Instant Soup: Cheap Pruning Ensembles in A Single Pass Can Draw Lottery Tickets from Large Models

Ajay Jaiswal [1] **Shiwei Liu** [1] **Tianlong Chen** [1] **Ying Ding** [1] **Zhangyang Wang** [1]

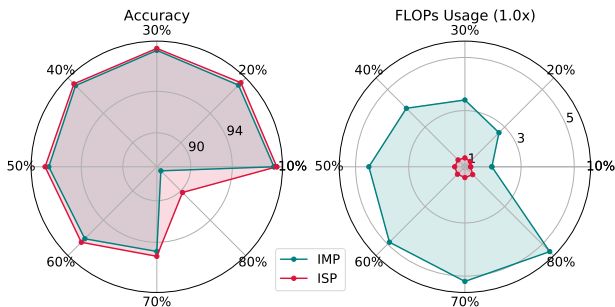*Figure 1.* Fine-tuning Accuracy (↑) and FLOPs counts (↓) of subnetworks with sparsity $s \in \{10\%, 20\%, ..., 80\%\}$ on `CIFAR-10` test set with a pre-trained CLIP (`ViT-B32`) model checkpoint. Note that ISP requires approximately single-pass computational cost of IMP to generate subnetworks with better performance than multiple rounds of IMP as required in Lottery Tickets.

Abstract

Large pre-trained transformers have been receiving explosive attention in the past few years, due to their wide adaptability for numerous downstream applications via fine-tuning, but their exponentially increasing parameter counts are becoming a primary hurdle to even just fine-tune them without industry-standard hardware. Recently, Lottery Ticket Hypothesis (LTH) and its variants, have been exploited to prune these large pretrained models generating subnetworks that can achieve similar performance as their dense counterparts, but LTH pragmatism is enormously inhibited by repetitive full training and pruning routine of iterative magnitude pruning (IMP) which worsens with increasing model size. Motivated by the recent observations of model soups, which suggest that fine-tuned weights of multiple models can be merged to a better minima, we propose **Instant Soup Pruning (ISP)** to generate

lottery ticket quality subnetworks, using a fraction of the original IMP cost by replacing the expensive intermediate pruning stages of IMP with computationally efficient weak mask generation and aggregation routine. More specifically, during the mask generation stage, ISP takes a small handful of iterations using varying training protocols and data subsets to generate many weak and noisy subnetworks, and superpose them to average out the noise creating a high-quality denoised subnetwork. Our extensive experiments and ablation on two popular largescale pre-trained models: `CLIP` (unexplored in pruning till date) and `BERT` across multiple benchmark vision {`MNIST`, `SVHN`, `Cars`, `GTSRB`, `CIFAR-10`, `CIFAR-100`} and language datasets {`MNLI`, `QNLI`, `QQP`, `SST`, `...`} validate the effectiveness of ISP compared to several state-of-the-art pruning methods. Additionally, we show that ISP can be easily modified with minimal overhead to produce benefits comparable to model soups, without the prerequisite to generate multiple candidates fine-tuned models. Codes are available at: `https://github.com/VITA-Group/instant_soup`.

1. Introduction

Large-scale transfer learning has recently become *showstealer* in modern deep learning, and transformer-based pretrained models (Devlin et al., 2018; Liu et al., 2019; Dosovitskiy et al., 2020; Liu et al., 2021; Radford et al., 2021) are now achieving state-of-the-art performance for a wide array of real-world computer vision (Dosovitskiy et al., 2020; Han et al., 2020; Li et al., 2023; Touvron et al., 2021; Mao et al., 2022; Jaiswal et al., 2021a; Zheng et al., 2021; Parmar et al., 2018) and natural language processing (Yang et al., 2019b; Liu et al., 2019; Talmor et al., 2018; Jaiswal et al., 2021b; Zheng et al., 2023; Yang et al., 2019a; Wang et al., 2018; Ding et al., 2019; Chowdhery et al., 2022; Wei et al., 2022; Jaiswal et al., 2023) applications. 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. In the hustle of building gigantic models, a parallel and growing field of model compression has been exploring the prospects to compress these enormous models at the cost of marginal/no sacrifice in performance, effectively reducing their computational and memory footprints.

Among many efforts for compressing models and accelerating inference (Frankle & Carbin, 2018; Chen et al., 2020a; Jaiswal et al., 2022b; Yin et al., 2022; Jaiswal et al., 2022c; Lee et al., 2018; Yu et al., 2017; 2020; Fang et al., 2023; Chen et al., 2023; Jaiswal et al., 2022a; Liu et al., 2023), network pruning eliminates unnecessary weights to generate smaller *subnetworks* in place of dense networks attaining similar performance, stands out as one of the most effective techniques. Lottery Ticket Hypothesis (LTH) (Frankle & Carbin, 2018) and its variants, reveal that dense, randomly-initialized networks contain small subnetworks which can match the test accuracy of original networks. Despite their insightful findings, there still exists a large **gap** in the practicality of these methods because of the fully dense training routine of IMP, which exacerbates with an increase in model capacity. Recently proposed EarlyBird routine (You et al., 2019; Chen et al., 2020b) which attempts to draw the winning tickets early in training, and pruning at initialization techniques have shown some promise in mitigating search through an expensive and tedious iterative process, yet their effectiveness for the large-scale pre-trained network is highly under-explored or they tend to have substandard performance at non-trivial sparsities. In this work, we ask: *Does there exist a principled and cheaper approach for fastly drawing high-quality lottery tickets in large pre-trained models within a limited computational budget, while preserving its performance and transferability?*

To this end, we explore the feasibility of obtaining cheap tickets for popular large pre-trained models CLIP(Radford et al., 2021) and BERT(Devlin et al., 2018) constrained by no multi-pass repetitive full-training, to meet the permissible computational budget. One straightforward approach is to curtail per round training cost of iterative magnitude pruning (IMP) with early stopping, but we found that such non-careful pruning approach often produces highly variable and substandard tickets, presumably due noisy and unstable state of the network, when pruned. Our close analysis of the pruned mask generated by LTH and cheap pruning methods (one-shot pruning) found that large-scale pre-trained transformers are highly overparameterized and pruning them at trivial sparsities does not necessarily require an expensive LTH paradigm, which encourages us to start pruning nonchalantly at trivial sparsities and become scrupulous at non-trivial sparsities. In addition, recently several works (Wortsman et al., 2022a; Ilharco et al., 2022; Juneja et al., 2022) investigate the intriguing phenomenon

of "model soups", and have shown that weights of multiple dense fine-tuned models can be merged together into better solutions lying in low error basins. In the sparse setting, a very recent attempt (Yin et al., 2022) reused the byproduct of IMP, and showed that tickets generated at each iteration of IMP could be superposed into a stronger subnetwork. However, its observations are limited to small-scale networks, and more importantly, the algorithm does not contribute to the computational efficiency of either finding lottery tickets or (re-)training networks. Motivated by these observations, we are interested to investigate: *if we can leverage the soup observations to eliminate noise induced by early pruning in IMP iteration, effectively leading to the stable sparse subnetwork and reduced cost.*

