Transfer Learning for Structured Pruning under Limited Task Data

Lucio Dery[1], **Awni Hannun**[2], **David Grangier**[2]
[1]Carnegie Mellon University, [2]Apple Machine Learning Research,
`ldery@andrew.cmu.edu`, `{awni,grangier}@apple.com`

Abstract

Pre-trained models are growing increasingly large which can be problematic for applications with strong inference constraints. Fortunately, task-aware structured pruning offers a solution. While existing pruning algorithms can be efficient, the common practical setting where task-specific data is limited is yet to be addressed. To address the data scarcity problem, we propose a structured pruning strategy that leverages transfer learning. Detailed analyses of simple transfer learning based remedies lead us to a simple, flexible formulation of what, how and when to transfer, resulting in pruned models with improved generalization over strong baselines.

1 Introduction

Large pre-trained language models have been successfully applied to a wide variety of application scenarios. However, not all applications can justify the cost of running such large models. E.g. an interactive, offline spellchecker for a phone has strong memory limits compared to a server-side chat model [5]. Even server-side, the benefit/cost of large models depends on the application. This situation motivates research into structured model pruning algorithms.

Structured pruning algorithms generate smaller, faster and yet reasonably accurate sub-models from large pre-trained ones by removing components (beyond individual parameters) like convolutional channels, attention heads and whole layers. Several works over the years [19, 15, 20] have been proposed to perform task-specific structured pruning. Unfortunately, to the best of our knowledge, all existing algorithms have been developed without consideration for the amount of training data available for the target task. Thus, as Figure 1 shows that, even state-of-the-art methods like CoFi [20], do not gracefully handle scenarios with limited training data. We argue that the data-limited structured pruning setting is important since limited compute for inference and data scarcity for training co-occurs often in practice [1].

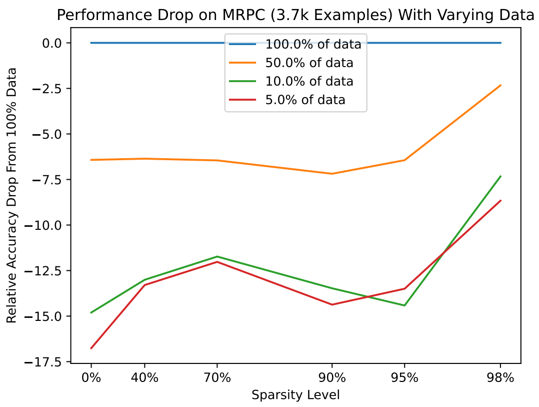

Figure 1: Accuracy degradation of CoFi [20] vs training data sizes. Sparsity level refers to the fraction of preserved weights (excluding embeddings). Accuracy at 50% data is stable across sparsity levels (except for 98% sparsity) while more data-limited regimes (10%–5%) exhibit stronger sensitivity to the sparsity level.

A popular remedy to the limited data problem at fixed model size, is to leverage transfer learning [3, 7, 4] by introducing external data or extra tasks. In this work, we investigate transfer learning based remedies for structured pruning under limited data. Structured pruning algorithms need to jointly learn both model weights and structural variables (which layers, attention heads, etc. to prune) for the final reduced-size model [19, 20]. This added complexity makes deploying transfer learning in the structured pruning setting non-trivial and raises several questions. Do we only perform transfer learning for model weights or do we include structural variables too? How do we learn structural variables for the target task in a way that benefits from the presence of a transfer task? When is it best to introduce transfer learning so as to produce the most accurate pruned target model?

This work aims to provide answers to these questions. We contribute (i) a flexible weight sharing δ formulation for effective transfer of both structural variables and model parameters and (ii) empirical analyses to provide prescriptions to researchers about what, how and when to transfer during structured pruning. Our effort results in significant improvements in generalization performance even at compression ratios as high as $50\times$.

