STARFISH: FAST ACCURACY RECOVERY IN PRUNED NETWORKS FROM INTERNAL STATE HEALING

Anonymous authors
Paper under double-blind review

ABSTRACT

Pruning is a process designed to reduce the number of weights in a large neural network. This can substantially speed up inference but might cause a considerable reduction in the model's accuracy, and thus it is usually followed by a healing process that regains some of the lost accuracy. In this paper, we propose a new healing method, *STARFISH*, that can recover (most of) the accuracy of any pruned network efficiently. The main idea of STARFISH is to optimize the pruned network to align with the original network's internal state representations using a tiny calibration set of unlabeled examples. For the case of removing 50% of the weights, STARFISH healing improves the recovered accuracy by up to 22% over the state-of-the-art methods on ViT-based networks. Its advantage is even more pronounced under aggressive pruning. For example, after eliminating 75% of the weights in a DeiT-B network for ImageNet, STARFISH uses only 0.4% of the number of training images as a calibration set and recovers 82% of the original dense accuracy, whereas competing recovery techniques reach only 40%. The paper includes numerous ablations demonstrating the advantages of STARFISH over other common healing and distillation methods. Open-source code will be made available upon acceptance.

1 INTRODUCTION

Modern neural networks achieve increasingly strong performance, often at the cost of a growing parameter count. This growth makes deployment expensive in terms of memory, latency, and energy consumption. *Pruning* addresses this problem by removing some weights, neurons, channels, attention heads, or other computational units from a pretrained model, producing a smaller and more efficient network while attempting to preserve the predictive performance of the original dense model.

When performing pruning, the desired *sparsity*, namely the fraction of removed weights, plays a key role. While low sparsity saves only a limited amount of space and computation, high sparsity pruning often causes a substantial drop in performance. Early pruning and recovery breakthroughs report high accuracy with very high sparsity, removing up to 98.5% of the weights in convolutional neural networks (O'Shea & Nash, 2015; Frankle et al., 2019). Yet, modern large-scale pruning and recovery methods struggle to restore accuracy after high-sparsity pruning.

Although optimal pruning may theoretically require no recovery (Malach et al., 2020; Ramanujan et al., 2020), practical pipelines often include a recovery procedure for the remaining weights, which is usually the most expensive part of the pipeline. This line of work started by recovering accuracy by fine-tuning or retraining the surviving weights, often through multiple steps of low-sparsity pruning and healing schedules until the target sparsity is reached (Han et al., 2015; Zhu & Gupta, 2017). The Lottery Ticket Hypothesis (Frankle & Carbin, 2018) inspired methods to instead rewind surviving weights to an early checkpoint before retraining (Frankle et al., 2019; Renda et al., 2020).

As models and datasets grow, these procedures require substantial computing resources. In addition, they require access to the original training data and knowledge of its randomness, which are often unrealistic in modern settings, where training details are typically treated as proprietary knowledge, even in open-source models. Thus, a line of work of efficiency-motivated methods, focuses on avoiding full retraining and uses local curvature restoration (Hassibi & Stork, 1992), which was later extended to approximate curvature restoration (Meng et al., 2024; Benbaki et al., 2023; Singh & Alistarh, 2020; Yu et al., 2022; Frantar & Alistarh, 2023). The latest recovery-focused methods aim