We propose **Instant Soup Pruning (ISP)**, a model soup-inspired perspective dedicated to generating lottery ticket quality subnetworks, using a fraction of the original IMP cost. More specifically, ISP uses a miniature random subset of training data to generate many weak and noisy subnetworks, and superpose them to average out the noise creating a high-quality denoised subnetwork. Similar to traditional IMP, ISP repeats the denoising routine following well-managed training iterations till the desired sparsity is reached, eliminating the need of IMP to perform a full pass of training before every pruning routine. Our experiments on CLIP (unexplored in pruning literature till date) and BERT across multiple datasets illustrate that ISP can find sparse subnetworks with better quality than LTH, with an affordable cost no more than a single pass of IMP. In addition to dense-to-sparse paradigm, interestingly, our ISP routine can be bluntly incorporated in dense-to-dense paradigm to incorporate the benefits of model soups in dense pre-trained models at almost negligible training cost at the pre-training stage, ultimately leading to better-fine tuning performance. Our contributions can be summarized as:

- We propose **Instant Soup Pruning (ISP)**, a novel pruning strategy that seamlessly integrates the "model soup" idea to significantly reduce the computational cost of IMP. ISP replaces the expensive intermediate pruning stages of IMP with computationally-cheap weak mask generation and denoising, while outperforming IMP for large pre-trained models.

- ISP naturally provides a "self-denoising" ability to eliminate the necessity of generating high quality/expensive masks (presumably "good solution basin") at each pruning stage, by instead generating multiple computationally inexpensive weak masks and averaging them out to reduce their solution noise.

- In the dense-to-dense paradigm, ISP can be adapted to **Instant Model Soup**, to inject the benefits of model soups in dense pre-trained models at marginal training cost, thereby improving fine-tuning performance com-
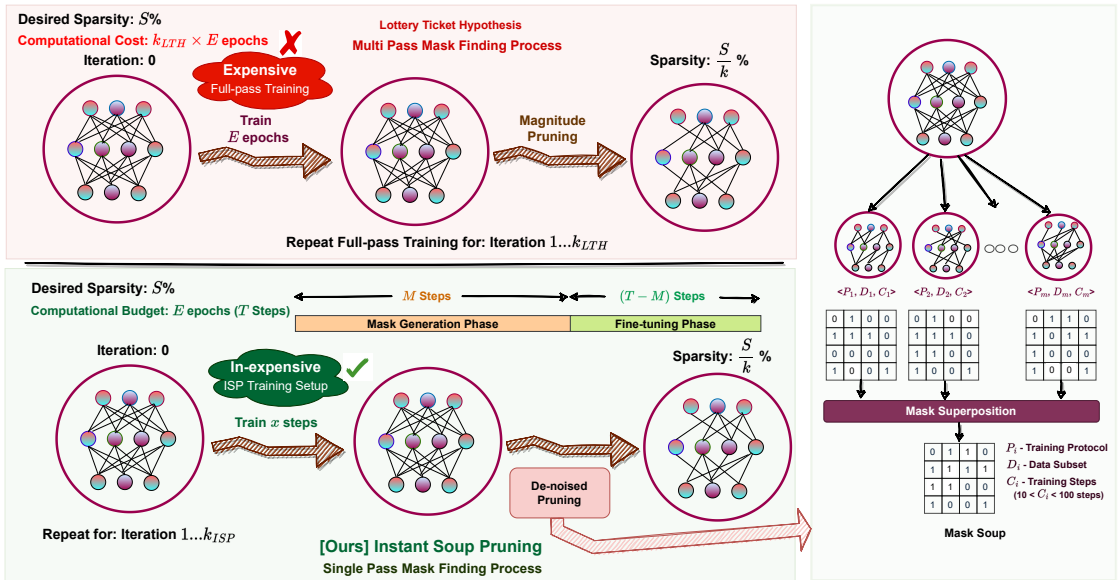
Figure 2. Overview of our proposed Instant Soup Pruning. We provide a detailed illustration of our proposed technique compared to conventional LTH with IMP. ISP replaces the expensive intermediate pruning stages of IMP with computationally-cheap weak mask generation and denoising while outperforming LTH. Unlike LTH, ISP consumes computation budget equivalent of a single pass of LTH.

parable to model soups without the need to generate multiple fine-tuned models.

■ Our extensive experiments on two popular large-scale pre-trained models (CLIP & BERT) across multiple benchmark vision {MNIST, SVHN, Cars, GTSRB, CIFAR-10, CIFAR-100} & language datasets {MNLI, QQP, STS-B, WNLI, QNLI, MPRC, RTS, SST-2, CoLA} validate the effectiveness of ISP wrt. several SOTA pruning methods.

2. Methodology

2.1. Revisiting LTH and Pre-trained Vision and Language Model Compression

In the past few years, scaling neural networks to improve information absorption has been pivotal for good optimization and generalization performance, but this unbounded parameter growth has made them computationally expensive with excessive memory requirements. The trend undoubtedly continues with the recent forefront of transformers, where more and more layers are stacked with dense attention blocks (eg. T5 has $\sim 10+$ billion parameters) calling for expensive computational resources and prolonged training or fine-tuning time. Recently, some work (Chen et al., 2021b; Gan et al., 2022; Chen et al., 2021a; 2020b; You et al., 2019; Prasanna et al., 2020) have explored Lottery Ticket Hypothesis (LTH) to understand the parameter redundancy in the current prevailing large scale transformer models, and attempted to compress it to non-trivial sparsities by repetitive initialization-training-pruning operation. Despite their success to find high-quality compressed mod-
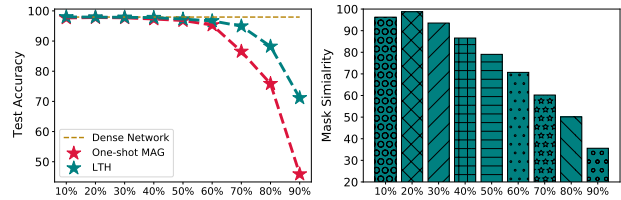

