The Emergence of Essential Sparsity in Large Pre-trained Models: The Weights that Matter

Ajay Jaiswal[1], **Shiwei Liu**[1,2], **Tianlong Chen**[1], **Zhangyang Wang**[1]
[1]University of Texas at Austin, [2]Eindhoven University of Technology
{ajayjaiswal, shiwei.liu, tianlong.chen, atlaswang}@utexas.edu

Abstract

Large pre-trained transformers are *show-stealer* in modern-day deep learning, and it becomes crucial to comprehend the parsimonious patterns that exist within them as they grow in scale. With exploding parameter counts, Lottery Ticket Hypothesis (LTH) and its variants, have lost their pragmatism in sparsifying them due to high computation and memory bottleneck of repetitive *train-prune-retrain* routine of iterative magnitude pruning (IMP) which worsens with increasing model size. This paper comprehensively studies *induced sparse patterns* across multiple large pre-trained vision and language transformers. We propose the existence of – **"essential sparsity"** defined with a **sharp dropping point** beyond which the performance declines much faster w.r.t the rise of sparsity level, when we directly remove weights with the smallest magnitudes in **one-shot without re-training**. We also find essential sparsity to hold valid for `N:M` **sparsity patterns** as well as on **modern-scale large language models** (`Vicuna-7B/13B`). We also present an intriguing emerging phenomenon of **abrupt sparsification** during the pre-training of BERT, i.e., BERT suddenly becomes heavily sparse in pre-training after certain iterations. Moreover, our observations also indicate a **counter-intuitive** finding that BERT trained with a larger amount of pre-training data tends to have a better ability to condense knowledge in comparatively relatively fewer parameters. Lastly, we investigate the effect of the pre-training loss on essential sparsity and discover that self-supervised learning (SSL) objectives trigger stronger emergent sparsification properties than supervised learning (SL). Our codes are available at `https://github.com/VITA-Group/essential_sparsity`.

1 Introduction

Enormous increases in scale often permeate systems with unique new behavior. Transformers [1], swiftly after their introduction are scaling every day, dramatically improving the state-of-the-art performance on a wide array of real-world computer vision [2, 3, 4, 5, 6, 7], natural language processing [8, 9, 10, 11, 12, 13, 14, 15, 16] applications and leaderboards. With the astonishing explosion of parameter counts (millions to billions) in the past few years, while chasing performance gains, fine-tuning these large pre-trained models with non-industry standard hardware is becoming seemingly impossible, in addition to expensive inference and steep environmental cost. For instance, GPT-3 [17] contains 175 billion parameters and requires at least five 80GB A100 GPUs for inference [18].

In the hustle of building gigantic models, a parallel and growing field of model compression has been exploring the prospects to compress these gigantic models at the cost of marginal/no sacrifice in performance. Among many efforts for compressing models and accelerating inference, network pruning [19, 20, 21, 22, 23, 24, 25, 26], which removes the least important components from the model, varying from the low granularity (e.g., individual weights) to the high granularity (such as blocks, filters, and attention heads), stands out as one of the most effective techniques. Despite the enormous success of the Lottery Ticket Hypothesis (LTH) [27] and its variants for small-scale neural

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

networks, there still exists a large **gap** in its practical adoption for large pre0trained transformers with billions of parameters because of the fully dense training routine of iterative magnitude pruning (IMP), which exacerbates with an increase in model capacity. Recently, many works [28, 29, 30, 20] are investigating strategies to make IMP pragmatic for large pre-trained models. However, they still rely on train-prune-retrain routine of IMP and downstream tasks to identify subnetworks. While chasing for gold-standard sparsity masks with computationally expensive approaches, it is critically important to understand pre-existing *high-quality sparse patterns* in these large transformers.

In our work, we reveal and study an intriguing, previously overlooked property of large pre-trained transformers – **"essential sparsity"**. In plain words (formal definition in Sec. 3), we define essential sparsity as ***sharp dropping point***, beyond which the fine-tuning performance after one-shot pruning declines much faster w.r.t. the sparsity level rise. We directly remove weights with the smallest magnitudes in **one-shot** from the pre-trained checkpoints, thereby the identified subnetwork mask requires no extra computational overhead to spot and remains identical across all downstream tasks.

Essential sparsity is an important yet understudied induced property of large pre-trained transformers, indicating that at any time, *a significant proportion of the weights in them can be removed **free** of any calibration data or post-optimization, although the proportion may vary depending on the complexity of the downstream task.* One important practical implication conveyed by this observation, as overlooked by prior work, is summarized as: *within the sparsity range induced by "essential sparsity", the simplest possible pruning technique as aforementioned performs the same well as any fancy technique such as LTH, and even their identified sparse masks are highly similar.*

Our work, for the **first** time, conducts a comprehensive study of the existence of essential sparsity across multiple vision and language transformers with varying scales and training strategies. Besides, we observe essential sparsity to **hold valid for various `N:M` sparsity patterns** with hardware speedup potentials. In addition, our experiments using the popular `MMLU` benchmark [31] on `Vicuna-7B/13B` illustrate essential sparsity observations remain **true for modern-scale large language models (LLMs)**, indicating the existence of high-quality sparse subnetworks within dense pre-trained checkpoint.

Next, we study the emergence of those sparsification properties during the pre-training dynamics, using BERT as the focused subject of study. We found an intriguing phenomenon of **abrupt sparsification**, i.e., BERT suddenly becomes heavily sparse after certain number of training iterations. As we vary the pre-training dataset size. our observation indicates another **counter-intuitive** finding that BERT trained with a *larger amount of pre-training data tend to have a better ability to condense knowledge in relatively fewer parameters*. We also dive deep into the effect of the pre-training loss on essential sparsity and discover that self-supervised learning (SSL) objectives trigger stronger emergent sparsification properties than supervised learning (SL).

Key contributions for our work can be summarized as:

- We found the **ubiquitous existence of essential sparsity** across large pre-trained transformer models of varying scales for vision and language, irrespective of the training strategy used for pre-training them. High-quality sparse masks comparable to lottery tickets [27] within the essential sparsity range can be spotted *free of inference or post-optimization* by merely selecting the lowest magnitude weights, that requires no expensive repetitive train-prune-retrain routine. The observation holds true for both unstructured and `N:M` sparsity patterns.
- Our comparison of the sparse masks obtained by selecting the lowest magnitude weights with the lottery tickets within the essential sparsity range, surprisingly unveils a **notably high cosine similarity** (>98%) across various downstream tasks from NLP and CV. This observation sends a strong message that in large pre-trained models, LTH perhaps enjoys little additional privilege despite utilizing enormous computational costs.
- While studying the emergence of sparsification properties during the pre-training dynamics in BERTs, we found an intriguing phenomenon of **abrupt sparsification**, i.e., BERT suddenly becomes heavily sparse after certain number of training iterations. We additionally found a **counter-intuitive** observation that BERT pre-trained with larger amount of data tends to be more sparser, i.e., they become better at *knowledge abstraction with fewer parameters*.
- We additionally study the effect of the pre-training loss on the emerging sparsity of FMs. When we switch between supervised learning (SL) and self-supervised learning (SSL) objectives on ViT, we observed that **SSL tends to have better emergent sparsification**

properties, thereby more friendly to pruning. We further provide layer-wise visualization to understand what the sparsity learned by SSL vs SL looks like.