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

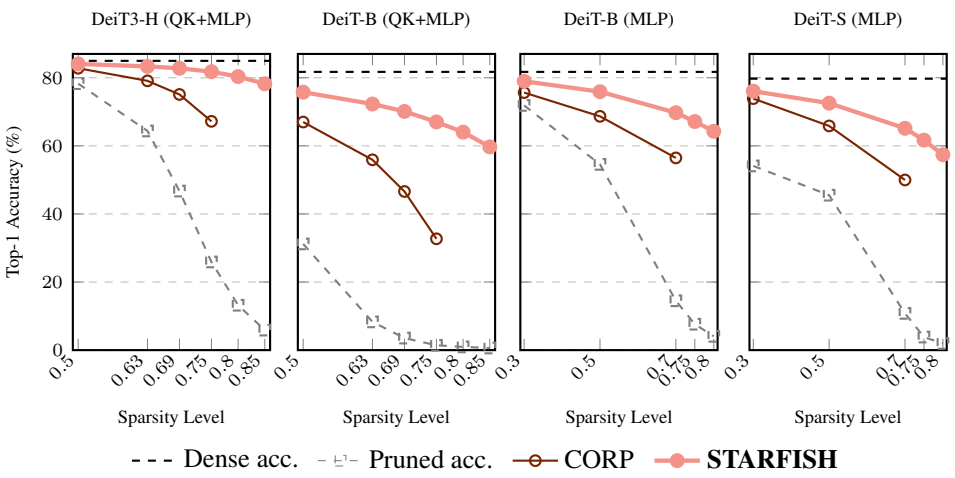


Figure 1: Comparison of the top-1 accuracy results after applying the STARFISH and CORP healing processes on different pruning scopes and sparsity levels for the DeiT-B, DeiT3-H, and DeiT-S architectures trained on ImageNet, both following CORP structured pruning. STARFISH recovery is more successful at all sparsity levels, reaching an even larger advantage as sparsity increases.

to repair the pruned model directly, using a small calibration set and fast local approximations based on the networks' hidden states. Two such methods are SNOWS (Lucas & Mazumder, 2024), which performs layerwise recovery by preserving downstream representations using Hessian approximation, and the very recent CORP (Zhang & Yang, 2026), which derives a closed-form representation recovery function for pruning in vision transformers.

In this paper, we present the **STARFISH** (fa**ST** **A**ccuracy **R**ecovery in pruned networks **F**rom **I**nternal **S**tate **H**ealing) method. The starfish is a marine animal with exceptional healing abilities that can regrow multiple arms even when completely cut off. Similarly, our STARFISH method excels at healing the performance even after the network undergoes aggressive pruning, regardless of the pruning method or type. STARFISH uses a small calibration set to quickly optimize the directional alignment between the intermediate representations of the pruned network and those of the original network, achieving remarkable accuracy restoration.

While intermediate feature matching is well established in knowledge distillation and representation learning (Romero et al., 2015; Sanh et al., 2019; Zagoruyko & Komodakis, 2017; Tian et al., 2020; Chen et al., 2020; Grill et al., 2020; Ericsson et al., 2022), the success of STARFISH shows that much information is retained in the pruned models' surviving weights. Rather than learning the dense model representations from scratch, STARFISH uses representation alignment to repair the computation disrupted by pruning. We introduce this simple post-pruning recovery mechanism, empirically showing that this objective is especially effective for high-sparsity pruning.

In Figure 1, we demonstrate this remarkable improvement for four modern transformer-based image classifiers (Touvron et al., 2021; 2022) at various sparsity levels: our STARFISH method is much better than the state-of-the-art recovery method CORP, especially in the most interesting regime of high sparsity pruning. In more detail, our main contributions are:

1. We present STARFISH, an efficient recovery method based on intermediate-representation alignment, which is independent of pruning types, methods, or network architecture. We empirically show that STARFISH is increasingly more effective at improving accuracy than other output-based or representation-based optimization as the sparsity level increases.
2. We show experimentally, on large image classifiers such as ViT-based architectures and MobileNet (Howard et al., 2017), that our method is the first to efficiently restore even 92% of the original accuracy at the very high sparsity level of 0.85, in which the pruning process almost completely degrades the original accuracy.
3. For the more modest sparsity of 0.5 across ViT (Dosovitskiy et al., 2020) and ViT-based architectures, we show that our method outperforms competing methods by a large factor of

up to 31.5 percentage-points in recovering the original network accuracy, reaching up to 99.8% retention of the dense model's accuracy.