2 Methodology

Structured pruning algorithms remove components from pre-trained models such as attention heads [14, 17], whole layers [8] or intermediate dimensions of fully connected layers [19] in order to produce faster, memory efficient sub-models without overly sacrificing downstream accuracy. This is unlike unstructured pruning [9, 2] that do not exhibit run-time speedups. In this work, we assume that we have a structured pruning algorithm that jointly learns structural variables $\{\mathbf{z}^k_{\text{target}}\}$ and their corresponding parameters $\{\theta^k_{\text{target}}\}$ for the target task. We are primarily concerned with how to incorporate a transfer task by learning $\left[\{\mathbf{z}^k_{\text{transfer}}\}, \{\theta^k_{\text{transfer}}\}\right]$ such that we enjoy improved generalization on the target task's final model. We focus on building on top of a state-of-the-art structured pruning algorithm, CoFi [20]. Whilst we describe CoFi below, for the rest of the paper, we will abstract away the details of the pruning algorithm and focus on the specifics of our transfer learning approach.

2.1 CoFi

CoFi (**Co**arse- and **Fi**ne-grained Pruning) is a mixed resolution structured pruning algorithm. Previous algorithms to prune transformer models [16] have focused on pruning high level units like whole layers [8] or intermediate ones like attention heads [17] or dimensions of fully connected layers [19]. CoFi introduces variables that account for pruning at multiple levels of granularity.
Coarse Grain: CoFi introduces variables sets $\{z^i_{\text{MHA}}\}_{i\in[N]}$ and $\{z^i_{\text{FFN}}\}_{i\in[N]}$ for each of the model N layers. z^i_{MHA} represents the probability that the **whole** attention component of the ith layer is removed whilst z^i_{FFN} is similarly defined for the fully connected component of the specified layer.
Fine Grain: Given a particular layer i, CoFi prunes subsets of the attention heads available. The variables $\{z^i_{j,\text{head}}\}_{[j\in n_h]}$ represent the jth attention head in the i layer which has n_h total attention heads. A similar set of variables is defined for the fully connected units within a layer : $\{z^i_{j,\text{FC}}\}_{[j\in n_f]}$ where the ith fully connected layer has n_f units.

For the jth attention head of the ith layer, the likelihood that this head is left unpruned is proportional to $z^i_{\text{MHA}} \cdot z^i_{j,\text{head}}$. This allows the algorithm to make coupled fine and coarse grained decisions that lead to improved results. We collectively represent $\{\mathbf{z}\}$ as the set of all structural variables that are learned by CoFi. $\{\mathbf{z}\}$ are learned by applying the reparameterisation trick on the hard concrete distribution [12] and minimizing a joint loss wrt $\{\mathbf{z}, \theta\}$ that includes distance from target size, target task loss and a distillation objective on the original large model.

2.2 Transfer Learning

When targeting a task $\mathbf{T}$ with limited training data, we want to improve CoFi by leveraging additional training data from an auxiliary task $\mathbf{A}$. At sparsity level γ, we expect the generalization performance when we learn $\{\mathbf{z}\}$ using only data from $\mathbf{T}$ to be less than when we learn them using data from $\mathbf{A}$ and $\mathbf{T}$ jointly in some appropriate form. We explore several options for incorporating the auxiliary task:
Single mask multi-task learns a single set of structural $\{\mathbf{z}\}$ and model $\{\theta\}$ parameters that are shared between both tasks. This choice tightly couples the two tasks. Whilst this allows maximal sharing of information between the target and transfer task, poor choices of transfer tasks could cause

this to perform worse than no transfer at all.
Multi-mask multi-task learns distinct structural parameters $\{\mathbf{z}\}_{\mathbf{T}}$ and $\{\mathbf{z}\}_{\mathbf{A}}$ for each task but a single set of model parameters $\{\theta\}$ is shared between both tasks. There is no transfer of structural information and only the shared model parameters provide a coupling of the two tasks.
Our δ-Formulation aims to leverage strength from both alternatives. In this method, both tasks share a base set of structural variables $\{\mathbf{z}\}_{\text{base}}$ but also have task specific addends such that: $\{\mathbf{z}\}_{\mathbf{T}} = \{\mathbf{z}_{\text{base}} + \delta_{\mathbf{T}}\}$ and $\{\mathbf{z}\}_{\mathbf{A}} = \{\mathbf{z}_{\text{base}} + \delta_{\mathbf{A}}\}$. We regularize δ_* to encourage sharing between tasks via $\mathbf{z}_{\text{base}}$ whilst maintaining flexibility for task-specific modelling.