Figure 3. Performance comparison of LTH and One-shot magnitude-based pruning of CLIP (ViT-B32) on CIFAR-10 at sparsity $S \in \{10\%, 20\%, ..., 90\%\}$ (left). Cosine similarity between the binary prune masks obtained by LTH and One-shot magnitude pruning of CLIP (ViT-B32) on CIFAR-10 (right).

els on a range of downstream tasks, it is impossible to ignore the cost of finding these subnetworks, since winning tickets can only be identified by pruning unimportant connections after fully training a dense network in a conventional LTH paradigm, which worsens significantly with increasing model size. For example, (Prasanna et al., 2020; Chen et al., 2020a) explored pruning BERT to matching subnetworks at 40% to 90% sparsity across multiple tasks, on the other hand, (Gan et al., 2022) explored LTH for compressing large pre-trained VL models while preserving its performance.

Recently, (Chen et al., 2020b) explored the EarlyBird (You et al., 2019) idea and proposed jointly training BERT and some sparsity-inducing coefficients which can be used to draw the subnetworks followed by fine-tuning, but its joint training step again is as expensive as normal BERT training, and experimentally we found that its performance becomes sub-standard compared to LTH in non-trivial sparsity range. In this work, we propose a novel pruning strategy based on strong insights of the inherent benefits of gigantic size

and model soups, which can be equivalent to or even better than LTH and its variant LTH-Rewind. We, for the first time, explore model compression for a recent extremely popular open vocabulary network CLIP along with BERT, to illustrate our approach benefits by using merely the computational cost equal to a single pass of conventional LTH.

2.2. Instant Soup Pruning: A novel cost-effective pruning perspective

In this section, we introduce a novel pruning algorithm, named **Instant Soup Pruning (ISP)** which primarily aims to reduce the computational overhead of conventional LTH while searching for lottery tickets in large-scale pre-trained transformers, facilitating benefits from both performance and computation perspective.

ISP is motivated from the following three observations:

- *Firstly*, large-scale pre-trained transformers are highly over-parameterized, and pruning them at trivial (eg. 10%, 20%, etc. depending on task and model size) sparsities does not require sophisticated pruning methods like LTH or LTH-Rewind to get high-quality sparse subnetworks. We surprisingly observed that *at trivial sparsities, the sparse mask generated by LTH and cheap one-shot magnitude pruning is significantly similar* which thereby reflects in the test performance of the subnetworks. For example, Figure 3(a) illustrates the performance of subnetworks obtained by LTH and one-shot pruning on CLIP (ViT-B32) and Figure 3(b) illustrates the cosine similarity between the binary prune mask identified by LTH and one-shot pruning. It can be clearly observed that *at trivial sparsities such as 10%, 20%, and 30%, both unreasonably cheap one-shot pruning and expensive LTH identify approximately similar masks with 96.27%, 98.75%, and 94.32% cosine similarity score*. It conveys a strong message to save the computation budget of full training passes of LTH, which are seemingly unnecessary.

- *Secondly*, Early-Bird (You et al., 2019) tickets, although limited to small architectures (ResNets, Vgg16, etc), conveyed a strong yet highly overlooked message that high-quality tickets can emerge at a very early training stage by pruning networks trained at much earlier points (before the accuracies reach their final top values). Recently, (Chen et al., 2020b) showed that this observation holds true for BERT, but it cannot recover the full performance of LTH. To this end, complementary to our first observation, our work extends the early-bird findings by *proposing to look more carefully while searching tickets at non-trivial sparsity (high sparsity regime) compared to trivial sparsity by progressively increasing training steps over each call to pruning routine* in the mask finding stage.

- *Lastly*, to mitigate the issue of pre-mature pruning to generate sub-standard pruning masks, we borrow inspiration from the intriguing phenomenon of "model soups", which illustrate weights of large-scale independently fine-tuned models can be merged together into a better solution. Our work proposes a novel approach of *mask soups by superposing multiple multiple cheap pruning masks, to attenuate the noise within them due to the presumably unstable state of the network while pruning*, giving a high-quality denoised pruning mask.

Algorithm Overview Consider a dense, pre-trained network $f(x; \theta)$, as shown in Figure 2, LTH trains f to achieve minimum validation loss f_{loss} using E epochs with a test accuracy f_{acc}, when optimized with Adam optimizer on a training dataset D. Once, f_{acc} is achieved, the fine-tuned network $f(x; \theta_E)$ is pruned using magnitude pruning to generate a subnetwork $f(x; m \odot \theta_E)$, with a mask $m \in \{0, 1\}$. This process is repeated for k iterations till the desired sparsity $S\%$ is achieved, generating a subnetwork $f_{LTH}(x; m \odot \theta_{\mathcal{O}(k \cdot E)})$ with accuracy f_{acc}^{LTH}.

In contrast, our proposed approach ISP aims to generate $f_{ISP}(x; m' \odot \theta_{\mathcal{O}(E)})$ with accuracy f_{acc}^{ISP}, such that $f_{acc}^{ISP} \geq f_{acc}^{LTH}$ with sparse mask $m' \in \{0, 1\}$ and sparsity $S\%$. Given the computational budget of E epochs, which translates to T steps with batch size B, ISP is composed of two distinct phases: *mask generation phase* and *fine-tuning phase* which uses M and $(T - M)$ steps respectively to produce a high-quality fine-tuned subnetwork with desired sparsity $S\%$, usually outperforming LTH.

2.2.1. MASK GENERATION STAGE

We will first discuss our novel, computationally efficient, and high-quality sparse mask generation steps for ISP incorporating the aforementioned motivation. Note that similar to IMP, ISP is also an iterative train-prune-retrain procedure, but it uses a highly optimized train/re-train subroutine starting from the parameter state obtained from the previous iteration. Given the computational budget of M steps, the mask generation stage of ISP is carefully designed to start in a relaxed fashion (spending few learning steps) while pruning at trivial sparsities and gradually become meticulous while approaching non-trivial sparsity regime, acknowledging the sensitivity of pruning while operating in high sparsities.

More specifically, we introduce a hyperparameter t, which is defined as a small initial seed step count usually equates to the number of steps required to look $\sim 10\%$ of training data with batch size B, before the first call to pruning routine. At i-th call to the pruning routine, we use $t \times (i + 1)$ training steps for calibrating (training) the network before pruning. For example, at the 0-th iteration when the network is dense, ISP uses steps equivalent to 10% of training data while at 5-th iteration, it uses 60% of training data in the re-training.