4. We empirically show that STARFISH requires only a tiny unlabeled calibration set (up to 0.7% of the training set size), and is even tolerant of calibration sets drawn from different distributions.

2 RELATED WORK

Creating more efficient machine learning models, pruning involves removing weights from the network while preserving the original accuracy. While theoretical works demonstrate the existence of optimal pruning that requires no recovery (Ramanujan et al., 2020; Malach et al., 2020), much of the pruning literature focuses on optimizing the pruning method and using fine-tuning, or equivalent training-length methods for recovery (LeCun et al., 1989; Han et al., 2015; Zhu & Gupta, 2017; Zheng et al., 2022; Kuznedelev et al., 2023; Chen et al., 2025). Later works observed that rewinding the remaining weights to an early checkpoint and retraining works better (Frankle & Carbin, 2018; Renda et al., 2020). Since such recovery procedures can be computationally expensive, another line of work seeks to restore performance without full retraining or rewinding. A classic example is Optimal Brain Surgeon (OBS) (Hassibi & Stork, 1992), which updates the remaining weights to recover the lost curvature caused by the pruning.

Modern methods scale these principles to large models using efficient curvature estimates. WoodFisher (Singh & Alistarh, 2020) accelerates OBS by approximating the inverse Hessian with the empirical Fisher matrix, while CBS (Yu et al., 2022) improves efficiency further through simultaneous weight removal. These ideas were also extended to large language models (Frantar & Alistarh, 2023; Kwon et al., 2022). A further speedup introduced by CHITA (Benbaki et al., 2023) exploits the low rank of the Fisher matrix by reformulating pruning as an ℓ_0 sparse linear regression problem, while FALCON (Meng et al., 2024) extends this line of work with a FLOP-cost constraint.

As model sizes increase, even approximating the model's geometric properties becomes computationally expensive, motivating a shift to explicitly examining layerwise computations and outputs, known as hidden representations, using a small calibration set. SNOWS (Lucas & Mazumder, 2024) performs layerwise representation reconstruction using a Hessian approximation to perform a Newton descent optimization step, while CORP (Zhang & Yang, 2026) uses a one-shot approximation of an affine transformation between the original and pruned representations in vision transformers. Similarly, in compression recovery methods, DC-ViT (Zhang et al., 2024) and Muralidharan et al. (2024) match the teacher and student representations at selected blocks. STARFISH extends these ideas by directionally aligning the intermediate representations at every block, using knowledge from the surviving weights for fast recovery rather than relearning.

Representations and learning. Representation information has been widely used in knowledge distillation (Hinton et al., 2015), providing stronger supervision than output matching alone, including hidden-state hints, attention maps, language-model hidden-state alignment, and contrastive feature matching (Romero et al., 2015; Zagoruyko & Komodakis, 2017; Sanh et al., 2019; Tian et al., 2020). A related line of work uses representation for the more general self-supervised and unsupervised learning, where models learn useful embeddings by aligning related views, target networks, or even neural network weights without relying on manual labels (Chen et al., 2020; Grill et al., 2020; Ericsson et al., 2022). More broadly, representation similarity measures, such as cosine-based comparisons, have been shown to better capture cross-network similarities (Kornblith et al., 2019).

3 METHOD

Given a neural network, referred to as the dense model, pruning is the process of sparsifying its parameters in order to reduce memory, computation, or inference cost. Let $\mathcal{D}$ denote the data distribution from which we sample the labeled data $(\mathbf{x}, y) \in \mathbb{R}^d \times [K]$, where K is the number of classes. Let $f_{\boldsymbol{\theta}} : \mathbb{R}^d \to \mathbb{R}^K$ be a pretrained K-class classifier with parameters $\boldsymbol{\theta}$, for which we are given a binary pruning mask $\mathbf{m} \in \{0,1\}^{|\boldsymbol{\theta}|}$ resulting in a pruned model $f_{\boldsymbol{\theta}\odot\mathbf{m}}$, where $\odot$ denotes elementwise multiplication. For any classifier $g : \mathbb{R}^d \to \mathbb{R}^K$, predictions are given by