3 Experimental Setup

Datasets We consider 2 pairs of tasks. One pair of classification tasks from the computer science domain tasks – SCIIE [13] and ACL-ARC [11] with 3.2k and 3.7k training samples respectively. The other pair is from GLUE [18]: STSB and MRPC are sentence similarity and paraphrase detection tasks with 7k and 3.7k train examples respectively. For the GLUE tasks, we follow previous work [10, 19, 20] and report results on the validation set. Appendix B.1 provides additional information about the datasets.
Model Details To compare with CoFi [20], we use the same model configuration. We use the BERT$_{\text{base}}$ [6] which has $\sim$ 110M parameters. We explore pruned model sparsities in the set $\{40\%, 70\%, 90\%, 95\%, 98\%\}$. $\gamma\%$ sparsity means that the model has been reduced to $(100-\gamma)\% \times$ 110M parameters. Similar to [15] we also freeze the model embedding weights. See Appendix B for details about training as well as hyper-parameter values.

4 Results And Discussion

4.1 How should you transfer?

In Section 2.2, we introduced various approaches for leveraging transfer learning for structured pruning. Figure 2, shows experimental results after implementing various options. For the SCIIE task, using ACL_ARC as an auxiliary task can negatively impact performance (single mask multi-task) compared to no transfer at all. Our δ-Formulation ensures that SCIIE actually benefits introducing transfer learning. For the ACL_ARC task, our formulation recovers close to the best performance (single mask multi-task). Note that in principle, our formulation can mimic the Single-mask multitask setting by using a high enough regularization on the δ offsets but we used a default l_2-regularization strength of $1e^{-2}$. From Figure 3, we see that using the Multi mask multi-task strategy with the MRPC task (STSB as transfer task) does not maximally exploit task relatedness for improved generalization. However, our δ-formulation gives improved results for both STSB and MRPC.

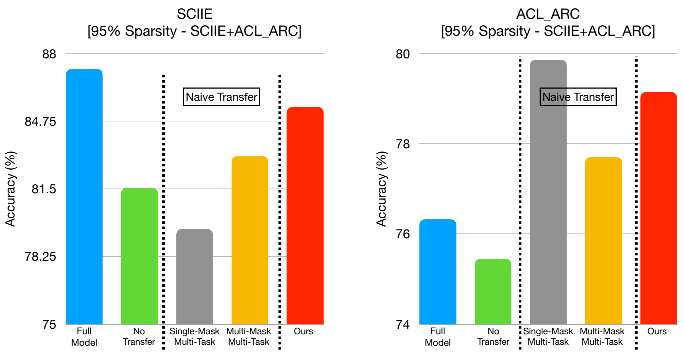

Figure 2: SCIIE and ACL_ARC performance at 95% sparsity.

4.2 What should you transfer?

We perform an ablation at 95% sparsity to determine what is most important to transfer:
Weights Only: We learn model weights and structural mask for the transfer task only. We then generate a random structural mask at the appropriate sparsity level (95%) and extract the model weights corresponding to this mask from the model trained on the transfer task. We then fine-tune this smaller, pruned model on the target task.
Masks Only: We learn model weights and structural mask for the transfer task only. We then reset the model weights to the pre-trained (not-yet-finetuned) state. Given the learned mask from the transfer task, and the untuned model weights, we then fine-tune this pruned model on the target task.
Masks and Weights: We use the transfer task to learn both the model weights and structural mask. We take weights and masks of this small model and fine-tune it on the target task.