2 Related Work

Sparse Neural Networks (SNNs). Sparsity in deep neural networks is usually introduced by model compression [32, 33] which removes the redundant parameters. Based on the type of sparsity patterns, it can be categorized into two families: (i) *unstructured sparsity* [33, 34, 35] where the non-zero elements are irregularly distributed; (ii) *structured sparsity* [36, 37, 38] where a group of parameters is eliminated like a convolutional kernel in convolutional neural networks or an attention head in the transformer. In general, the former sparsity pattern obtains a better performance thanks to its flexibility, while the latter sparse pattern tends to be more hardware friendly. Many important SNN works start by studying the former and then turn to the latter as a special case.

Meantime, according to the timing to perform dimensionality reduction, sparsity can be obtained in the post-training, during-training, and prior-training of deep neural networks. (i) *Post-Training*. To pursue inference time efficiency, trained models can be pruned dramatically with marginal loss of performance with respect to certain heuristics, including zero-order [33], first-order [39, 40, 41], and second-order [32, 42, 43] criteria. Among these algorithms, the weight magnitude-based approach (*e.g.* iterative magnitude pruning) is a popular option. Especially, it is frequently adopted by the Lottery Ticket Hypothesis [19] to produce sparse NNs with undamaged trainability and expressiveness. (ii) *During-Training*. On the contrary to pruning a well-trained model for inference efficiency, during-training sparsification [44] also enjoys computational savings for model training. It normally first optimizes a dense network for several iterations, then gradually sparsifies it with a pre-defined schedule, and finally leads to lightweight sparse NNs. For example, [45, 46, 47] leverage ℓ_0 or ℓ_1 penalties to encourage weight sparsity during the network training, hoping to zero out unimportant parameters. [48, 49] cast it as an optimization problem with reparameterization and bi-level forms, respectively. Another fashion is dynamic sparsity exploration [50, 51, 52, 53, 54, 55, 56, 57, 58, 59] which allows pruned connections can be re-activated in the latter training stage. (iii) *Prior-Training*. Identifying the critical sparsity patterns at the initialization is one exciting rising sub-field. It determines the sparse connectivities in a very early stage, like one iteration [25, 60] or a few iterations [61, 62]. In this paper, we mainly investigate post-training unstructured SNNs.

Sparsity in Pre-Trained Transformers. Pre-trained Transformers have become *de facto* choice for numerous applications of natural language processing (NLP) [8, 9, 10, 15, 16] and computer vision [2, 7, 63]. Their impressive performance is partially credited to their tremendous learning capacity empowered by huge amounts of model parameters. Unfortunately, such successes come with burning thousands of GPUs/TPUs for thousands of hours [64, 17], even for a single round of model training. To address this resource-intensive concern and improve the affordability of these transformers, plenty of pioneering researchers devote themselves in this area [40, 20, 30, 65, 66, 67, 68, 69, 70]. Rather than proposing a new sparsification method, this paper reveals the blessings of induced essential sparsity during the pre-training and how we can capitalize it to prune large pre-trained models without any computational overhead.

3 Experimental Settings

Network and Pre-Trained Transformers. We consider {$\texttt{BERT}_{\texttt{Base}}$ [64], $\texttt{BERT}_{\texttt{Large}}$ [64], $\texttt{OPT}_{\texttt{125M}}$ [71], $\texttt{OPT}_{\texttt{350M}}$ [71], and $\texttt{OPT}_{\texttt{1.3B}}$ [71]} and {$\texttt{ViT}_{\texttt{Base}}$ [2], $\texttt{ViT}_{\texttt{Large}}$ [2], and $\texttt{DiNO}_{\texttt{Base}}$ [63] for NLP and CV applications respectively. Their officially pre-trained models[1] are adopted in our experiments. To be specific, let $f(x;\theta_p)$ be the output of a transformer with pre-trained model parameters θ_p and input sample x.

Datasets, Training, and Evaluation. For downstream tasks in NLP, we consider {MNLI, QNLI, QQP, SST-2} from GLUE [13], RTE from SuperGLUE [72], and SQuAD v1.1 [73]. As for vision downstream applications, we examine {CIFAR-10, CIFAR-100, Tiny-ImageNet [74]}. We also use the Arithmetic Reasoning task from the recently proposed SMC-Benchmark [75] and consider popular math word problem datasets: (1) the widely used MAWPS benchmark [76] composed of 2,373 problems; (2) the arithmetic subset of ASDiv-A [77] - with 1,218 math problems; (3) the more

[1] Check supplementary materials for pre-trained model details.

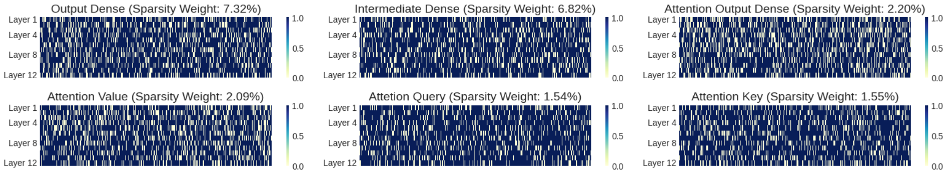

Figure 1: Naturally induced sparsity patterns of `bert-base-uncased` across the components of transformer blocks. The pre-trained model is pruned by 21.50% using one-shot-magnitude pruning. Yellow dots indicate the location of pruned low-magnitude weights.

Table 1: Downstream tasks fine-tuning details. Learning rate decay linearly from initial value to 0.