$\arg\max_{k\in[K]} g(\mathbf{x})_k$, and we define

$$\mathrm{Acc}_{\mathcal{D}}(g) = \Pr_{(\mathbf{x},y)\sim\mathcal{D}}\left[\arg\max_{k\in[K]} g(\mathbf{x})_k = y\right].$$

As pruning typically degrades performance, the recovery goal is to restore this lost performance without changing the mask. In particular, we seek to obtain parameters $\widetilde{\boldsymbol{\theta}}$ such that the recovered pruned model $f_{\widetilde{\boldsymbol{\theta}}\odot\mathbf{m}}$ matches the predictive accuracy of the dense model

$$\mathrm{Acc}_{\mathcal{D}}(f_{\widetilde{\boldsymbol{\theta}}\odot\mathbf{m}}) \approx \mathrm{Acc}_{\mathcal{D}}(f_{\boldsymbol{\theta}}).$$

3.1 The STARFISH algorithm

We present the STARFISH method, which encourages the pruned model to align with the dense model's intermediate representations across the network, by a simple optimization on a small calibration set. Let $S_{\mathrm{cal}} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a small unlabeled calibration set such that for all i, $\mathbf{x}_i \notin S_{\mathrm{train}}$, and $|S_{\mathrm{cal}}| \ll |S_{\mathrm{train}}|$. First, we run the dense model on S_{cal} and cache the target intermediate representations. Then, we train $f_{\boldsymbol{\theta}\odot\mathbf{m}}$ to align with those cached representations.

For a formal algorithm description, let L be the number of blocks in $f_{\boldsymbol{\theta}}$. Given an input $\mathbf{x} \in \mathbb{R}^d$, for each $\ell \in [L]$, we define $\mathbf{r}^{\ell}_{\mathbf{x}}$ as the vectorized output of the dense model $f_{\boldsymbol{\theta}}$ after block ℓ (i.e., taken after all operations that form the block output). Denote by $f^{\ell}_{\boldsymbol{\theta}\odot\mathbf{m}}(\mathbf{x})$ the corresponding ℓ-th representation of the pruned model $f_{\boldsymbol{\theta}\odot\mathbf{m}}$. We define the alignment loss as

$$\mathcal{L}_{\mathrm{align}}(\boldsymbol{\theta}\odot\mathbf{m}, \mathbf{x}) = \frac{1}{L}\sum_{\ell\in[L]}\left(1 - \frac{\left\langle f^{\ell}_{\boldsymbol{\theta}\odot\mathbf{m}}(\mathbf{x}), \mathbf{r}^{\ell}_{\mathbf{x}}\right\rangle}{\left\|f^{\ell}_{\boldsymbol{\theta}\odot\mathbf{m}}(\mathbf{x})\right\| \left\|\mathbf{r}^{\ell}_{\mathbf{x}}\right\|}\right), \tag{1}$$

and train the pruned model $f_{\boldsymbol{\theta}\odot\mathbf{m}}$ by minimizing the empirical loss over S_{cal}, namely,

$$L(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i\in[n]} \mathcal{L}_{\mathrm{align}}\left(\boldsymbol{\theta}\odot\mathbf{m}, \mathbf{x}_i\right).$$

The STARFISH recovery method offers two significant advantages over existing methods. First, the number of samples in S_{cal} is extremely small compared to the train set, and does not need to be drawn from it. This is particularly desirable in modern ML settings, where the trained model might be open source, while the original training set is not necessarily accessible. Second, while different recovery methods are often dependent and even coupled with a specific pruning method, STARFISH adjusts the weights after the damage has occurred, independently of the pruning. This allows STARFISH to be easily applicable to any pruning method, sparsity pattern, pruning scope, and architecture.