Algorithm 1 Instant Soup Pruning

Input　: Pre-trained Network: $f(x; \theta_0)$; Initial Training Seed: t; Compression ratio: $s\%$; Desired sparsity: $S\%$; Training Budget: T

Output : Pruned trained subnetwork with a mask $m' \in \{0, 1\}$: $f_{ISP}(x; m' \odot \theta_{\mathcal{O}(E)})$

```
/* k is chosen st.  ∑ᵏᵢ₌₀(i+1)·t ≤ M         */
```
for $i \leftarrow 0$ **to** k **do**
```
  /* Sparsity dependent training steps
     before call to pruning routine.
     Spend low training steps for trivial
     sparsity compared to non-trivial
     sparsity.                            */
```
　　Train $f(x; \theta_{i \cdot t}) \rightarrow f(x; \theta_{(i+1) \cdot t})$
　　if $\text{sparsity}(f) < S\%$ **then**
　　　　$m_{(i+1)} \leftarrow \text{DenosiedPrune}(f, s, m_i)$
```
      /* Compress network with newly
         returned denoised mask        */
```
　　　　Apply $m_{(i+1)} \rightarrow f(x; \theta_{(i+1) \cdot t})$
　　end if
end for
```
/* Fine-tune the obtained subnetwork for
   the remaining (T − ((k + 1) · t)) steps in
   computational budget.                    */
```
Fine-tune $f(x; \theta_{(k+1) \cdot t}) \rightarrow f(x; \theta_{T-((k+1) \cdot t)})$

Algorithm 2 Denoised Pruning Procedure

Input　: Network to prune: $f(x; m_i \odot \theta)$ where m_i denotes binary mask at i-th iteration of ISP; Denoiser Count: N; Compression Rate: $s\%$; Look-ahead steps: C (typically $10 \leq C \leq 100$ steps)

Output : New denoised mask: $m_{(i+1)}$

$m_{temp} \leftarrow \text{MagnitudePruning}(f(x; \theta), s\%)$
for $n \leftarrow 0$ **to** N **do**
```
  /* Look-ahead training to generate
     multiple masks for denoising      */
```
　　$P_n \leftarrow$ Training protocol // learning rate, weight decay, etc.
　　$D_n \leftarrow$ Random data samples
　　Look-ahead training: $f(x; \theta) \rightarrow f(x; \theta_C)$
```
  /* Denosing to average mask noise      */
```
　　$m_{temp} \leftarrow m_{temp} \cup \text{MagnitudePruning}(f(x; \theta_C), s\%)$
end for
$m_{(i+1)} \leftarrow \text{One-shot-adjustment}(m_{temp}, f(x; \theta))$
return $m_{(i+1)}$

To ensure that ISP does not produce sub-standard quality mask at each pruning iteration, we next provide details of our novel **denoised pruning routine** which is inspired by recently proposed "model soups" phenomenon, to average out noise induced due to pre-mature pruning.

Denoised Pruning:　Recently, several works (Wortsman et al., 2022a; Ilharco et al., 2022; Juneja et al., 2022) have validated the intriguing phenomenon of model soups, which suggest training multiple models with various hyperparameters and average the weights of models fine-tuned independently at no cost to achieve comparatively high performance. Motivated by their argument that fine-tuned models optimized independently from the same pre-trained initialization lies in the same basin of the error landscape and averaging them improves generalization, we are tempted to ask an unexplored question: *Can we generate extremely cheap pruning masks using varying hyperparameters and average them out to improve the quality?*

To this end, we propose a novel **Denoised Pruning Procedure**, which explores the superimposition of computationally cheap pruning mask obtained by magnitude-based one-shot pruning of the network with marginal look-ahead training. Algorithm 2 illustrates the details of our denoising procedure which perform look-ahead training of the network using varying training protocol for a random subset of training samples, merely using $10 - 100$ training steps,

to generate N candidate binary masks. In order to superimpose them, we found that a simple <u>union</u> of these N binary masks is sufficient to improve the quality, facilitating ISP to achieve comparable/even better performance than expensive LTH. We hypothesize (and later experimentally validate) that our denoising procedure significantly helps eliminate any induced noise due to pre-mature pruning during ISP iterations, specifically at non-trivial sparsities.

2.2.2. FINE-TUNING STAGE

Our proposed method (ISP), is a single pass pruning algorithm that generates a high-quality fine-tuned subnetwork with desired sparsity $S\%$ using the computational budget equivalent to a single pass of LTH (T steps). As mentioned before, ISP is composed of the mask generation phase (M steps) followed by fine-tuning the obtained subnetwork for the remaining ($T - M$ steps). Note that, unlike conventional LTH, ISP does not restart the network training from the initial pre-trained weight; instead, as explained in Algorithm 1, it simply fine-tunes the network state (unchanged optimizer, learning rate, etc.) obtained immediately after pruning $S\%$ of parameters for ($T - M$) steps.

2.3. Instant Model Soup: A sparsity-inspired extension to dense training

In the conventional setting, to improve the generalization performance of a model, it is highly recommended to train multiple models and use their ensemble on the test set, but it comes at a heavy inference and training cost. Recently, several works (Wortsman et al., 2022a; Ilharco et al., 2022; Juneja et al., 2022) have illustrated that, unlike conventional

ensembles, the weights of multiple fine-tuned large pre-trained models can be merged together by simply averaging their weights (aka. model soup) to beat the performance of ensembles without incurring any additional inference or memory cost. While this approach effectively reduces the additional inference overhead of ensembles, it still requires expensive fine-tuning of multiple large pre-trained models with varying hyperparameters. Inspired by our idea of Instant Soup Pruning, which illustrates that binary masks originated from cheap training with different training protocols can be superimposed/denoised to improve quality, we are enticed to explore: *Can we denoise the initial pre-trained weights of dense large transformers using sparse cheap training to inject the model soup benefits early during fine-tuning, eliminating the need to generate multiple fully fine-tuned models for model soups?*

Algorithm 3 Instant Model Soup

Input　: Pre-trained Model: $f(x; \theta_0)$; D_{Train}; D_{Val}; Denosier Count: K