Settings	Natural Language Processing						Computer Vision			
	MNLI	QNLI	QQP	RTE	SST-2	SQuAD v1.1	CIFAR-10	CIFAR-100	Fashion-MNIST	Tiny-ImageNet
# Training Ex	392,704	104,768	363,872	2,496	67,360	88,656	45,000	45,000	55,000	90,000
# Epoch	3	4	3	5	5	3	8	8	8	5
Batch Size	32	32	32	32	32	16	64	64	64	64
Learning Rate	$2e-5$	$2e-5$	$2e-5$	$2e-5$	$2e-5$	$3e-5$	$2e-5$	$2e-5$	$2e-5$	$2e-5$
Optimizer	AdamW with decay (α) = 1×10^{-8}						AdamW with decay (α) = 2×10^{-8}			
Eval. Metric	Matched Acc.	Accuracy				F1-score	Accuracy (Top-1)			

challenging SVAMP [78] dataset which is created by applying complex types of variations to the samples from ASDiv-A. The task difficulty monotonically increases from MAWPS to ASDiv-A, and to SVAMP. We adopt the default dataset split for training and evaluation for our downstream application. More detailed configurations are collected in Table 1. For SMC-benchmark, we strictly followed the settings proposed in the official implementation[2]

Sparse Neural Networks (SNNs). The weights of a SNN can be depicted as $m \odot \theta$, where $m \in \{0,1\}^{|\theta|}$ is a binary mask with the same dimensionality as θ and $\odot$ is the elment-wise product. Let $\mathcal{E}^{\mathcal{T}}(f(x;\theta))$ denotes the evaluation function of model outputs $f(x;\theta)$ on the corresponding task $\mathcal{T}$ (which might involve fine-tuning). $\mathcal{P}_\rho(\cdot)$ is the sparsification algorithm which turns a portion ρ of "1" elements in the sparse mask m into "0"s. Here is a formal definition of **Essential Sparsity** below.

Essential Sparsity. If $\mathcal{E}^{\mathcal{T}}(f(x;m \odot \theta)) \geq \mathcal{E}^{\mathcal{T}}(f(x;\theta)) - \epsilon$, and $\mathcal{E}^{\mathcal{T}}(f(x;\mathcal{P}_\rho(m) \odot \theta)) < \mathcal{E}^{\mathcal{T}}(f(x;\theta)) - \epsilon$ where the value of ρ and ϵ are small. Then, the according sparsity $1 - \frac{\|m\|_0}{|m|}$ is named as Essential Sparsity for the model f on task $\mathcal{T}$.

As detailed above, the model at the essential sparsity usually has a turning point performance, which means further pruning even a small portion ρ of weights leads to at least ϵ performance drop, compared to its dense counterpart $\mathcal{E}^{\mathcal{T}}(f(x,\theta))$. In our case, ϵ is set as 1%. In plain language, the turning point of essential sparsity defines a *sparsity range* with two characteristics: (i) *within this range*, the one-shot pruned model performs as well as or better than dense baseline, without re-training; (ii) *beyond this range*, a notable accuracy drop becomes observable after one-shot magnitude pruning.

Pruning Methods. To find the desired sparse mask m, we use two classic weight magnitude pruning algorithms [33, 19]. *One-shot Magnitude Pruning* (OMP): we directly eliminate a pre-defined portion of parameters from θ_p with the least absolute magnitude. `Lottery Tickets Pruning` (LTP): (i) we first train the model to converge on a downstream task $\mathcal{T}$; (ii) then remove a portion of the smallest weights and reset the remaining weight to their initialized value from pre-training θ_p; (iii) such processes will be repeated until reaching the desired sparsity.

Two crucial facts are noteworthy. First, starting from the pre-trained model, unlike LTP which seeks downstream-specific masks with additional training, the OMP sparse masks require **no additional training** to identify and are **agnostic to downstream applications**. Since they are estimated directly from the pre-trained weight checkpoint, the OMP sparse mask remains the same for all downstream applications (we only continue to fine-tune the non-zero weights with the same mask, over different downstream datasets). Second, for the same above reason, a higher-sparsity OMP mask is naturally **nested** within a lower-sparsity mask, as long as they are pruned from the same pre-trained checkpoint using OMP. The same nested property does not hold for LTP-obtained masks.

[2] `SMC-Benchmark` fine-tuning setting details: Arithmetic Reasoning

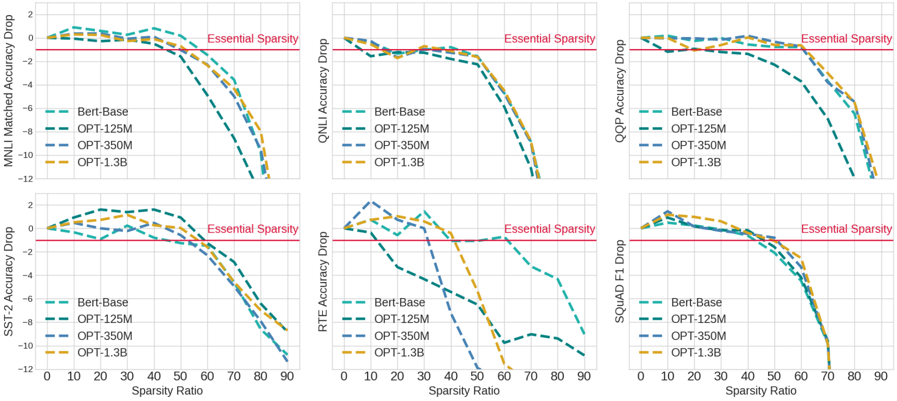

Figure 2: Fine-tuning performance drop estimated with respect to dense counterpart for various downstream tasks of NLP pre-trained models (`bert-base, OPT-125m, OPT-350m, OPT-1.3B`). Note that for fair evaluation, we have used exactly same fine-tuning settings across all pruning ratios.

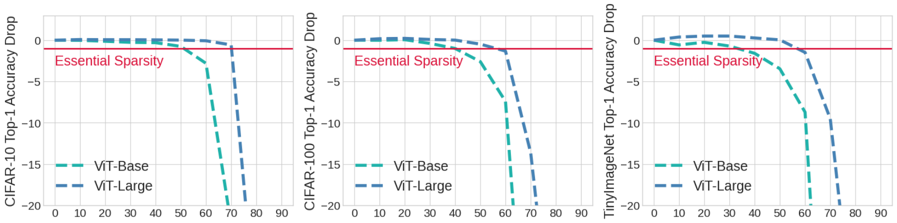

Figure 3: Fine-tuning performance drop estimated with respect to dense counterpart for various downstream tasks of CV pre-trained models (`ViT-base` & `ViT-large`).

4 Essential Sparsity Exists in Pre-trained Language and Vision Models

Revisiting sparsity in Pre-trained Transformers: The unconstrained growth in parameters has resulted in significant computational costs and excessive memory demands. The trend undoubtedly continues with transformers on the forefront, where more and more layers are stacked with dense attention blocks (eg. GPT-3 has surprisingly 175 billion parameters) necessitating substantial computational resources and extended training or fine-tuning durations. Unlike extensive efforts explored in the past for pruning ResNets and related families of networks, pruning large-scale pre-trained transformer models hasn't received enough attention.