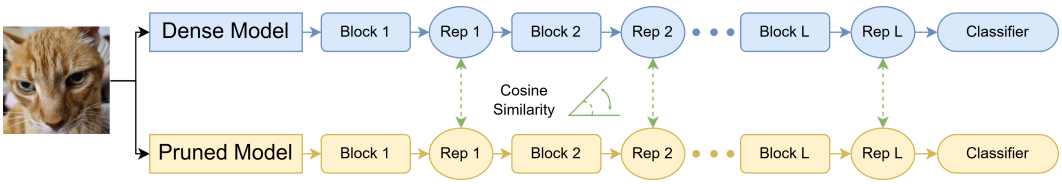


Figure 2: Visualization of the representation computation and alignment of STARFISH, where the cosine similarity is computed using the hidden representation taken after each corresponding computation block.

3.2 Computational complexity

Compared to similar optimization-based recovery methods that use the entire training set, STARFISH is extremely efficient, utilizing a small calibration set for a one-time, dense forward pass to cache target representations, followed by a few recovery epochs of the pruned model.

Specifically, let T be the number of recovery epochs, S_{cal} be the calibration set of size $S_{\text{cal}} = n$, P be the cost of one dense forward or backward pass, and $\widetilde{P}$ be the equivalent for the pruned network. Since the loss computation is negligible, the time complexity of the STARFISH method is

$$O\left(nP + Tn\widetilde{P}\right).$$

In this paper, across all architectures and comparisons, we use small values of $n \leq 9{,}000$ and $T \leq 10$.

Comparison with other efficient recovery methods. CORP (Zhang & Yang, 2026) uses a closed-form layerwise recovery for a restricted ViT sparsity pattern. Under this restricted setting, we analyze the complexity where ρ is the sparsity level, i.e., $(1-\rho)$-fraction of weights remain active and optimally, $\widetilde{P} = (1-\rho)P$. Let L be the number of blocks, Q be the ViT sequence length and d_t be a single token dimension. Its cost including the dense calibration pass is

$$O\left(nP + L(1-\rho)^3 d_t^3 + nQL(1-\rho)^2 d_t^2\right).$$

Where $\left(L(1-\rho)^3 d_t^3 + nQL(1-\rho)^2 d_t^2\right)$ is the overhead of computing the closed form solution, which is comparable to a single forward or backward pass. Thus its overall complexity is comparable to STARFISH. SNOWS (Lucas & Mazumder, 2024) is more general and, like STARFISH, can recover a fixed-mask model by matching intermediate representations. However, SNOWS performs local Hessian approximation computation using gradients for every C-layer sliding window, and for I Newton steps, thus its ViT recovery cost is approximately $O\left(ICn\widetilde{P}\right)$.

In practice, on a single A10 GPU, STARFISH recovers a 0.8-sparse DeiT3-H to 73% accuracy in 16 minutes using 1,000 examples and 5 epochs. With only 500 examples—less than one example per class—recovery takes 8 minutes and still outperforms competing methods evaluated at lower sparsity levels. Additional results are in Appendix B.6.

4 Representation Alignment

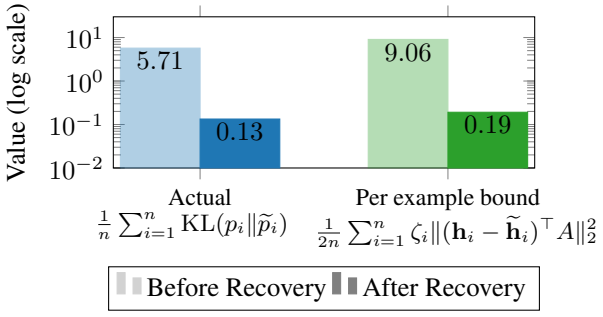


Figure 3: Empirical visualization of the bound for KL divergence before and after recovery. After STARFISH recovery, the representation error decreases, and with it both the empirical KL and the corresponding bound decrease sharply.

