explained in terms of competing subnetworks (Merrill et al., 2023; Varma et al., 2023). Grokking-like phenomena have also been identified in other settings (Liu et al., 2023) and for small transformers (Murty et al., 2023) on hierarchical depth generalization. Simultaneously, questions about memorizing in pretrained LMs have been asked in other settings (Tänzer et al., 2022). Friedman et al. (2024) document cases where simplifying a Transformer's representations preserves in-domain performance but impairs systematic generalization; we discover a similar "generalization gap" for another simplification method, pruning.

Subnetworks as modular components. An emerging body of work (Lepori et al., 2023b; Choenni et al., 2023; Hupkes et al., 2020) suggests that neural networks and language models may consist of modular subnetworks each performing different subtasks. Bayazit et al. (2023) find a subnetwork in GPT-2 responsible for most of its factual knowledge – removing it does not lead to loss of performance on non-knowledge-intensive tasks. Li et al. (2023) report the possibility of ablating away undesired behavior in a model (e.g. toxicity), while Wei et al. (2024) identify regions that when pruned away compromise the safety alignment of a Language Model. In concurrent work, Zhang et al. (2024) find a subset of less than 1% of an LM's parameters vital for linguistic competence across 30 languages—which they call the "Linguistic Core". Prakash et al. (2024) report that entity-tracking circuits in fine-tuned models exist even before fine-tuning, providing a potential explanation for why the heuristic core heads we found for MNLI and QQP had a substantial overlap. Panigrahi et al. (2023) find that grafting $\sim 0.01\%$ of model parameters from a fine-tuned version can recover most of the gains from fine-tuning.

8 Conclusion

We showed that pruning a pretrained LM with different random seeds can yield subnetworks that perform similarly to the original model in-domain but generalize differently. Moreover, sparser subnetworks generalize worse. To explain this result, we first considered the *Competing Subnetworks Hypothesis*, which has been used to explain grokking on algorithmic tasks; it suggests that the model consists of disjoint subnetworks representing different solutions, some which generalize and some which do not. We found that our results are instead more consistent with the existence of a *Heuristic Core*: a subset of attention heads that appear in all subnetworks and calculate shallow, non-generalizing features. The model generalizes by incorporating additional heads that interact with the heuristic core. Our results have practical implications about the effect of pruning on generalization and raise intriguing questions about how language models learn to generalize by composing simple components.

Limitations

Our approaches are not without limitations. Our analysis is based on a top-down approach of identifying subnetworks. Thus, we do not look for or discover "circuits" that are atomically interpretable. Our experiments only use text classification tasks, although we believe that such a task is ideal due to clear definitions of heuristics. Additionally, we only work with BERT models—it is possible that larger transformers, or those pretrained differently, exhibit different behavior. Finally, although we have analyzed the mechanisms of generalization in this paper, the question of why they are undertaken remains elusive for the moment. Our experiments also raised many novel questions, which require further research to answer.

Ethical Considerations

We do not foresee any ethical considerations raised due to our work. We work with pretrained language models, which can at times hallucinate or demonstrate other undesirable behavior. Our work, however, does not interact with these facets of model behavior. Additionally, we only work with English datasets in our paper.

Acknowledgements

We thank Michael Tang for his detailed suggestions and alacritous insights, without which this project would be incomplete. We are grateful to Alexander Wettig and Zirui Wang for their feedback on paper drafts. Finally, we thank everyone at the Princeton NLP group—especially Mengzhou Xia and Tianyu Gao, for fruitful discussions regarding our experiments. This research is funded by the National Science Foundation (IIS-2211779), a Sloan Research Fellowship, and a Hisashi and Masae Kobayashi *67 Fellowship.

References

Boaz Barak, Benjamin L. Edelman, Surbhi Goel, Sham M. Kakade, eran malach, and Cyril Zhang. 2022. Hidden progress in deep learning: SGD learns parities near the computational limit. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Deniz Bayazit, Negar Foroutan, Zeming Chen, Gail Weiss, and Antoine Bosselut. 2023. Discovering knowledge-critical subnetworks in pretrained language models. *arXiv preprint arXiv:2310.03084*.

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in NLI: Ways (not) to go beyond simple heuristics. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 125–135, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rochelle Choenni, Ekaterina Shutova, and Dan Garrette. 2023. Examining modularity in multilingual LMs via language-specialized subnetworks. *arXiv preprint arXiv:2311.08273*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? An analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards automated circuit discovery for mechanistic interpretability. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*.

Nicola De Cao, Michael Sejr Schlichtkrull, Wilker Aziz, and Ivan Titov. 2020. How do decisions emerge across layers in neural models? interpretation with differentiable masking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3243–3255, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional Transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Dan Friedman, Andrew Kyle Lampinen, Lucas Dixon, Danqi Chen, and Asma Ghandeharioun. 2024. Interpretability illusions in the generalization of simplified models. In *International Conference on Machine Learning (ICML)*.

Babak Hassibi and David Stork. 1992. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 5. Morgan-Kaufmann.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization. Journal track.

Shankar Iyer, Nikhil Dandekar, Kornél Csernai, et al. 2017. First Quora dataset release: Question pairs. *quora.com*.