Lottery Ticket Hypothesis (LTH) based approaches have been recently explored to prune BERT-base size pre-trained models. However, they lose their pragmatism due to the expensive dense train-prune-retrain routine of IMP and non-transferable characteristics of identified subnetworks across similar tasks [20]. oBERT [66] proposes second-order to BERT-level scale by allowing for pruning blocks of weights. Note that, although these methods provide interesting approaches to prune BERT scale transformers, they *fail to sufficiently recognize the strength of pre-existing, more easily accessible high-quality sparse patterns* in pre-trained large models invariant of the scale, training data modality, and strategy. This paper's primary goal is to bring the sparse community's attention towards the strength of ubiquitously induced sparse patterns during the pre-training of large transformers and encourage them to effectively capitalize it during the design of more sophisticated pruning algorithms.

Essential Sparsity: As defined in Section 3, essential sparsity indicates the presence of naturally induced sparse structures in large pre-trained models with respect to the lowest magnitude weights. Figure 1 represents the induced sparsity distribution of `bert-base-uncased` pre-trained model with 21.50% close to zero weights, which we have been able to remove without **ANY** drop of performance across any of our evaluation downstream datasets. This sought a strong message about the existence of free, universal, and available to consume sparse mask $(m \cdot \theta_p)$ without any computational overhead induced by sophisticated pruning algorithms. In addition, a closer observation of the distributed sparsity ratios across different modules of the transformer block indicates that the majority of the low-magnitude weights are concentrated in the dense feed-forward layers. Such findings conform with the success of Mixture-of-Experts [79] and provide cues for future development of pruning algorithms to exploit the over-parameterization of dense feed-forward layers.

We now enlist our key findings related to essential sparsity in large pre-trained transformer models:

- In all vision and language models, we find the existence of essential sparsity and the sharp turning point of the sparsity-performance curve.
- The sharp turning point of essential sparsity is downstream task-dependent and can vary depending on the task complexity.
- The sharpness behavior of essential sparsity is more profound in vision pre-trained models in comparison with language models.
- Essential sparsity holds valid for recently proposed pruning benchmark SMC-bench [75].
- Essential sparsity holds valid for `N:M` structured sparsity patterns [80] with potential speedup.
- Self-supervised learning objective triggers stronger emergent sparsification properties than supervised learning in models having exactly the same architecture.

Universal Existence of Essential Sparsity: We comprehensively evaluate the extent to which essential sparsity exists in the large pre-trained language models (CV and NLP) with a standard pre-trained initialization θ_p. For NLP, we studied `bert-base, OPT-125M, OPT-350M, OPT-1.3B` and used `MNLI, QNLI, QQP, SST-2, RTE, and SQuAD1.1` to estimate the effect the removing low-magnitude weights on downstream performance. On the other hand, for CV, we rely on `ViT-Base` and `ViT-Large` models and use `CIFAR-10, CIFAR-100, TinyImageNet` datasets. Figure 2 illustrate the performance drop (y-axis) of various NLP downstream task fine-tuning with mask ($m \cdot \theta_p^{x\%}$), when we remove $x\%$ of lowest magnitude weights. Similarly, Figure 3 illustrates the effect of pruning $x\%$ low-magnitude weights on the fine-tuning performance of CV downstream tasks. Moreover, Figure 4 presents the existence of essential sparsity for a recently proposed sparsity benchmark, named SMC-Bench [75]. It is interesting to observe that essential sparsity exists in all CV and NLP pre-trained models, and $\sim 30-50\%$ of weights can be removed at free without any significant drop in performance. Note that these masks are identified prior to fine-tuning, on the pre-trained weights and thereby remain the same for all the downstream tasks, indicating no requirement for bookkeeping LTH-type task-specific masks. It is also important to appreciate the significance of removing $\sim 40\%$ (which translates to > 500 million parameters) of OPT-1.3B at no cost without any significant performance drop.

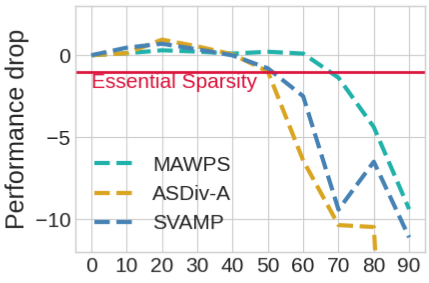

Figure 4: Fine-tuning performance drop of `bert-base` on Arithmetic Reasoning datasets in the SMC-benchmark [75].

Eseential Sparsity for Structured `N:M` Sparse Patterns: Considering the demand for practical speedup, we also conduct evaluations to understand the essential sparsity for the hardware-friendly structured `N:M` sparsity. A neural network with `N:M` sparsity satisfies that, in each group of M consecutive weights of the network, there are at most N weights have non-zero values. We studied `OPT-350M` and used `MNLI, QNLI, RTE, SST-2` to estimate the effect of removing low-magnitude weights in structured `N:M` fashion following [80]. Figure 5 illustrates the performance drop (y-axis) of `OPT-350M` on various downstream datasets with multiple `N:M` sparse mask ($m \cdot \theta_p^{x\%}$). It can be in general observed essential sparsity holds valid for structured `N:M` sparsity patterns, and a large fraction of low-magnitude weights can be removed at free without significant drop in downstream performance.

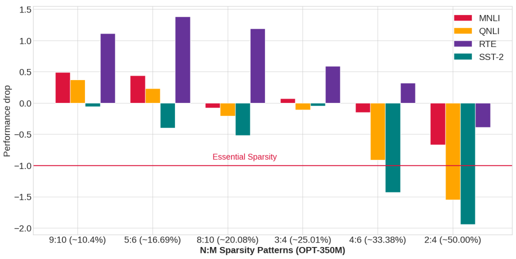

Figure 5: Fine-tuning performance drop of `OPT-350M` at GLUE dataset wrt. dense counterpart on multiple `N:M` sparsity patterns [80] masks.

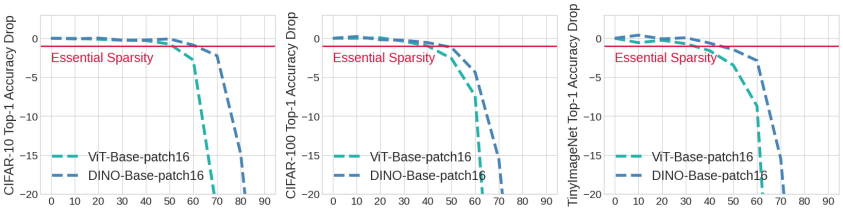

Figure 6: Essential Sparsity and performance comparison `ViT-base` and `DINO-base` which share the same architecture but pre-trained using supervised (SL) and self-supervised learning (SSL) objectives. It can be observed that the SSL induces a better sparsification ability in the pre-trained checkpoint.