Output : New denoised Pre-trained Model: $f(x; \theta_{new})$

$f(x; \theta_{interpolated}) \leftarrow \texttt{DeepCopy}(f(x; \theta_0))$

for $k \leftarrow 0$ **to** K **do**

　　/* Prune to create a sparse subnetwork from input model   */
　　$f(x; m_s \odot \theta_0) \leftarrow$
　　$\texttt{MagnitudePruning}(f(x; \theta_0), s\%)$
　　/* Weakly train sparse subnetwork for denoising   */
　　$H_k, C_k \leftarrow$ Training Protocol, Steps
　　$D_k \leftarrow \texttt{sample}(D_{Train})$
　　$f_{weak}(x; m_s \odot \theta_k) \leftarrow \texttt{Train}(f(x; m_s \odot \theta_0), H_k, C_k, D_k)$
　　/* Denoising using linear interpolation with val data   */
　　$f(x; \theta_{interpolated}) \leftarrow (f_{weak}(x; m_s \odot \theta_k), f(x; \theta_{interpolated}), D_{Val})$

end for

$f(x; \theta_{new}) \leftarrow f(x; \theta_{interpolated})$

return $f(x; \theta_{new})$

Our primary objective is to ripe the benefits of model soups without the need to generate multiple fully fine-tuned models, thereby contenting the hurdle of high training overhead of model soups. Algorithm 3 provide details of the pseudocode of our new **Instant Model Soup (IMS)** approach that uses cheap sparse training for injecting model soup benefits at marginal cost early before fine-tuning to save the hurdle of multiple finetuning and averaging. More specifically, IMS first creates multiple subnetworks with varying sparsities from the pre-trained dense model and train them independently for a few iterations ($\sim$100 iterations) using different hyperparameter configuration, and data subsets. Next, all the weakly trained subnetwork weights are merged

together with the initial pre-trained model weights using linear interpolation following (Ilharco et al., 2022). Our proposed usage of sparse subnetworks for denoising significantly reduces the computational cost of the dense training step, thereby making it more efficient. Our experiments on CLIP (ViT-B32) and BERT-BASE impart an interesting finding that pre-trained models denoised by IMS can achieve performance comparable to model soups, without expensive full fine-tuning and then averaging.

	Initial LR	Epochs	Compression Rate	Look Ahead (C)	Weight Decay	Denoiser
BERT$_{\text{BASE}}$	2×10^{-5}	10	10%	30 iter	0.0-AdamW	4
CLIP$_{\text{ViT-B32}}$	1×10^{-5}	22	15%	50 iter	0.1-AdamW	5

Table 1. Details of our primary hyperparameter configurations used in our experiments across different evaluation datasets.

3. Experiments and Analysis

3.1. Network, Dataset, and Settings

In our experiments, we adopted the official CLIP implementation provided by (Radford et al., 2021) as our starting point for our experiments using pre-trained vision transformer (Dosovitskiy et al., 2020) (ViT-B32) models. For fine-tuning CLIP, we use the frozen final classification layer output by CLIP's text tower to eliminate the necessity of introducing any learnable parameters while in the case of BERT, we add a final task-specific classification layer($\sim 3\%$ of total parameter count). For our BERT-related experiments, we use the HuggingFace (Wolf et al., 2019) pre-trained weights of BERT$_{\text{BASE}}$ transformer blocks and hidden state size 768. Additional necessary details of our hyperparameters required for fine-tuning are provided in Table 1. Note that during pruning, we prune the key trainable part of the network *(key, query, value, dense)* ignoring embeddings for simplicity. We consider a diverse set of image classification tasks from (Radford et al., 2021): Cars, GTSRB, MNIST, SVHN, CIFAR10/100 and downstream NLP tasks from GLUE (Wang et al., 2018) benchmark: MNLI, QQP, STS-B, WNLI, QNLI, MPRC, RTS, SST-2, CoLA; to thoroughly investigate the effectiveness of our proposed approaches wrt. state-of-the-art pruning benchmarks. In addition to Lottery tickets, we have compared Instant Soup Pruning (ISP) against several recently proposed pruning methods such as Lottery Pools (Yin et al., 2022), Early bird (You et al., 2019), two popular pruning at initialization methods (SNIP (Lee et al., 2018), GraSP (Yu et al., 2020)), Progressive pruning (Iterative pruning and training), and one-shot magnitude pruning.

3.2. Performance comparison of Instant Soup Pruning wrt. SOTA pruning methods

In this section, we conduct a systematic and extensive study to understand the performance benefits of our proposed Instant Soup Pruning in terms of fine-tuning accuracy vs.

Table 2. Details of fine-tuning CLIP (ViT-B32) at varying sparsity levels using Instant Soup Pruning following the settings listed in Table 1. Learning rate decays linearly from the initial value to zero. The evaluation metrics follow standards in (Radford et al., 2021). Entries with errors are the average across three runs, and errors are the standard deviations. LTH results are obtained using IMP.

Pruning Method	Cars			MNIST			SVHN			GTSRB			CIFAR10			CIFAR100		
	30%	40%	50%	70%	80%	90%	70%	80%	90%	50%	60%	70%	60%	70%	80%	60%	70%	80%
Full CLIP$_{ViT-B32}$	76.43 ± 0.5			99.61 ± 0.07			97.40 ± 0.11			99.08 ± 0.24			97.6 ± 0.24			89.35 ± 0.19		
Random	9.11	5.58	4.47	98.71	97.42	87.04	89.85	85.61	73.74	93.76	93.65	90.97	74.51	69.84	64.96	45.20	39.92	43.24
One-shot [Mag]	71.95	68.07	56.79	99.27	98.87	97.47	95.02	91.96	85.76	98.60	97.84	96.14	95.31	86.56	75.85	80.39	63.18	46.63
Progressive [Mag]	69.85	68.62	64.43	99.52	97.77	95.19	95.78	90.53	85.75	98.97	97.57	96.34	95.25	90.87	78.11	81.79	70.53	60.93
EarlyBird (You et al., 2019)	72.53	70.76	65.90	99.38	98.96	97.64	96.34	95.93	87.02	98.15	98.19	97.26	96.06	94.18	86.84	84.22	76.79	65.67
SNIP (Lee et al., 2018)	71.51	68.79	59.01	99.25	98.72	97.50	95.33	91.94	82.98	98.62	97.95	96.22	95.01	87.45	76.12	81.10	62.89	55.89
GraSP (Wang et al., 2020)	71.42	68.55	58.12	99.30	98.51	97.15	95.09	91.44	84.72	98.37	97.42	95.91	95.20	86.89	75.88	80.67	66.31	52.30
LTH (Frankle & Carbin, 2018)	73.97	72.02	66.12	99.41	99.38	98.22	96.69	95.28	87.41	98.71	98.35	97.79	96.42	94.91	87.47	84.25	78.60	65.38
LTH - Rewind	74.28	72.09	66.07	99.62	**99.64**	98.18	96.72	95.22	87.47	98.78	98.36	97.87	96.53	94.88	87.28	84.46	78.62	65.71
Lottery Pool (Yin et al., 2022)	73.10	70.53	64.67	99.25	98.97	97.76	96.54	95.12	87.29	98.52	98.30	97.55	96.14	94.50	87.11	84.07	78.21	64.39
ISP [Ours]	**75.13**	**72.20**	**66.32**	**99.69**	99.61	**98.82**	**96.93**	**96.46**	**87.59**	**99.06**	**99.01**	**98.52**	**96.82**	**95.18**	**91.20**	**85.11**	**79.57**	**71.09**
(std.)	±0.34	±0.27	±0.82	±0.07	±0.15	±0.21	±0.08	±0.05	±0.11	±0.15	±0.29	±0.32	±0.15	±0.20	±0.14	±0.19	±0.22	±0.17