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P. Dick, Hidenori Tanaka, Edward Grefenstette, Tim Rocktäschel, and David Scott Krueger. 2023. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks. *arXiv preprint arXiv:2311.12786*.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. 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)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

François Lagunas, Ella Charlaix, Victor Sanh, and Alexander Rush. 2021. Block pruning for faster transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10619–10629, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 2. Morgan-Kaufmann.

Michael A. Lepori, Ellie Pavlick, and Thomas Serre. 2023a. NeuroSurgeon: A toolkit for subnetwork analysis. *arXiv preprint arXiv:2309.00244*.

Michael A. Lepori, Thomas Serre, and Ellie Pavlick. 2023b. Break it down: Evidence for structural compositionality in neural networks. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*.

Maximilian Li, Xander Davies, and Max Nadeau. 2023. Circuit breaking: Removing model behaviors with targeted ablation. *Workshop on Challenges in Deployable Generative AI at the International Conference on Machine Learning (ICML)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ziming Liu, Eric J Michaud, and Max Tegmark. 2023. Omnigrok: Grokking beyond algorithmic data. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. Learning sparse neural networks through l_0 regularization. In *International Conference on Learning Representations (ICLR)*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-pruner: On the structural pruning of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*.

R. Thomas McCoy, Junghyun Min, and Tal Linzen. 2020. BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 217–227, Online. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Association for Computational Linguistics (ACL)*, pages 3428–3448.

William Merrill, Nikolaos Tsilivis, and Aman Shukla. 2023. A tale of two circuits: Grokking as competition of sparse and dense subnetworks. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher Manning. 2023. Grokking of hierarchical structure in vanilla transformers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 439–448, Toronto, Canada. Association for Computational Linguistics.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*.

Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. 2023. Task-specific skill localization in fine-tuned language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 27011–27033. PMLR.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. 2022. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*.

Nikhil Prakash, Tamar Rott Shaham, Tal Haklay, Yonatan Belinkov, and David Bau. 2024. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *arXiv*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Michael Tänzer, Sebastian Ruder, and Marek Rei. 2022. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7564–7578, Dublin, Ireland. Association for Computational Linguistics.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics (TACL)*, 8:621–633.

Vikrant Varma, Rohin Shah, Zachary Kenton, János Kramár, and Ramana Kumar. 2023. Explaining grokking through circuit efficiency. *arXiv preprint arXiv:2309.02390*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30. Curran Associates, Inc.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2020. Structured pruning of large language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162, Online. Association for Computational Linguistics.

Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. 2024. Assessing the brittleness of safety alignment via pruning and low-rank modifications. In *International Conference on Machine Learning (ICML)*.

Kaiyue Wen, Yuchen Li, Bingbin Liu, and Andrej Risteski. 2023. Transformers are uninterpretable with myopic methods: A case study with bounded Dyck grammars. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1112–1122.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1513–1528, Dublin, Ireland. Association for Computational Linguistics.

Linyi Yang, Yaoxian Song, Xuan Ren, Chenyang Lyu, Yidong Wang, Jingming Zhuo, Lingqiao Liu, Jindong Wang, Jennifer Foster, and Yue Zhang. 2023. Out-of-distribution generalization in Natural Language Processing: Past, present, and future. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4533–4559, Singapore. Association for Computational Linguistics.

Fred Zhang and Neel Nanda. 2024. Towards best practices of activation patching in language models: Metrics and methods. In *The Twelfth International Conference on Learning Representations*.

Minjia Zhang and Yuxiong He. 2020. Accelerating training of transformer-based language models with progressive layer dropping. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 14011–14023. Curran Associates, Inc.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. 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)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhihao Zhang, Jun Zhao, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024. Unveiling linguistic regions in Large Language Models. *arXiv preprint arXiv:2402.14700*.

A Model and Training Details

BERT Architecture Our experiments use fine-tuned BERT models (Devlin et al., 2019). In the simplest setting, a Transformer Model (Vaswani et al., 2017) consists of L blocks, where each block is in turn composed of a Multi-Head *Attention* layer followed by an *MLP* layer. Given an input X, the Attention Layer outputs

$$X + \sum_{i=1}^{N_h} \text{Attn}(W_Q^{(i)}, W_K^{(i)}, W_V^{(i)}, X) W_O^{(i)}$$

where N_h is the number of attention heads, and $W_K^{(i)}$, $W_Q^{(i)}$ and $W_V^{(i)}$ denote the Key, Query, and Value matrices, respectively. $W_O^{(i)}$ performs a linear output projection, before adding the result to the *residual stream* via a skip connection. On the other hand, an MLP layer first up-projects its input X by multiplying with $W_U \in \mathbb{R}^{d \times d_h}$, then passes the result through an activation function such as GeLU, then down-projects the result back via $W_D \in \mathbb{R}^{d \times d_h}$. Here d is the hidden dimension and d_h typically equals $4d$. This is then added to the residual stream to produce

$$X + \text{GeLU}(XW_U)W_D$$

Layer Normalization precedes both layers.