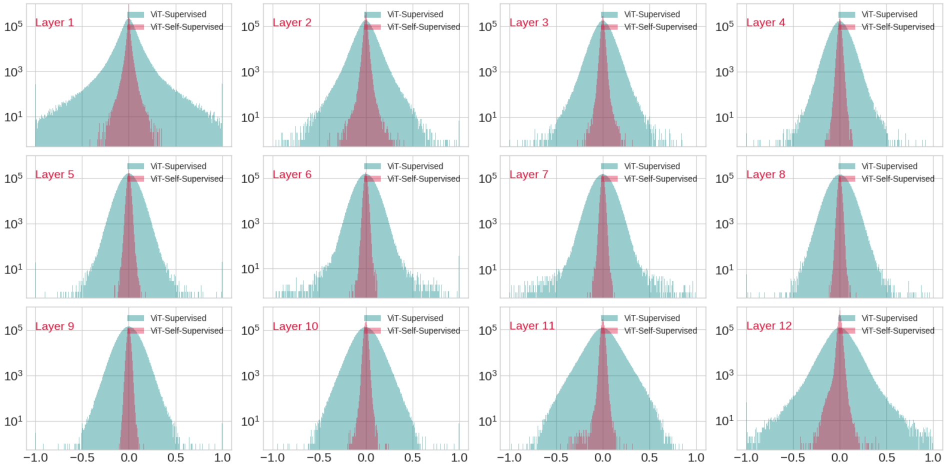

Figure 7: Layer-wise weight distribution of `ViT-base` and `DINO-base` trained using supervised and self-supervised learning objective. Note that the weights of both pre-trained models are normalized using `sklearn` for fair comparison. Additionally, `DINO` has 14.37% more zero weights than ViT.

Essential Sparsity and Sharp Turning Behaviour: In this section, we attempt to bring attention towards the sharp turning point behavior of essential sparsity. Across all our vision and language models and related downstream tasks (Figure 2, 3, & 4), we observed that after a certain removal of a certain proportion of low-magnitude weights, the downstream fine-tuning ability of pre-trained models **sharply** drops. This is a clear indication that the pre-training knowledge resides in a fraction of high-magnitude weight regions and if we our one-shot pruning touches that region, it non-linearly impacts the transfer learning ability of the pre-trained checkpoint. Our experiments reveal that this *sharp-turning behavior is not dependent on model size but on the downstream task*. Larger model doesn't implicate that it can be pruned for a higher proportion of low-magnitude weight, without observing the sharp drop. For example, `bert-base` observes the sharp turning point at around 60% sparsity while OPT-1.3B can not be pruned beyond 40% without observing the sharp performance drop on `RTE` task, although it has ~ 10 times more parameters than `bert-base`. Also, it is interesting to observe that although `OPT-125M` and `bert-base` have similar performance count, `bert-base` illustrate more friendly behavior to pruning, and can be pruned to comparatively higher sparsity ratio than `OPT-125M` on all our evaluation downstream tasks.

Influence of Supervised versus Self-supervised Learning Objectives: With the recent success of self-supervised learning (SSL) in pre-training large transformer models and its ability to scale to enormous internet-size data, it is interesting to understand how SSL learning objectives favor sparsity. Due to the unavailability of supervised datasets to pre-train BERT-scale models, we switch to Vision transformers, and use `ViT-base` and its self-supervised version `DINO-base` which inherit exactly the same architecture. Figure 6 illustrates fine-tuning performance drop of `ViT-base` and `DINO` with an increase in the proportion of low-magnitude weight removal from pre-trained checkpoints. Across all tasks, fine-tuning performance doesn't suffer any drop till $30-40\%$ weights are removed.

One interesting observation is that `DINO-base` tends to be more robust to low-weight removal, and has comparatively better essential sparsity across all tasks. To further investigate why we can prune

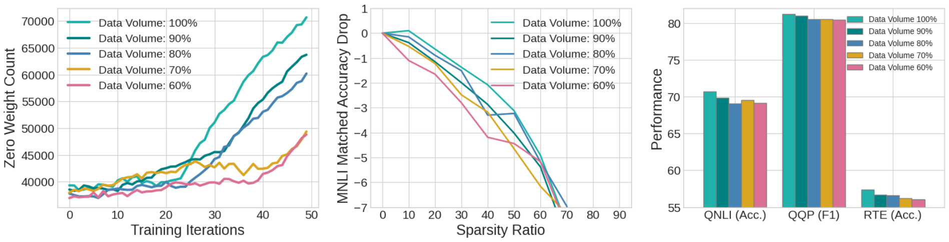

Figure 8: Plot Description in order left-right. (i) Zero-weight count of 5 pre-training experiments of `Bert-base` using `bookcorpus` dataset from HuggingFace with varying percentages of randomly selected data volume with exactly the same pre-training setting. (ii) Downstream performance of the pre-trained `Bert-base` models with varying data volume across different sparsity ratios on `MNLI`. (iii) Downstream performance of 5 dense pre-trained models on `QNLI, QQP, RTE`.

`DINO-base` more than `ViT-base`, we examine the weight magnitude distribution across the layers of the transformer blocks from the pre-trained checkpoints. Figure 7 presents the normalized layer-wise weight distribution of `DINO-base` and `ViT-base`. It can be clearly observed that SSL-based `DINO` tends to have a weight distribution more friendly to pruning with a significantly large amount of weights concentrated around zero magnitudes. More concretely, we found that `DINO-base` have $\sim 14\%$ more zero weights than `ViT-base`, which justifies its higher essential sparsity.

5 How Essential Sparsity Emerges during the Pre-Training Dynamics

Scaling the volume of pre-training data volume is widely believed to favor transfer performance in many downstream tasks in a desirable way [81]. Although, this relationship has been extensively studied recently in many works [81, 82, 83, 84], pre-training data volume role in inducing sparse properties in the large transformers is still **unexplored**. In this section, we ask an important question: *How does the volume of pre-training data impact the emerging sparse patterns in large transformers, and if it improvises their prunability?*

To this end, we designed custom experiments to pre-train `bert-base` from scratch using HuggingFace `bookcorpus` with a vocabulary size of 30,522. We created 5 different pre-training datasets by randomly selecting $100\%, 90\%, 80\%, 70\%, 60\%$ of the training samples from `bookcorpus` and pre-train for 50k iteration each to ensure that all models receive the same amount of gradients. Note that we maintain exactly same training settings for all models to retain fair comparison and save the pre-training checkpoints every 1000 iterations. We now enlist our key findings related to pre-training data volume and induced sparse patterns in transformers:

- We observe an interesting new phenomenon of **abrupt sparsification**, i.e., the introduction of sudden high sparsity, during pre-training `bert-base` models, regardless of data size.
- We additionally observed a counter-intuitive finding that `bert-base` trained with a larger amount of pre-training data tends to have better emergence of induced sparsity.
- Across all sparsity level, we found `bert-base` trained with a larger volume of training data, enjoys better prunability, and achieve better performance on the downstream task (`MNLI`).