Table 1: It is most beneficial to transfer both the learned weights and structural variables (masks)

	Metric	No Transfer	Weights Only	Structure Only	Both
STSB $\rightarrow$ MRPC	Accuracy %	79.2	68.4 ($\downarrow$)	76.96 ($\downarrow$)	79.7 ($\uparrow$)
MRPC $\rightarrow$ STSB	Pearson C.	0.868	0.23 ($\downarrow$)	0.8527 ($\downarrow$)	0.871 ($\uparrow$)

Note that we do hyper-parameter search for these experiments in the space given by Table 4 and we report the best results for each approach in Table 1. In all these experiments, we do not see the target task until we start fine-tuning the pruned model.

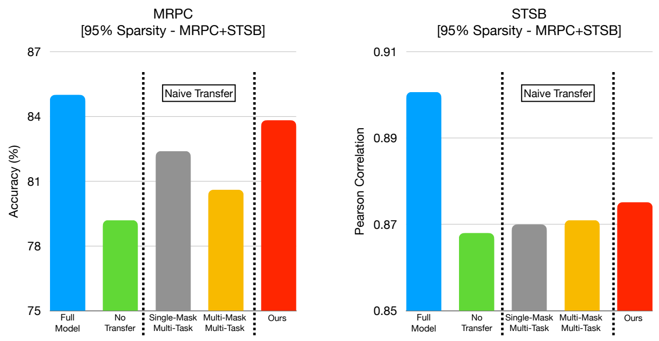

Figure 3: STSB and MRPC performance 95% sparsity.

4.3 When should you transfer?

We ablate at 95% sparsity to determine what to introduce the transfer task (T) :

Prune(T) $\rightarrow$ **FT**(P): We do structural pruning to learn both the weights and structure for a small model using the transfer task, T. We then fine-tune (FT) the pruned model on the primary task (P) only.
Prune(P) $\rightarrow$ **FT**(P, T): We learn both the weights and structure for a small model using the primary task, P. We then fine-tune the pruned model on both (T) and (P).
Prune(P, T) $\rightarrow$ **FT**(P): We learn both the weights and structure for a small model using both the transfer and primary task. We use our δ-formulation for this joint learning stage. We then fine-tune (FT) the pruned model on the primary task (P) only.

Table 2: We find that it is optimal to introduce the primary task early.

	Metric	No Transfer	Prune(T) $\rightarrow FT$(P)	Prune(P) $\rightarrow FT$(T, P)	Prune(P,T) $\rightarrow FT$(P)
STSB $\rightarrow$ MRPC	Accuracy %	79.2	79.7 ($\uparrow$)	83.09 ($\uparrow$)	83.82 ($\uparrow$)
MRPC $\rightarrow$ STSB	Pearson C.	0.868	0.871 ($\uparrow$)	0.861 ($\downarrow$)	0.8751 ($\uparrow$)

4.4 Does the learned structured sparsity translate to hardware speedups?

Taking SCIIE as our primary task and ACL-ARC as the transfer task, we explore the accuracy-speedup tradeoff that is induced by leveraging transfer learning for structured pruning. We vary the degree of compression from 40% sparsity to 98%. Figure 4 summarises our findings. For SCIIE+ACL, we fix the task weighting to the best performing configuration from our 95% sparsity experiments, the rest of the hyper-parameters are cross-validated from values in Table 4. At 50$\times$ compression (95%) sparsity, we are able to obtain a $\sim$ 5% boost in accuracy over not using a transfer task, whilst achieving a $\sim$ 10$\times$ speedup in inference.

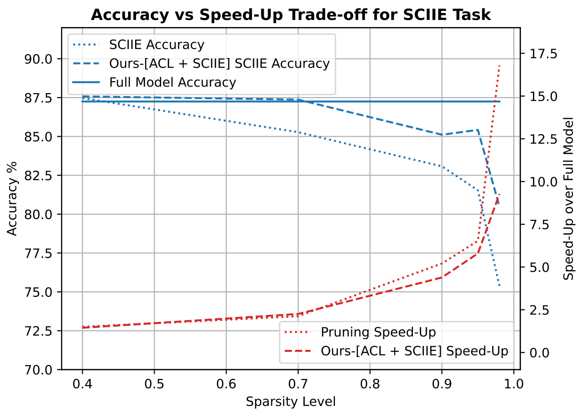

Figure 4: Leveraging a transfer task via our δ-formulation, gives accuracy boosts on SCIIE at varying levels of compression.