Table 3. Details of fine-tuning BERT (BASE) at varying sparsity levels using Instant Soup Pruning following the settings listed in Table 1. Learning rate decays linearly from the initial value to zero. The evaluation metrics follow standards in (Wolf et al., 2019). Entries with errors are the average across three runs, and errors are the standard deviations. LTH results are obtained using IMP.

Dataset	MNLI	QQP	STS-B	WNLI	QNLI	MPRC	RTS	SST-2	CoLA
Sparsity	70%	90%	50%	90%	70%	50%	60%	60%	50%
Full BERT$_{BASE}$	82.4 ± 0.5	90.2 ± 0.5	88.4 ± 0.3	54.9 ± 1.2	89.1 ± 1.0	85.2 ± 0.1	66.2 ± 3.6	92.1 ± 0.1	54.5 ± 0.4
Random	67.5	76.3	21.0	53.5	61.9	69.6	56.0	83.1	9.6
One-shot	78.8	86.2	83.9	53.1	86.2	83.7	62.9	86.5	49.7
Progressive	79.1	87.5	85.0	53.3	87.2	83.8	65.4	86.6	52.2
EarlyBird	82.5	89.4	88.1	54.0	88.5	84.6	66.1	91.2	53.5
Lottery Ticket	82.6	90.0	88.2	54.9	88.9	84.9	65.0	91.9	53.8
Lottery Pool	80.4	89.1	86.4	50.9	87.6	84.5	62.7	90.9	52.6
ISP	**82.71 ± 0.6**	**90.59 ± 0.5**	**88.64 ± 0.1**	**55.33 ± 0.3**	**90.06 ± 1.0**	**85.38 ± 0.1**	**65.96 ± 0.3**	**92.43 ± 0.6**	53.61 ± 0.2

pruning ratios by comparing against multiple state-of-the-art pruning methods. We first test our approach on recently proposed CLIP (ViT-B32) (Radford et al., 2021) (unexplored for pruning till date) pre-trained with contrastive supervision from image-text pairs. For effective comparison and simplicity, all baselines and our proposed approach ISP are trained with similar optimizer and training settings provided in Table 1. Our EarlyBird (You et al., 2019), SNIP (Lee et al., 2018), and GraSP (Yu et al., 2020) baselines closely follow the implementation provided by their official GitHub repositories. In addition, to distinguish ISP from conventional progressive pruning, our progressive pruning baseline is implemented as periodic pruning using magnitude-based pruning followed by retraining. Lottery pools (Yin et al., 2022) is an interesting way to merge LTH by-product tickets. To ensure that the sparsity ratio remains comparable to ISP, we further prune the merged tickets obtained by pooling to the required sparsity, for reporting performance.

Our results for CLIP are summarized in Table 2. We first observe that among all baselines for CLIP compression, LTH consistently performs better, and rewinding helps in further improving its performance. We found that among recent pruning at initialization methods (SNIP and GraSP), SNIP has comparatively better performance than GraSP and they tend to be slightly better than one-shot magnitude prun-

ing. It can be clearly observed that ISP can beat expensive LTH (including rewinding) as well as all other baselines for almost all benchmark datasets and sparsity ratios. Very interestingly, for CIFAR-10 and CIFAR-100, we found that performance improvement of ISP increases with the sparsity ratio. For example, ISP surprisingly outperforms LTH-rewind by ~ 3.92 and $\sim 5.38\%$ on CIFAR-10 and CIFAR-100 respectively, while consuming merely fine-tuning cost equivalent to one single pass of LTH without the necessity of bookkeeping the rewinding weights of LTH-rewind. Our experimental results for BERT-base are summarized in Table 3. For BERT-related experiments, we have replicated the setting in (Chen et al., 2020a) and reported the performance of ISP across various GLUE benchmarking datasets at the sparsity level where LTH is able to identify winning tickets. Note that ISP is able to comfortably outperform LTH across 8 out of 9 tasks (noticeably for QNLI where ISP beat LTH by $> 1\%$).

3.3. Analysis of Denoising Iterations in ISP

Our proposed approach ISP is augmented by a novel idea of mask denoising, which explores the superimposition of computationally cheap pruning mask obtained by magnitude-based one-shot pruning of the network with marginal look-ahead training. In this section, we try to investigate the

Table 4. Impact of Denoising Module in improving the performance of ISP. Results are reported for three independent runs.

Approach	CLIP$_{ViT-B32}$		BERT$_{Base}$	
	SVHN	CIFAR-100	QQP	QNLI
ISP - Denoiser	96.11±0.21	70.32±0.13	89.96±0.39	89.28±0.87
ISP (Ours)	96.46±0.05	71.09±0.17	90.59±0.47	90.06±1.01

Table 5. Performance comparison of ISP wrt. denoiser count on CIFAR10 with CLIP (ViT-B32) pruned at 80% sparsity.

Denoiser Count	0	2	4	6	8	16
Performance	70.32	70.95	71.15	71.16	71.07	71.16

implication of our denoising iterations in improving ISP performance. Table 4 summarizes the performance comparison of ISP with/without the mask denoising while keeping the training settings exactly the same. Across both candidate architectures (CLIP and BERT), it can be clearly observed that ISP performance is significantly boosted by replacing the simple one-shot pruning with our denoise pruning routine. In addition, we also investigated how the number of denoising iterations will impact the ISP performance (see Table 5) and found that 4-5 denoising steps are sufficient for the denoised pruning, and increasing them beyond that does not provide a very noticeable performance gain. For consistency, in CLIP-related experiments, we have used 5 denoising iterations while for BERT, our results are reported using 4 denoising iterations.