Figure 8(i) illustrate the emergence of sparse patterns during pre-training of `bert-base` with different pre-training data volume. We plotted the number of zero weights that emerged in the pre-training checkpoints every 1k iterations. It can be clearly observed that in all settings, the number of zero weights suddenly grow up, indicating the models start abstracting pre-training data knowledge into fewer parameter set. For instance, we find this sharp turn around 22-25k iterations for $100\%, 90\%, 80\%$ settings while around 40k iterations for $70\%, 60\%$ settings. In addition, we found that `bert` trained with a larger amount of pre-training data tends to have better emergence of induced sparsity. We argue that `bert-base` tend to learn more generalizable features and develop the capability to abstract knowledge To further investigate how this sparsity emergence further influences the prunability of the pre-trained models, we examined their performance across multiple sparsity ratios (Figure 8(ii)) on `MNLI` and found it to be in harmony with the magnitude of induced sparsity, i.e., models with high induced sparsity patterns during pre-training tend to perform better when we remove the existing low magnitude weights and fine-tune them on downstream tasks. In dense

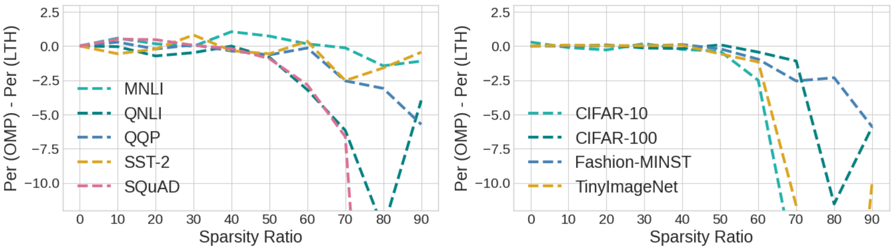

Figure 9: Performance difference comparison of fine-tuning of masks identified by LTH and OMP from `bert-base` (left) and `ViT-base` (right) across multiple downstream tasks.

settings on `QNLI, QQP, RTE` (Figure 8(iii)), we found that an increase in pre-training data volume has favorable benefits on the downstream performance with similar fine-tuning hyperparameters.

6 Essential Sparsity and Lottery Ticket Hypothesis

Lottery Ticket Hypothesis deviates from the convention of after-training pruning, and points to the existence of independently trainable sparse subnetworks from scratch that can match the performance of dense networks. Since its introduction, it has been very successful and widely adopted in compressing small-scale (eg. ResNet family) models. However, it is significantly limited in the practical adoption for large pre-trained models due to *train-prune-retrain* routine of IMP. In this section, we ask: *How effective LTH is within the essential sparsity range, and does it bring any additional privilege which can substantiate its computational cost?* Our key takeaway can be summarized as:

- Within the essential sparsity range of `bert-base` and `ViT-base`, we do not find any significant fine-tuning performance difference of the mask identified by LTH and one by removing lowest magnitude weights (OMP) across multiple downstream tasks.
- We surprisingly found the existence of high cosine similarity($> 96\%$) across the masks identified by LTH and OMP within the essential sparsity range.
- Mask similarity between LTH and OMP decreases with an increase in the sparsity ratio. This corroborates with the benefit of LTH in the high sparsity range, where LTH is able to identify task-fitting masks to retain performance due to repetitive prune and retrain routine.

Figure 9 illustrates the performance difference comparison of fine-tuning of masks identified by LTH and OMP from `bert-base` and `ViT-base` across multiple downstream tasks. It can be clearly observed that for sparsity ratio below 50%, we do not find any *significant difference in performance between expensive LTH and free-of-cost OMP pruning within the essential sparsity range*. A deeper analysis (Figure 10) across the masks from LTH and OMP unveils a surprising observation about the existence of significantly high cosine similarity across. This observation supports the findings in Figure 9 about the matching performance of LTH and OMP and convey a strong and impressive message that with the essential sparsity range, LTH doesn't provide any additional privilege. However, it is important to note that LTH is very effective in the high sparsity range beyond essential sparsity, due to its ability to find task-fitting mask using train-prune-retrain procedure but they tend to be non-transferable across different downstream tasks [20]. Moreover, it can be observed from Figure 10, even in the essential sparsity range, mask similarity across tasks by LTH is comparatively low than OMP based masks due to the strong intertwine of LTH with the downstream tasks.

7 Scaling Essential Sparsity to Modern-Scale LLMs: A Case Study

Large Language Models (LLMs) recently reshape the field of NLP with remarkable performance benefits across a range of complex language benchmarks. However, due to their gigantic size and computational costs; they still remain out of reach for many researchers and small industries. In this section, we investigate the presence of essential sparsity in two popular LLM (`Vicuna-7B` & `13B`) with one-shot magnitude pruning. Note that unlike our previous setting, where we perform sparse fine-tuning after the mask is identified; here we do not perform downstream task-specific fine-tuning and evaluate the performance directly "zero-shot" on the sparsified pre-trained checkpoint.

Figure 11(a) illustrates the performance drop of `Vicuna-7B` and `Vicuna-13B` on popular `MMLU` benchmark (Stem, Humanities, Social Science, Others) when $x\%$ of the lowest magnitude weights

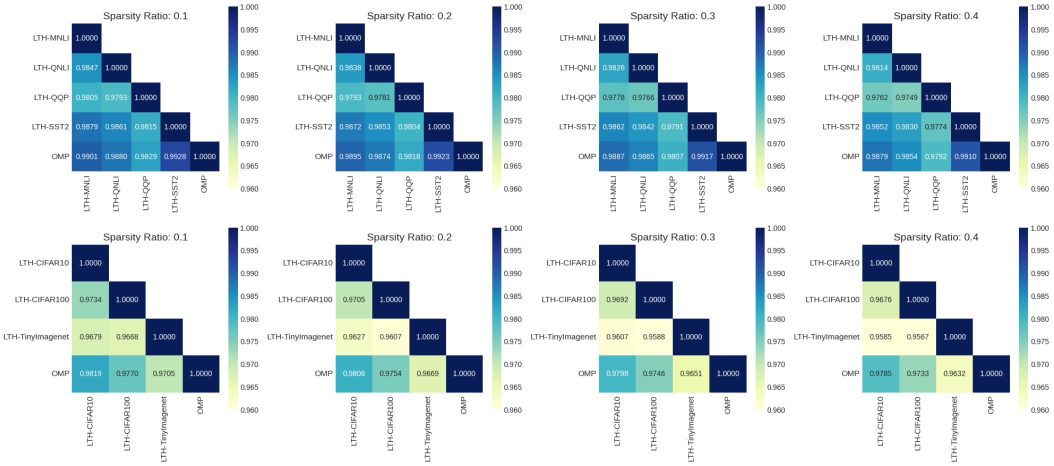

Figure 10: Cosine similarity between the masks obtained by LTH (depending on downstream task) and OMP on `bert-base` (Row 1) and `ViT-base` (Row 2) for sparsity ratio $s \in \{10\%, 20\%, 30\%, 40\%\}$. High cosine similarity indicate masks identified by LTH and OMP are significantly similar.