BERT Fine-Tuning We fine-tuned BERT (`bert_base_cased`, 125M parameters) on MNLI for 5 epochs (61360 steps with batch size $= 32$), with a learning rate of $2 \cdot 10^{-5}$, a and a maximum sequence length of 128. Checkpointing and evaluation were performed every 500 steps. For QQP, we opted for 40000 steps of fine-tuning with an effective batch size of 256 and a learning rate of $2 \cdot 10^{-5}$. Evaluation and checkpoint were once again performed every 500 steps. All runs used the Adam optimizer (Kingma and Ba, 2015), with $\beta = (0.9, 0.999)$ and $\epsilon = 10^{-8}$.

Each run takes approximately 3 hours to complete on one NVIDIA A5000 GPU.

B Details of Pruning and L0 Regularization

Our formulation of the masks is the same as CoFi Pruning (Xia et al., 2022)'s, and is built on top of Louizos et al. (2018)'s work. Specifically, the masks z are modeled as hard concrete distributions:

$$\mathbf{u} \sim \text{Uniform}(\epsilon, 1-\epsilon)$$

$$\mathbf{s} = \sigma\left(\frac{1}{\beta} \cdot \frac{\mathbf{u}}{1-\mathbf{u}} + \log \alpha\right)$$

$$\tilde{\mathbf{s}} = \mathbf{s} \times (r - l) + l$$

$$\mathbf{z} = \min(1, \max(0, \tilde{\mathbf{s}}))$$

Here, σ denotes the sigmoid function, and $\epsilon = 10^{-6}$ is chosen to avoid division by 0 errors. The temperature is fixed at $\frac{1}{\beta} = \frac{2}{3}$. The result is then stretched to the interval $[l, r] = [-0.1, 1.1]$, with the probability mass from $[-0.1, 0]$ and $[1, 1.1]$ accumulated to 0 and 1 in the end. This ensures that the mask is incentivized to contain values close to 0 or 1. Here, the log alphas $\log \alpha$ are the main learnable parameters.

Following Wang et al. (2020) and Xia et al. (2022), we enforce a target sparsity via a Lagrangian Term. Supposing the target and current sparsity to be s and t, we add

$$\mathcal{L}_s = \lambda_1(t - s) + \lambda_2(t - s)^2$$

to the loss (CoFi uses $s - t$ instead of $t - s$, but the two are functionally the same; consider the transform $\lambda_1 \longrightarrow -\lambda_1$). Along with training the model parameters, a gradient *ascent* is performed on λ_1 and λ_2, so that the updates of the lambdas always keep adjust the contribution of $\mathcal{L}_s$ to keep the sparsity in check.

Pruning - General Commentary We found that the hyperparameters were very sensitive to the choice of learning rate. Deviation even by a factor of 3 would cause instability in training. Specifically, if the learning rate for the ascent on the Lagrangian Parameters (λ_1 and λ_2) was too low, the model would stay under a sparsity of 10^{-5} until the target sparsity climbed to 0.15 or so, and would then jump to a value beyond the target, sacrificing a lot of performance in the process. On the other hand, a large value also leads to insufficient time for "settling into" the intermediate sparsities. The exact choice of the number of optimization steps was also important, although not as sensitive as the learning rate.

BERT Pruning We used a maximum sequence length of 128 as before, with an effective batch size of 128 including gradient accumulation. The learning rates for log alphas and lambdas were 0.1 and 1, respectively. We use the formulae

$$\texttt{warmup\_steps} = 6500 + 50 \times \texttt{sparsity\_pct}$$

$$\texttt{total\_steps} = \texttt{warmup\_steps} + 6 \times \texttt{sparsity\_pct}$$

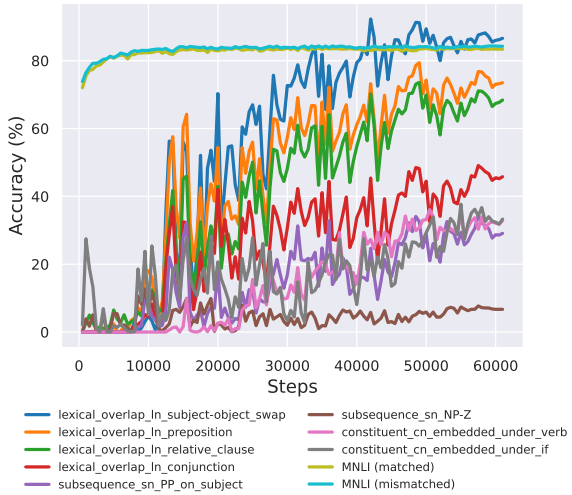


Figure 10: Model performance saturates early on MNLI, but generalizing traits are learned much later.