3.4. Understanding the benefits of Instant Model Soups for pre-trained models

In this section, we discuss the benefits of our sparsity-inspired extension, Instant Model Soup (IMS), and experimentally validate its surprising ability to improve the quality of pre-trained models at marginal cost. Unlike model soups, IMS provides a unique opportunity to eliminate the requirement to generate multiple fully fine-tuned models to average, thereby restricting the computational complexity equivalent to the cost of fine-tuning a single model. Table 6 illustrates the performance comparison of IMS with respect to two model soup variants (uniform and greedy) proposed in (Wortsman et al., 2022a). Note that uniform and greedy soups results are generated using the amalgamation of 8 independent models fine-tuned till the final accuracy with different hyperparameters. Our experiments across CLIP and BERT illustrate that by carefully fine-tuning IMS, it is surprisingly possible to comfortably beat the model soup variants significantly. The denoised pre-trained model generated by IMS has the ability to converge to equivalent (even better) performance than model soups. Adhering to the theme of ISP, IMS also conveys a strong message that it is not necessarily important to wait till model convergence to ripe the benefits of soup, but astonishingly soup benefits are available to ripe early during the fine-tuning at a marginal computational cost.

Table 6. Fine-tuning performance comparison of our proposed approach (IMS) wrt. basic fine-tuning and model soup variants.

Approach	CLIP$_{ViT-B32}$			BERT$_{Base}$	
	Cars	CIFAR10	CIFAR100	MNLI	QNLI
Pretrained-BASE	76.43	97.60	89.35	82.39	90.04
Uniform Soup (Wortsman et al., 2022a)	76.32	97.68	89.20	82.41	89.76
Greedy Soup (Wortsman et al., 2022a)	77.95	98.05	89.54	83.01	90.64
IMS [Ours]	78.79	98.01	89.64	83.63	91.23
(std.)	± 0.32	± 0.07	± 0.12	± 0.43	± 0.19

Table 7. Performance comparison of ISP wrt. look-ahead on CIFAR100 with CLIP (ViT-B32) pruned at 60% sparsity.

Look-ahead Count(C)	10	30	50	100	150	200
Performance	84.71	84.92	85.18	85.19	84.41	85.41

4. Related Work

Linear interpolation of neural network weights has recently achieved significant attention, but due to numerous nonlinear activations within a neural network, it is still debatable if linearly interpolating between two sets of weights can result in a high accuracy solution. Recently, (Frankle et al., 2020; Nagarajan & Kolter, 2019; Von Oswald et al., 2020; Matena & Raffel, 2021; Wortsman et al., 2022a;b; Choshen et al., 2022; Izmailov et al., 2018; Neyshabur et al., 2020) have studied the interpolation of deep networks and validated performance benefits when training starts from a common initialization or some segment of the optimization trajectories are shared. While (Nagarajan & Kolter, 2019; Frankle et al., 2020) focused on mergability in the case of models trained on a single task, (Wortsman et al., 2022b) found that weight interpolation can not only benefit fine-tuning tasks but also under distribution shift. More specifically, they average zero-shot and fine-tuned models, finding improvements in- and out-of-distribution.

Recently, (Matena & Raffel, 2021) used Fisher-weighted averaging of language models before and after fine-tuning on downstream tasks. They merged models with the same pre-trained initialization that are fine-tuned on different text classification tasks. In the late phases of training, (Von Oswald et al., 2020) studied making copies of a subset of the neural network parameters and proposed to independently optimize them, followed by averaging. Moreover, (Wortsman et al., 2022a) proposed to average fine-tuned models across independent runs with hyperparameter diversity, modifying all the weights of the network, and showing significant performance benefits. In addition to model weight averaging, (Bansal et al., 2021; Yang et al., 2022) explored the idea of model stitching, where given two trained and frozen models A and B, a "stitched model" formed by connecting the bottom-layers of A to the top-layers of B, with a simple trainable layer between them. (Fort et al., 2019) studied deep ensembles which have empirically shown promise for improving the accuracy, uncertainty and out-of-distribution robustness of deep learning models.

5. Conclusion

In this work, we introduced Instant Soup Pruning, a model soup-inspired perspective dedicated to generating LTH quality subnetworks, using a fraction of the original IMP cost. ISP is augmented by a denoising pruning module which helps in replacing the expensive intermediate pruning stages of IMP with computationally efficient weak mask generation and aggregation routine. Additionally, we present Instant Model Soup, which provides an opportunity to inject the benefits of model soups in dense pre-trained models at marginal training cost, thereby improving fine-tuning performance comparable to model soups. Our future work will aim for a more theoretical understanding of the role of our denoisers in providing experimental benefits.

6. Acknowledgement

References

Bansal, Y., Nakkiran, P., and Barak, B. Revisiting model stitching to compare neural representations. *Advances in neural information processing systems*, 34:225–236, 2021. 8

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Wang, Z., and Carbin, M. The lottery ticket hypothesis for pretrained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020a. 2, 3, 7

Chen, T., Cheng, Y., Gan, Z., Yuan, L., Zhang, L., and Wang, Z. Chasing sparsity in vision transformers: An end-to-end exploration. *Advances in Neural Information Processing Systems*, 34:19974–19988, 2021a. 3

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Carbin, M., and Wang, Z. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16306–16316, 2021b. 3

Chen, T., Zhang, Z., JAISWAL, A. K., Liu, S., and Wang, Z. Sparse moe as the new dropout: Scaling dense and self-slimmable transformers. In *The Eleventh International Conference on Learning Representations*, 2023. 2

Chen, X., Cheng, Y., Wang, S., Gan, Z., Wang, Z., and Liu, J. Earlybert: Efficient bert training via early-bird lottery tickets. *arXiv preprint arXiv:2101.00063*, 2020b. 2, 3, 4

Choshen, L., Venezian, E., Slonim, N., and Katz, Y. Fusing finetuned models for better pretraining. *arXiv preprint arXiv:2204.03044*, 2022. 8

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 1

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1, 2

Ding, M., Zhou, C., Chen, Q., Yang, H., and Tang, J. Cognitive graph for multi-hop reading comprehension at scale. *arXiv preprint arXiv:1905.05460*, 2019. 1

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 6

Fang, G., Ma, X., Song, M., Mi, M. B., and Wang, X. Depgraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16091–16101, 2023. 2