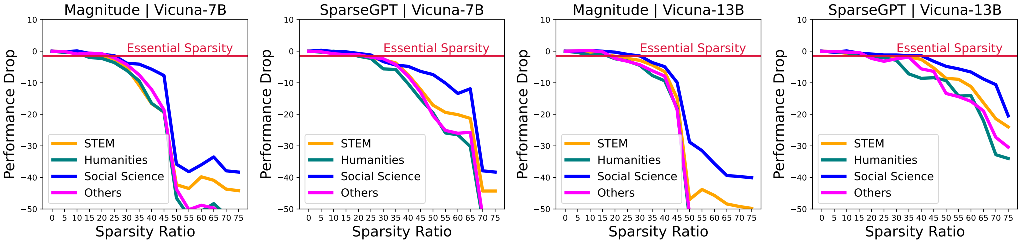

Figure 11: Performance drop of `Vicuna-7B` and `13B` on `MMLU` benchmark [31] w.r.t. the dense counterpart, using OMP and the recently proposed SparseGPT [85]. This indicates a notable fraction of weights of `Vicuna-7B/13B` can also be removed at free without any significant drop in performance.

are removed from the dense pre-trained checkpoint. It is interesting to see that our essential sparsity observations hold true even for modern LLMs, sending a favorable signal about the hidden existence of high-quality sparse subnetwork which can be identified free of calibration data or post-optimization, in dense pre-trained checkpoints. To further enrich our study, we replaced OMP with the recently proposed SparseGPT [85] and found it to have generally consistent trends with OMP (Figure 11(b)).

Compared to SparseGPT, our research uncovers the surprising simplicity of LLM pruning *within a specific sparsity range*. The remarkable effectiveness of one-shot, magnitude-based pruning not only establishes a robust baseline but also suggests considerable practical value, for instance, in enabling economical “on-the-fly" LLM pruning that adapts to fluctuating resource availability during testing. Nonetheless, it’s worth noting that more refined pruning strategies like SparseGPT can further push the boundary of essential sparsity and identify better sparse subnetworks at comparatively higher sparsity ratios, albeit at relatively higher expenses. Bridging the performance divide between OMP and SparseGPT remains an intriguing avenue for future investigation.

8 Conclusion

We comprehensively study induced sparse patterns across large pre-trained vision and language transformers. We experimentally validated the ubiquitous existence of essential sparsity across large pre-trained transformer models of varying scales for vision and language (including LLMs), irrespective of the training strategy used for pre-training them. We also present an intriguing emerging phenomenon of abrupt sparsification during the pre-training of transformers and its ability to abstract knowledge within few parameter subset with increasing pre-training data volume. Lastly, we studied the performance of LTH with respect to essential sparsity. Our future work will aim to extend our essential sparsity observations to more gigantic models, and to push for higher sparsity ranges.

Acknowledgement

The research is in part supported by the Intelligence Advanced Research Projects Activity (IARPA) under Contract No. 2022-21102100004. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of IARPA or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[3] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A survey on visual transformer. *ArXiv*, abs/2012.12556, 2020.

[4] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herv'e J'egou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[5] Zhiyuan Mao, Ajay Jaiswal, Zhangyang Wang, and Stanley H. Chan. Single frame atmospheric turbulence mitigation: A benchmark study and a new physics-inspired transformer model. *ArXiv*, abs/2207.10040, 2022.

[6] Minghang Zheng, Peng Gao, Renrui Zhang, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *ArXiv*, abs/2011.09315, 2021.

[7] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam M. Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018.

[8] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

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

[10] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.

[11] Ajay Jaiswal, Liyan Tang, Meheli Ghosh, Justin Rousseau, Yifan Peng, and Ying Ding. Radbert-cl: Factually-aware contrastive learning for radiology report classification. *Proceedings of machine learning research*, 158:196–208, 2021.

[12] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718*, 2019.

[13] 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.

[14] Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. Cognitive graph for multi-hop reading comprehension at scale. *arXiv preprint arXiv:1905.05460*, 2019.

[15] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[16] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[17] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[18] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot, 2023.

[19] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

[20] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020.

[21] Ajay Jaiswal, Haoyu Ma, Tianlong Chen, Ying Ding, and Zhangyang Wang. Spending your winning lottery better after drawing it, 2022.

[22] Lu Yin, Shiwei Liu, Fang Meng, Tianjin Huang, Vlado Menkovski, and Mykola Pechenizkiy. Lottery pools: Winning more by interpolating tickets without increasing training or inference cost. *arXiv preprint arXiv:2208.10842*, 2022.

[23] Haoran You, Chaojian Li, Pengfei Xu, Yonggan Fu, Yue Wang, Xiaohan Chen, Richard G Baraniuk, Zhangyang Wang, and Yingyan Lin. Drawing early-bird tickets: Towards more efficient training of deep networks. *arXiv preprint arXiv:1909.11957*, 2019.

[24] Ajay Kumar Jaiswal, Haoyu Ma, Tianlong Chen, Ying Ding, and Zhangyang Wang. Training your sparse neural network better with any mask. In *International Conference on Machine Learning*, pages 9833–9844. PMLR, 2022.

[25] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019.

[26] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020.

[27] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. The lottery ticket hypothesis at scale. *arXiv preprint arXiv:1903.01611*, 2019.

[28] Xiaohan Chen, Yu Cheng, Shuohang Wang, Zhe Gan, Zhangyang Wang, and Jingjing Liu. Earlybert: Efficient bert training via early-bird lottery tickets. *arXiv preprint arXiv:2101.00063*, 2020.

[29] Sai Prasanna, Anna Rogers, and Anna Rumshisky. When bert plays the lottery, all tickets are winning. *arXiv preprint arXiv:2005.00561*, 2020.

[30] Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. *Advances in Neural Information Processing Systems*, 34:19974–19988, 2021.

[31] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[32] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[33] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations*, 2016.

[34] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1135–1143. Curran Associates, Inc., 2015.

[35] Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 598–605. Morgan-Kaufmann, 1990.