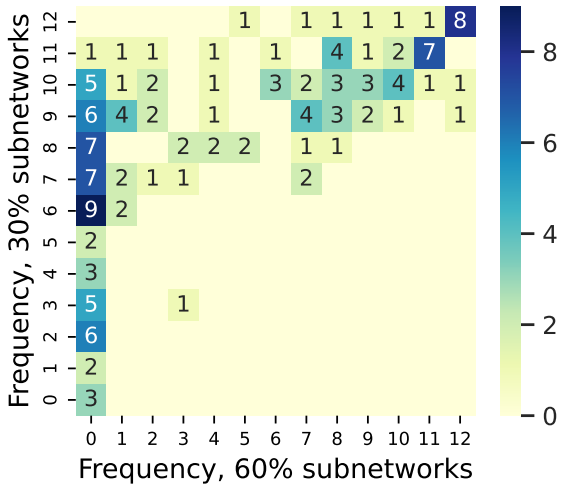


Figure 11: This heatmap quantifies the frequencies of attention heads in the 30% and 60% sparsity subnetworks of the QQP model. Entry (i, j) denotes the number of heads appearing in $i/12$ 30% subnetworks and $j/12$ 60% subnetworks. Eight attention heads appear in all subnetworks.

to compute the number of steps taken to linearly warm up the target sparsity, and the total number of steps, respectively. For sparsity targets of 50%, these translate to 9300 training steps, of which the first 9000 warm up the target sparsity. For 70% target, these change to 10500 and 10000, respectively. Evaluation and checkpointing were done every 64 steps but we always used the final checkpoint to report results. The thresholds were chosen with a grid search on $[0, 1]$ with stride 0.05 to match the target sparsity as closely as possible. They were found to always be 0.5 and 0.35 for target sparsities of 0.5 and 0.7.

Each pruning run takes approximately 90 minutes to complete on an A5000 GPU. We estimate our total computational budget to be 750 GPU hours.

C More results

In this section, we provide further exposition on some of the results from the main text. For these results, we use the full names of the HANS subcases instead of abbreviations.

Figure 10 showcases the in-domain v/s out-of-domain training dynamics on MNLI/HANS for more subcases. The trend remains the same—OOD accuracy starts increasing much after ID accuracy saturates. For the other subcases not shown here, the model never gets off the ground.

Figures 13 and 14 show that the frequency of the heads is strongly correlated between the 50% and 70% sparsity subnetworks for MNLI/HANS, and the 30% and 60% sparsity subnetworks for QQP. This point is further elaborated by the frequency heatmap of Figure 11, the counterpart of Figure 5. Essentially, the heads most important to the heuristic subnetworks also appear often in the generalizing subnetworks, once again showing that the heuristic subnetworks are subsets of generalizing subnetworks.

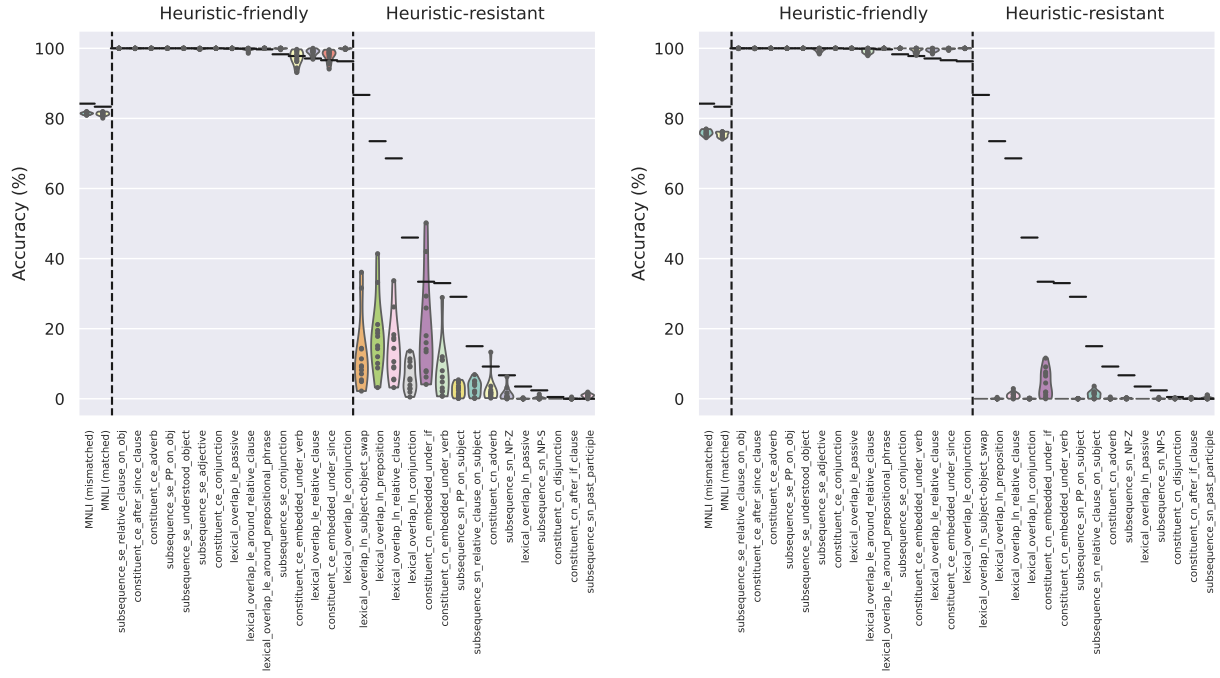
Subcase	Model	Subnetworks, 50% sp.			
		Min.	*Max.*	*Mean*	*STD*
MNLI (m)	87.5	84.1	85.8	84.9	0.4
MNLI (mm)	87.2	84.1	85.7	84.8	0.43
Prep	93.2	52.2	86.4	69.1	8.9
SO-Swap	99.4	46.0	87.0	67.4	12.2
Embed-If	83.1	45.6	79.8	62.2	11.0
Embed-Verb	72.6	42.3	75.6	60.4	9.6

Table 5: Different subnetworks at 50% sparsity show ID performance similar to the full RoBERTa model, but generalize differently.