Fort, S., Hu, H., and Lakshminarayanan, B. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019. 8

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. 2, 7

Frankle, J., Dziugaite, G. K., Roy, D., and Carbin, M. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pp. 3259–3269. PMLR, 2020. 8

Gan, Z., Chen, Y.-C., Li, L., Chen, T., Cheng, Y., Wang, S., Liu, J., Wang, L., and Liu, Z. Playing lottery tickets with vision and language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 652–660, 2022. 3

Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y., Yang, Z., Zhang, Y., and Tao, D. A survey on visual transformer. *ArXiv*, abs/2012.12556, 2020. 1

Ilharco, G., Wortsman, M., Gadre, S. Y., Song, S., Hajishirzi, H., Kornblith, S., Farhadi, A., and Schmidt, L. Patching open-vocabulary models by interpolating weights. *arXiv preprint arXiv:2208.05592*, 2022. 2, 5, 6

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018. 8

Jaiswal, A., Li, T., Zander, C., Han, Y., Rousseau, J. F., Peng, Y., and Ding, Y. Scalp-supervised contrastive learning for cardiopulmonary disease classification and localization in chest x-rays using patient metadata. In *2021 IEEE International Conference on Data Mining (ICDM)*, pp. 1132–1137. IEEE, 2021a. 1

Jaiswal, A., Tang, L., Ghosh, M., Rousseau, J., Peng, Y., and Ding, Y. Radbert-cl: Factually-aware contrastive learning for radiology report classification. *Proceedings of machine learning research*, 158:196–208, 2021b. 1

Jaiswal, A., Ashutosh, K., Rousseau, J. F., Peng, Y., Wang, Z., and Ding, Y. Ros-kd: A robust stochastic knowledge distillation approach for noisy medical imaging. *arXiv preprint arXiv:2210.08388*, 2022a. 2

Jaiswal, A., Ma, H., Chen, T., Ding, Y., and Wang, Z. Spending your winning lottery better after drawing it, 2022b. URL https://openreview.net/forum?id=O4dxuEsIo9S. 2

Jaiswal, A., Chen, T., Rousseau, J. F., Peng, Y., Ding, Y., and Wang, Z. Attend who is weak: Pruning-assisted medical image localization under sophisticated and implicit imbalances. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4987–4996, 2023. 1

Jaiswal, A. K., Ma, H., Chen, T., Ding, Y., and Wang, Z. Training your sparse neural network better with any mask. In *International Conference on Machine Learning*, pp. 9833–9844. PMLR, 2022c. 2

Juneja, J., Bansal, R., Cho, K., Sedoc, J., and Saphra, N. Linear connectivity reveals generalization strategies. *arXiv preprint arXiv:2205.12411*, 2022. 2, 5

Lee, N., Ajanthan, T., and Torr, P. H. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018. 2, 6, 7

Li, T., Shetty, S., Kamath, A., Jaiswal, A., Jiang, X., Ding, Y., and Kim, Y. Cancergpt: Few-shot drug pair synergy prediction using large pre-trained language models. *arXiv preprint arXiv:2304.10946*, 2023. 1

Liu, S., Chen, T., Zhang, Z., Chen, X., Huang, T., Jaiswal, A., and Wang, Z. Sparsity may cry: Let us fail (current) sparse neural networks together! *arXiv preprint arXiv:2303.02141*, 2023. 2

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 1

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021. 1

Mao, Z., Jaiswal, A., Wang, Z., and Chan, S. H. Single frame atmospheric turbulence mitigation: A benchmark study and a new physics-inspired transformer model. *ArXiv*, abs/2207.10040, 2022. 1

Matena, M. and Raffel, C. Merging models with fisherweighted averaging. *arXiv preprint arXiv:2111.09832*, 2021. 8

Nagarajan, V. and Kolter, J. Z. Uniform convergence may be unable to explain generalization in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019. 8

Neyshabur, B., Sedghi, H., and Zhang, C. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020. 8

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N. M., Ku, A., and Tran, D. Image transformer. In *ICML*, 2018. 1

Prasanna, S., Rogers, A., and Rumshisky, A. When bert plays the lottery, all tickets are winning. *arXiv preprint arXiv:2005.00561*, 2020. 3

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021. 1, 2, 6, 7

Talmor, A., Herzig, J., Lourie, N., and Berant, J. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018. 1

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and J'egou, H. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 1

Von Oswald, J., Kobayashi, S., Sacramento, J., Meulemans, A., Henning, C., and Grewe, B. F. Neural networks with late-phase weights. *arXiv preprint arXiv:2007.12927*, 2020. 8

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018. 1, 6

Wang, Z., Tsvetkov, Y., Firat, O., and Cao, Y. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *arXiv preprint arXiv:2010.05874*, 2020. 7

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022. 1

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019. 6, 7

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pp. 23965–23998. PMLR, 2022a. 2, 5, 8

Wortsman, M., Ilharco, G., Kim, J. W., Li, M., Kornblith, S., Roelofs, R., Lopes, R. G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7959–7971, 2022b. 8

Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., Li, M., and Lin, J. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718*, 2019a. 1

Yang, X., Zhou, D., Liu, S., Ye, J., and Wang, X. Deep model reassembly. *Advances in neural information processing systems*, 35:25739–25753, 2022. 8

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019b. 1

Yin, L., Liu, S., Meng, F., Huang, T., Menkovski, V., and Pechenizkiy, M. Lottery pools: Winning more by interpolating tickets without increasing training or inference cost. *arXiv preprint arXiv:2208.10842*, 2022. 2, 6, 7

You, H., Li, C., Xu, P., Fu, Y., Wang, Y., Chen, X., Baraniuk, R. G., Wang, Z., and Lin, Y. Drawing early-bird tickets: Towards more efficient training of deep networks. *arXiv preprint arXiv:1909.11957*, 2019. 2, 3, 4, 6, 7

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33: 5824–5836, 2020. 2, 6, 7

Yu, X., Liu, T., Wang, X., and Tao, D. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7370–7379, 2017. 2

Zheng, M., Gao, P., Zhang, R., Wang, X., Li, H., and Dong, H. End-to-end object detection with adaptive clustering transformer. *ArXiv*, abs/2011.09315, 2021. 1

Zheng, W., Sharan, S., Jaiswal, A. K., Wang, K., Xi, Y., Xu, D., and Wang, Z. Outline, then details: Syntactically guided coarse-to-fine code generation. *arXiv preprint arXiv:2305.00909*, 2023. 1