5 Conclusion

We presented the challenge of structured pruning under limited data. Our work provides various prescriptions to practitioners for effectively leveraging transfer learning as a remedy. For future work, it would be interesting to investigate robust ways of choosing the transfer task.

References

[1] Orevaoghene Ahia, Julia Kreutzer, and Sara Hooker. The low-resource double bind: An empirical study of pruning for low-resource machine translation. *arXiv preprint arXiv:2110.03036*, 2021.

[2] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *Proceedings of machine learning and systems*, 2:129–146, 2020.

[3] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[4] Lucio M Dery, Paul Michel, Mikhail Khodak, Graham Neubig, and Ameet Talwalkar. Aang: Automating auxiliary learning. *arXiv preprint arXiv:2205.14082*, 2022.

[5] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

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

[7] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010.

[8] Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*, 2019.

[9] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[10] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

[11] David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406, 2018.

[12] Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through l_0 regularization, 2018.

[13] Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. *arXiv preprint arXiv:1808.09602*, 2018.

[14] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

[15] Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389, 2020.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[17] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*, 2019.

[18] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[19] Ziheng Wang, Jeremy Wohlwend, and Tao Lei. Structured pruning of large language models. *arXiv preprint arXiv:1910.04732*, 2019.

[20] Mengzhou Xia, Zexuan Zhong, and Danqi Chen. Structured pruning learns compact and accurate models. *arXiv preprint arXiv:2204.00408*, 2022.

A CoFi Degradation with Data Scarcity

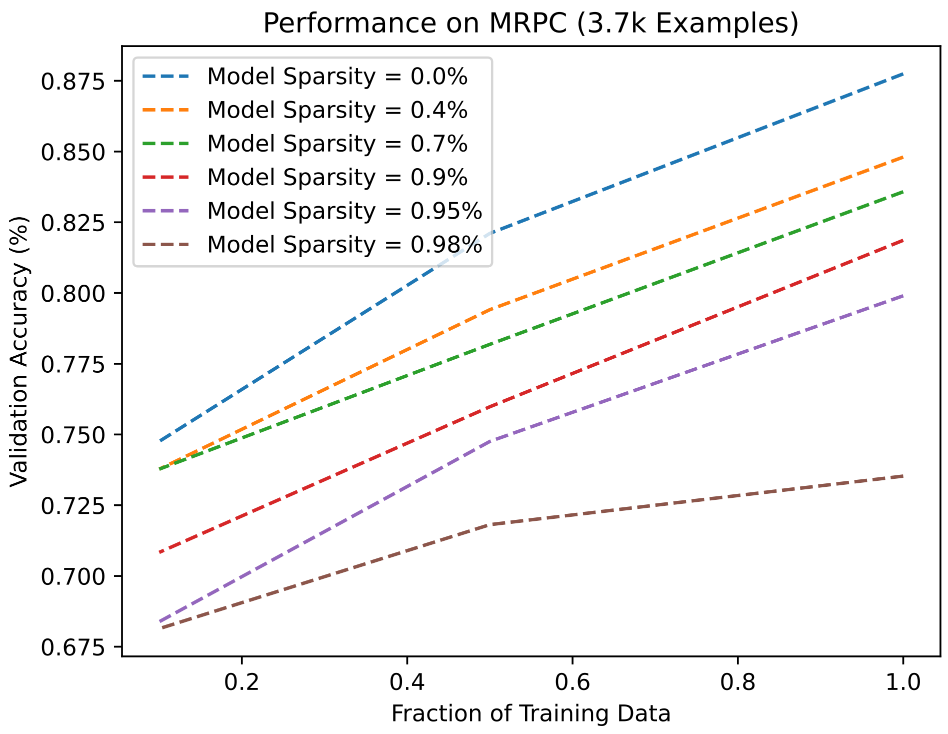

Figure 5: Performance degradation of CoFi [20], a SOTA structured pruning algorithm, with varying data sizes.