D Results on RoBERTa

In this section, we verify that our findings extend to a RoBERTa (Liu et al., 2019) (`roberta-base`) model. We follow the same recipes as the main text, using the same hyperparameters as Appendices A and B for fine-tuning and pruning. Once again, we evaluate on the subcases SO-Swap, Prep, Embed-If, and Embed-Verb in the MNLI-HANS setting.

Table 5 demonstrates that different subnetworks at 50% sparsity still generalize differently despite



(a) Sparsity = 50%

(b) Sparsity = 70%

Figure 12: The full results of the multi-seed pruning experiment on MNLI/HANS. The subnetworks perform close to the model both in-domain and on heuristic-friendly subcases of HANS. Ay 50% sparsity, they show disparate generalization to the heuristic-resistant subcases, while at 70%, they degenerate to heuristic behavior. This is also corroborated by the fact that they slightly overperform the full-model on the heuristic-friendly cases.

performing similarly to the full model in-domain. Figure 15 tells us that the effective size again goes up along with generalization. The 12 subnetworks share 11 attention heads that also appear in the effective size subnetworks before generalization. We list these heads along with the effect of ablating them in Table 6. The effects are smaller but consistent with the findings on the BERT model: these heads have much larger ablation effects than random other heads despite not generalizing on their own.

E Results on GPT-2

In this subsection, we ask if our findings and explanations can extend to decoder-only models. We work with a GPT-2 small (117M) model (Radford et al., 2019). Most of the hyperparameters for both fine-tuning and pruning are the same as those listed in Appendices A and B. However, we found that the GPT-2 model, even as a whole, generalizes very poorly to the HANS subcases. After training for 10 epochs (12270 steps), `constituent_cn_embed_under_if` is the only "contradiction" subcase with an accuracy above 30%. Hence, we select this setting, and Embed-If as the only adversarial out-of-domain subcase. Interestingly, however, we observed that the accuracies of some of the *heuristic-friendly* subcases started dropping later into the training. We select `subsequence_se_relative_clause_on_obj` (Rel-Clause [SE]) and `constituent_ce_embedded_under_since` (Embed-Since [CE]) as two representative subcases and group them under the split OOD-E (OOD-Entailment). We perform the experiments for both the OOD and OOD-E subcases.

We evaluate 12 pruned subnetworks at 40% and 60% sparsity in Table 7. Like before, the subnetworks perform similarly in-domain but generalize differently. However, one striking feature stands out. As we increase the sparsity, the average performance on the 'contradiction' OOD cases *increases*, whereas that of the 'entailment' cases *decreases*.

One way to reconcile these results is for some heuristics to be *contradiction* heuristics, and for the model to develop further non-heuristic heads around a *contradiction* heuristic core. Indeed, in Figure 16, we note two interesting trends: (1) the model accuracy decreases slightly on the OOD-E subcases late into training and (2) a mild increase

Ablated Head	Accuracy (%) ↑				
	MNLI (matched)	SO-Swap [LO]	Prep [LO]	Embed-If [C]	Embed-Verb [C]
None	87.5	99.4	93.2	83.1	72.6
Avg. (non-HC)	$87.3_{\downarrow 0.2}$	$99.1_{\downarrow 0.3}$	$93.1_{\downarrow 0.1}$	$83.2_{\uparrow 0.1}$	$71.8_{\downarrow 0.8}$
Layer 2, Head 4	$87.5_{\downarrow 0.0}$	$95.9_{\downarrow 3.5}$	$89.8_{\downarrow 3.4}$	$84.1_{\uparrow 1.0}$	$62.5_{\downarrow 10.1}$
Layer 3, Head 4	$87.2_{\downarrow 0.3}$	$95.3_{\downarrow 4.1}$	$86.8_{\downarrow 6.4}$	$80.8_{\downarrow 2.3}$	$62.6_{\downarrow 10.0}$
Layer 4, Head 9	$86.9_{\downarrow 0.6}$	$98.8_{\downarrow 0.6}$	$93.5_{\uparrow 0.3}$	$84.0_{\uparrow 0.9}$	$70.8_{\downarrow 1.8}$
Layer 5, Head 1	$87.3_{\downarrow 0.2}$	$98.5_{\downarrow 0.9}$	$91.8_{\downarrow 1.4}$	$75.5_{\downarrow 7.6}$	$78.9_{\uparrow 6.3}$
Layer 5, Head 8	$86.9_{\downarrow 0.6}$	$98.9_{\downarrow 0.5}$	$94.6_{\uparrow 1.4}$	$76.4_{\downarrow 6.7}$	$60.7_{\downarrow 11.9}$
Layer 6, Head 2	$87.5_{\downarrow 0.0}$	$98.1_{\downarrow 1.3}$	$91.5_{\downarrow 1.7}$	$77.3_{\downarrow 5.8}$	$62.0_{\downarrow 10.5}$
Layer 6, Head 10	$87.1_{\downarrow 0.4}$	$99.0_{\downarrow 0.4}$	$94.2_{\uparrow 1.0}$	$80.5_{\downarrow 2.6}$	$71.7_{\downarrow 0.9}$
Layer 6, Head 11	$87.5_{\downarrow 0.0}$	$89.8_{\downarrow 9.6}$	$73.9_{\downarrow 29.3}$	$80.4_{\downarrow 2.7}$	$59.3_{\downarrow 13.3}$
Layer 7, Head 9	$87.3_{\downarrow 0.2}$	$99.6_{\uparrow 0.2}$	$95.2_{\uparrow 2.0}$	$87.2_{\uparrow 4.1}$	$62.6_{\downarrow 10.0}$
Layer 7, Head 10	$87.1_{\downarrow 0.4}$	$98.4_{\downarrow 1.0}$	$89.9_{\downarrow 3.3}$	$77.6_{\downarrow 5.5}$	$58.9_{\downarrow 13.7}$
Layer 8, Head 3	$87.3_{\downarrow 0.2}$	$99.3_{\downarrow 0.1}$	$93.1_{\downarrow 0.1}$	$87.1_{\uparrow 4.0}$	$68.5_{\downarrow 3.9}$