[36] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[37] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[38] Hao Zhou, Jose M Alvarez, and Fatih Porikli. Less is more: Towards compact cnns. In *European Conference on Computer Vision*, pages 662–677. Springer, 2016.

[39] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

[40] Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20378–20389. Curran Associates, Inc., 2020.

[41] Hao Jiang, Ke Zhan, Jianwei Qu, Yongkang Wu, Zhaoye Fei, Xinyu Zhang, Lei Chen, Zhicheng Dou, Xipeng Qiu, Zikai Guo, et al. Towards more effective and economic sparsely-activated model. *arXiv preprint arXiv:2110.07431*, 2021.

[42] Babak Hassibi and David Stork. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5, 1992.

[43] Xin Dong, Shangyu Chen, and Sinno Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. *Advances in Neural Information Processing Systems*, 30, 2017.

[44] William Finnoff, Ferdinand Hergert, and Hans Georg Zimmermann. Improving model selection by nonconvergent methods. *Neural Networks*, 6(6):771–783, 1993.

[45] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017.

[46] Jian-Hao Luo and Jianxin Wu. Autopruner: An end-to-end trainable filter pruning method for efficient deep model inference. *Pattern Recognition*, 107:107461, 2020.

[47] Pedro Savarese, Hugo Silva, and Michael Maire. Winning the lottery with continuous sparsification. In *Advances in Neural Information Processing Systems 33 pre-proceedings*, 2020.

[48] Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana. Hydra: Pruning adversarially robust neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.

[49] Yihua Zhang, Yuguang Yao, Parikshit Ram, Pu Zhao, Tianlong Chen, Mingyi Hong, Yanzhi Wang, and Sijia Liu. Advancing model pruning via bi-level optimization. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[50] Michael Zhu and Suyog Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.

[51] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.

[52] Tao Lin, Sebastian U. Stich, Luis Barba, Daniil Dmitriev, and Martin Jaggi. Dynamic model pruning with feedback. In *International Conference on Learning Representations*, 2020.

[53] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Zahra Atashgahi, Lu Yin, Huanyu Kou, Li Shen, Mykola Pechenizkiy, Zhangyang Wang, and Decebal Constantin Mocanu. Sparse training via boosting pruning plasticity with neuroregeneration. *Advances in Neural Information Processing Systems (NeurIPs).*, 2021.

[54] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications*, 9(1):2383, 2018.

[55] Hesham Mostafa and Xin Wang. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, 2019.

[56] Tim Dettmers and Luke Zettlemoyer. Sparse networks from scratch: Faster training without losing performance. *arXiv preprint arXiv:1907.04840*, 2019.

[57] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pages 2943–2952. PMLR, 2020.

[58] Shiwei Liu, Lu Yin, Decebal Constantin Mocanu, and Mykola Pechenizkiy. Do we actually need dense over-parameterization? in-time over-parameterization in sparse training. *arXiv preprint arXiv:2102.02887*, 2021.

[59] Jonathan Schwarz, Siddhant Jayakumar, Razvan Pascanu, Peter E Latham, and Yee Teh. Powerpropagation: A sparsity inducing weight reparameterisation. *Advances in neural information processing systems*, 34:28889–28903, 2021.

[60] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2020.

[61] Hidenori Tanaka, Daniel Kunin, Daniel LK Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. In *Advances in Neural Information Processing Systems 33 pre-proceedings*, 2020.

[62] Pau de Jorge, Amartya Sanyal, Harkirat S Behl, Philip HS Torr, Gregory Rogez, and Puneet K Dokania. Progressive skeletonization: Trimming more fat from a network at initialization. *arXiv preprint arXiv:2006.09081*, 2020.

[63] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[64] 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.

[65] Ofir Zafrir, Ariel Larey, Guy Boudoukh, Haihao Shen, and Moshe Wasserblat. Prune once for all: Sparse pre-trained language models. *arXiv preprint arXiv:2111.05754*, 2021.

[66] Eldar Kurtic, Daniel Campos, Tuan Nguyen, Elias Frantar, Mark Kurtz, Benjamin Fineran, Michael Goin, and Dan Alistarh. The optimal bert surgeon: Scalable and accurate second-order pruning for large language models. *arXiv preprint arXiv:2203.07259*, 2022.

[67] Dongkuan Xu, Ian EH Yen, Jinxi Zhao, and Zhibin Xiao. Rethinking network pruning–under the pre-train and fine-tune paradigm. *arXiv preprint arXiv:2104.08682*, 2021.

[68] François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M Rush. Block pruning for faster transformers. *arXiv preprint arXiv:2109.04838*, 2021.

[69] Qingru Zhang, Simiao Zuo, Chen Liang, Alexander Bukharin, Pengcheng He, Weizhu Chen, and Tuo Zhao. Platon: Pruning large transformer models with upper confidence bound of weight importance. In *International Conference on Machine Learning*, pages 26809–26823. PMLR, 2022.

[70] Elias Frantar, Eldar Kurtic, and Dan Alistarh. M-fac: Efficient matrix-free approximations of second-order information. *Advances in Neural Information Processing Systems*, 34:14873–14886, 2021.

[71] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[72] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020.

[73] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[74] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[75] Shiwei Liu, Tianlong Chen, Zhenyu Zhang, Xuxi Chen, Tianjin Huang, Ajay Jaiswal, and Zhangyang Wang. Sparsity may cry: Let us fail (current) sparse neural networks together! *arXiv preprint arXiv:2303.02141*, 2023.

[76] Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California, June 2016. Association for Computational Linguistics.

[77] Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. *arXiv preprint arXiv:2106.15772*, 2021.

[78] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*, 2021.

[79] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.

[80] Aojun Zhou, Yukun Ma, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, and Hongsheng Li. Learning n: m fine-grained structured sparse neural networks from scratch. *arXiv preprint arXiv:2102.04010*, 2021.

[81] Samira Abnar, Mostafa Dehghani, Behnam Neyshabur, and Hanie Sedghi. Exploring the limits of large scale pre-training. *arXiv preprint arXiv:2110.02095*, 2021.

[82] Donghyun Kim, Kaihong Wang, Stan Sclaroff, and Kate Saenko. A broad study of pre-training for domain generalization and adaptation. In *European Conference on Computer Vision*, 2022.

[83] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[84] Ziquan Liu, Yi Xu, Yuanhong Xu, Qi Qian, Hao Li, Antoni Chan, and Rong Jin. Improved fine-tuning by leveraging pre-training data: Theory and practice. *arXiv preprint arXiv:2111.12292*, 2021.

[85] Elias Frantar and Dan Alistarh. Massive language models can be accurately pruned in one-shot. *arXiv preprint arXiv:2301.00774*, 2023.