B Experimental Details

B.1 Dataset Information

Table 3: Specifications of datasets used to evaluate our methods.

Domain	Task	Task-Type	Train Size	Metric
CS	SCIIE [13]	Classification	3219	Accuracy
	ACL-ARC [11]	Classification	1688	Accuracy
GLUE	STSB [18]	Sentence Similarity	7K	Pearson's Correlation
	MRPC [18]	Paraphrase Detection	3.7K	Accuracy

B.2 Training Parameters

We follow as closely as possible the hyper-parameters that are used in the original CoFi code base. For CoFi specific hyper-parameter settings please see AppendixC During pruning, we perform 10K gradient descent steps to learn both the structural and parameter variables of the model. We perform 20 epochs of post-pruning finetuning on the target task.

Table 4: Hyper-parameter choices

Hyper-parameter	Values	Description
Task pair weightings	(1, 1), (1, 2), (2, 1)	Weightings applied to transfer task vs target task during training.
Model LR - Pruning	1e-4, 2e-5	Learning rate used for model parameters during pruning.
Model LR - Finetuning	1e-4, 2e-5	Learning rate used for finetuning pruned model.
Structure LR	0.1, 0.01	Learning rate used for learning structural parameters.
δ-l_2 Reg Weight	1e-2	Regularization weight used in δ-formulation.

C CoFi-specific Details

We turn off output prediction distillation for all experiments. ie – we do not distill the predictions from the pre-trained models since unlike in the original CoFi paper, we are not starting from a model that has already been fine-tuned on the target task but rather we are starting from the pre-trained model itself.

D What impacts the quality of transfer ?

Table 5 contains experimental results highlighting our investigation of different variables that can impact the quality of a transfer task.

Target	Full BERT	No Transfer	Domain	Resourced-ness	Transfer Task	Performance
MRPC	83.48	79.2	In-Domain	High (364k)	QQP	**85.78**
			In-Domain	Low (7k)	STSB	83.82
			Out-of-Domain	High (180k)	RCT	85.29
STSB	0.901	0.868	In-Domain	High (364k)	QQP	**0.877**
			In-Domain	Low (3.7k)	MRPC	0.875
			Out-of-Domain	High (180k)	RCT	0.873

Table 5: Using a high-resource, in-domain transfer task leads to best transfer performance. For all experiments, best results from hyper-parameter search are reported. All models (apart from Full BERT) are pruned to 98% sparsity.

E What are the structural differences between a pruned model using transfer learning and without ?

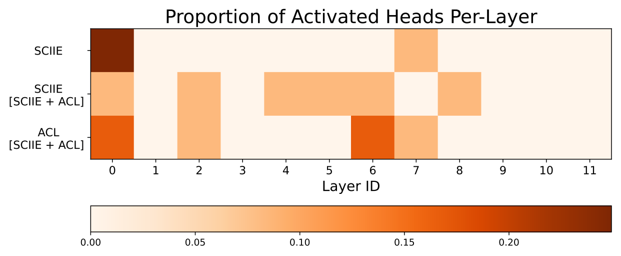

Figure 6: Structural visualization at 98% sparsity. Qualitatively, using a transfer task changes the pruned model structure significantly. The ACL transfer task in this case induces the learned SCIIE structure to be more diffuse across the layers of the model.

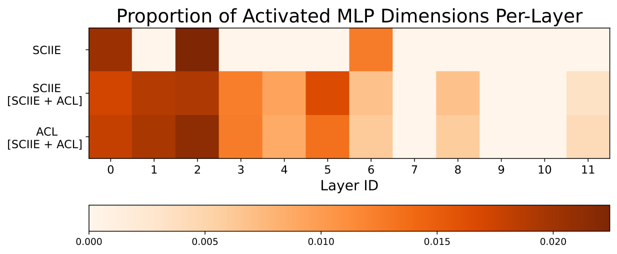

Figure 7: Structural visualization at 98% sparsity. Qualitatively, using a transfer task changes the pruned model structure significantly. The ACL transfer task in this case induces the learned SCIIE structure to be more diffuse across the layers of the model.