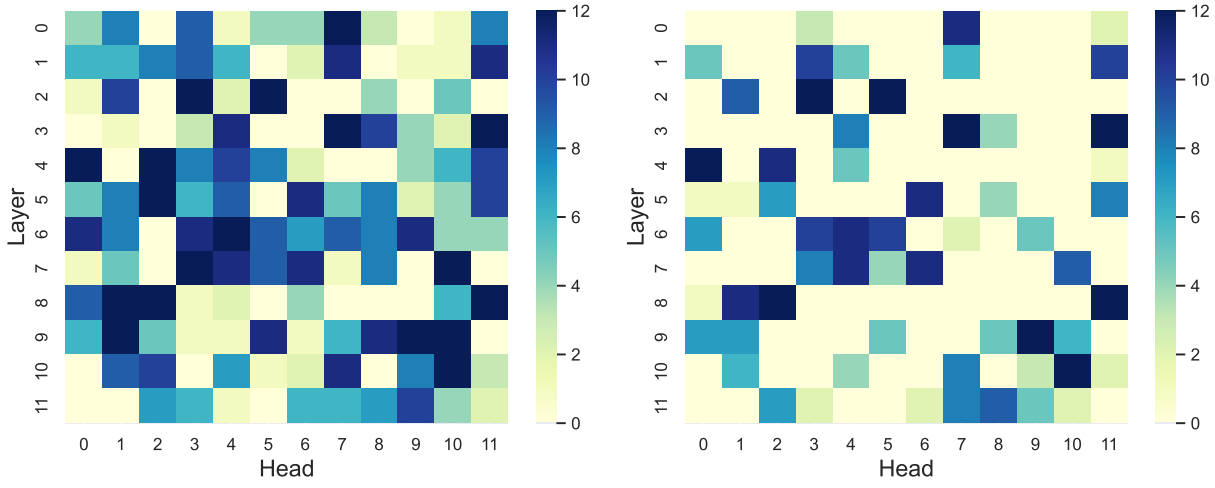
Table 6: The heuristic core of RoBERTa, and the ablation effects of its attention heads. Despite not generalizing on their own, these cause much larger ablation effects than random other heads—head 6.11 shows the maximum effect.

Subcase	Model	Subnetworks, 40% sp.				Subnetworks, 60% sp.			
		Min.	*Max.*	*Mean*	*STD*	*Min.*	*Max.*	*Mean*	*STD*
In-domain									
MNLI (matched)	82.0	78.7	80.3	79.4	0.5	74.3	76.9	75.5	0.8
MNLI (mismatched)	82.2	79.3	80.8	80.1	0.5	74.4	77.5	76.3	1.1
Out-of-domain (contradiction)									
Embed-If [C]	47.4	17.8	36.3	26.9	5.8	16.8	51.9	33.9	12.2
Out-of-domain (entailment)									
Rel-Clause [S]	100.0	96.4	99.7	98.1	1.1	66.3	97.8	85.3	9.8
Embed-Since [C]	88.8	85.0	98.6	92.0	4.2	57.9	92.0	79.9	11.7

Table 7: The performance of 12 different pruned subnetworks of GPT-2 at 40% and 60% sparsity. We still see massive variance out-of-domain compared to in-domain evaluation but the performance on the 'contradiction' examples (OOD) goes up with sparsity, and that of 'entailment' (OOD-E) goes down.

Ablated Head	Accuracy (%) ↑			
	ID	*OOD*	*OOD-E*	
	MNLI (matched)	Embed-If [C]	Rel-Clause [S]	Embed-Since [C]
None	82.0	47.4	100.0	88.8
Avg. (non-HC)	$81.9_{\downarrow 0.1}$	$47.2_{\downarrow 0.2}$	$100.0_{\downarrow 0.0}$	$89.0_{\uparrow 0.2}$
Layer 0, Head 7	$81.6_{\downarrow 0.4}$	$49.4_{\uparrow 2.0}$	$100.0_{\downarrow 0.0}$	$88.5_{\downarrow 0.3}$
Layer 0, Head 9	$81.8_{\downarrow 0.2}$	$46.8_{\downarrow 0.6}$	$100.0_{\downarrow 0.0}$	$88.2_{\downarrow 0.6}$
Layer 1, Head 0	$81.2_{\downarrow 0.8}$	$34.5_{\downarrow 12.9}$	$100.0_{\downarrow 0.0}$	$92.4_{\uparrow 3.6}$
Layer 1, Head 1	$81.9_{\downarrow 0.1}$	$48.2_{\uparrow 0.8}$	$100.0_{\downarrow 0.0}$	$89.1_{\uparrow 0.3}$
Layer 3, Head 0	$81.9_{\downarrow 0.1}$	$51.6_{\uparrow 4.2}$	$99.9_{\downarrow 0.1}$	$87.1_{\downarrow 1.7}$
Layer 4, Head 9	$81.4_{\downarrow 0.6}$	$34.5_{\downarrow 12.9}$	$100.0_{\downarrow 0.0}$	$91.9_{\uparrow 3.1}$
Layer 4, Head 11	$80.9_{\downarrow 1.1}$	$54.5_{\uparrow 7.1}$	$100.0_{\downarrow 0.0}$	$88.4_{\downarrow 0.4}$
Layer 5, Head 5	$80.7_{\downarrow 1.3}$	$57.6_{\uparrow 10.2}$	$99.5_{\downarrow 0.5}$	$85.1_{\downarrow 3.7}$
Layer 5, Head 8	$81.5_{\downarrow 0.5}$	$44.2_{\downarrow 3.2}$	$100.0_{\downarrow 0.0}$	$89.4_{\uparrow 0.6}$
Layer 6, Head 3	$81.2_{\downarrow 0.8}$	$40.6_{\downarrow 6.8}$	$100.0_{\downarrow 0.0}$	$90.4_{\uparrow 1.6}$
Layer 6, Head 7	$81.0_{\downarrow 1.0}$	$52.5_{\uparrow 5.1}$	$100.0_{\downarrow 0.0}$	$88.5_{\downarrow 0.3}$
Layer 6, Head 11	$81.5_{\downarrow 0.5}$	$44.5_{\downarrow 2.9}$	$100.0_{\downarrow 0.0}$	$89.9_{\uparrow 1.1}$

Table 8: The heads common to all 40% and 60% sparsity subnetworks of GPT-2, and their ablation effects. On Embed-If, we see that some heads (4.11, 5.5, 6.7) show an increased accuracy on ablation and some (1.0, 4.9, 6.3) a strong decrease. On the 'entailment' subcases, the ablation effects of individual heads is generally small.



(a) Sparsity $= 50\%$

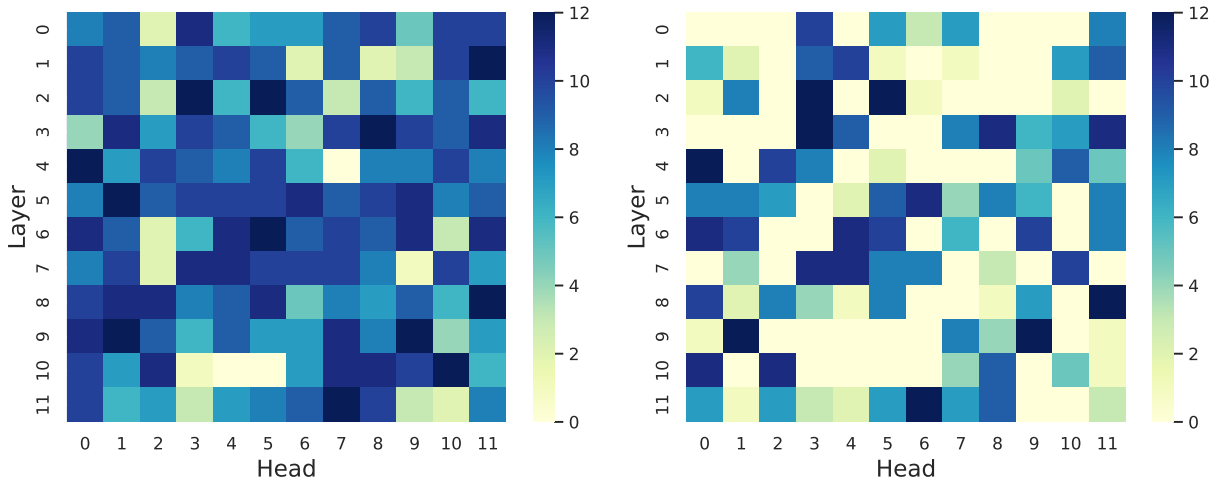
(b) Sparsity $= 70\%$

Figure 13: The frequency of various attention heads across the subnetworks pruned with 12 random seeds. The heads part of the 70% sparsity subnetworks (non-generalizing) are also popular in 50% sparsity (partially generalizing) subnetworks. The Separman's Rho agreement between the frequencies of attention heads at the two sparsities is 0.82 (p-value = $1.6 \cdot 10^{-36}$), corresponding to very strong agreement.

in the OOD-E effective size accompanies this decrease in accuracy. A possible explanation for the latter could be that some heuristics for spotting 'contradiction' are also learned, and play an analogous role in the heuristic core.

We find the heads common to all 40% and 60% subnetworks and tabulate their ablation effects in Table 8. We note the following: (1) On Embed-If, some heads show a strong increase in accuracy on ablation, and some a strong decrease. This would be consistent with some heads implementing entailment heuristics and some contradiction heuristics. On the other hand, (2) None of the heads show large ablation effects on Rel-Clause. On Embed-Since—the other entailment subcase—some heads show small but significant effects.

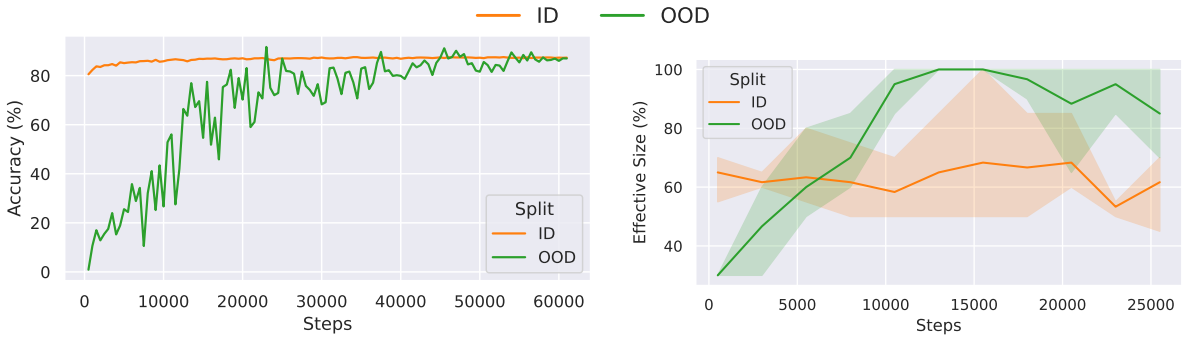
Therefore, our results on the OOD subcases are consistent with the model implementing both entailment and contradiction heuristics. On the other hand, some questions remain regarding the OOD-E subcases in GPT-2—namely, why the performance of the model tapers off late into the training, and why there is a variance in performance at *high* sparsities (in contrast to the results on BERT) despite none of the common heads having significant ablation effects. We leave it to future work to answer these questions.



(a) Sparsity $= 30\%$

(b) Sparsity $= 60\%$

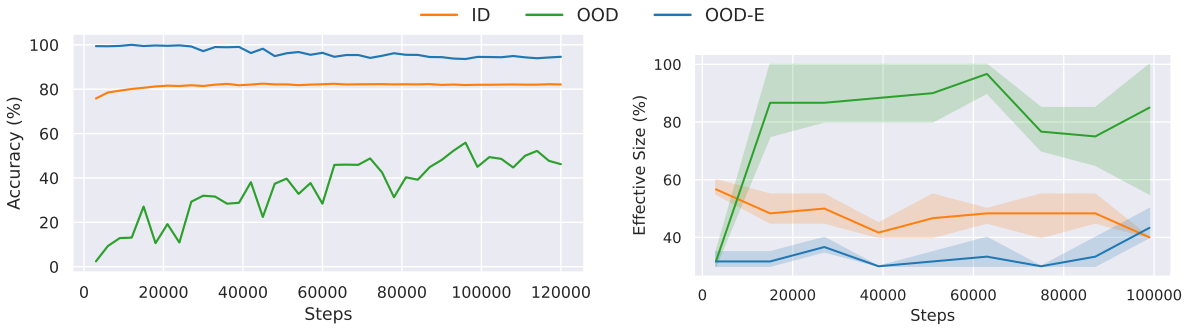
Figure 14: The frequency of various attention heads across in the QQP subnetworks, pruned with 12 random seeds. Once again, the frequencies of the heads are highly correlated between the two sparsities. The Separman's Rho agreement in this case is 0.74 (p-value = $7.7 \cdot 10^{-26}$), corresponding to strong agreement.



(a) ID/OOD Accuracy

(b) Effective Size

Figure 15: The accuracy and effective size of the RoBERTa model over training (3 seeds). Generalization is accompanied by an increase in effective size once again. The effective size plot includes three seeds at each sparsity.



(a) ID/OOD Accuracy

(b) Effective Size

Figure 16: The accuracy and effective size of the GPT-2 model over training (3 seeds). The effective size on OOD increases with generalization. Interestingly, the accuracy on the "entailment" OOD (OOD-E) cases decays slowly later into the training, and is accompanied by a slight increase in the corresponding effective size.