In this section, we discuss the advantage of intermediate representation similarity optimization over regular output-based optimization. The STARFISH method treats pruning recovery as internal-state recovery, rather than matching only the final outputs of the dense and pruned models, thus encouraging the pruned model to follow the dense model's computation throughout the network. We motivate the representation-based optimization by proving that the distance between the final hidden representations of the dense and recovered models upper-bounds the classification head's output KL divergence. This remains relevant even when the classification head itself is unpruned, since pruning earlier layers changes its input.

Formally, for an input $\mathbf{x}_i \in S_{\text{cal}}$, we denote by $\mathbf{h}_i, \widetilde{\mathbf{h}}_i \in \mathbb{R}^{d_{\text{out}}}$ the last hidden representations (i.e., the input of the classification head) of the dense and recovered pruned models, respectively. Let $A \in \mathbb{R}^{d_{\text{out}} \times K}$ and $\widetilde{A} \in \mathbb{R}^{d_{\text{out}} \times K}$ be the dense and recovered pruned classification heads, where K is the number of classes, and denote by $\mathbf{p}_i, \widetilde{\mathbf{p}}_i$ the corresponding predictive distribution vectors. We prove that

$$\frac{1}{n}\sum_{i=1}^{n} \mathrm{KL}(\mathbf{p}_i \,\|\, \widetilde{\mathbf{p}}_i) \leq \frac{1}{2n}\sum_{i=1}^{n} \zeta_i \left(\|(\mathbf{h}_i - \widetilde{\mathbf{h}}_i)^\top A\|_2 + \|\widetilde{\mathbf{h}}_i^\top (\widetilde{A} - A)\|_2\right)^2.$$

for some small parameters ζ_i, dependent on the network's normalized output logits. The formal theorem and full proof are given in Appendix A.1. Figure 3 illustrates the bound in the common case of an unpruned classification head following a high-sparsity pruned network. On a magnitude-pruned DeiT-B model at 0.8-sparsity, STARFISH recovery sharply reduces both representation error and predictive KL divergence.

We note that our choice of cosine-similarity is motivated by prior work using cosine-based criteria for representation alignment and comparison (Sanh et al., 2019; Kornblith et al., 2019; Chen et al., 2020), applied to every block to avoid accumulated errors. Appendices B.2 and B.3 shows that aligning all blocks with cosine similarity outperforms MSE and partial-depth alternatives.

4.1 Recovery objective and pretrained initialization

To further motivate the STARFISH recovery method, we experimentally show that it performs as an alignment and recovery method rather than as a learning or knowledge distillation method. We first show that aligning with the dense model representations yields a more effective approach than learning with output-level supervision. We then emphasize the significance of the knowledge encoded in the pruned weights by comparing STARFISH recovery with knowledge-distillation methods.

In Table 1, we consider output supervision with the baseline cross-entropy (CE) loss and the Kullback–Leibler (KL) divergence. We compare both, as well as the joint STARFISH+KL objective, on a magnitude-pruned DeiT-B. STARFISH consistently outperforms the other objectives, peaking at 0.8-sparsity, where it achieves 73.66 accuracy, exceeding KL by 10.11 points, CE by 16.08 points, and STARFISH+KL by 4.59 points. Further implementation details provided in Appendices A.2 and B.

Table 1: Comparison of top-1 accuracy after performing recovery using representation-based and output-based objectives, on a pruned DeiT-B (Dense Acc.: 81.73%) at different sparsity levels. Values in parentheses indicate accuracy retention relative to the dense accuracy. The representation-based recovery surpasses all output-based objectives, as well as the combined alternative.

Method/Sparsity	**0.5**	**0.6**	**0.7**	**0.8**
Pruned Acc.	74.59 (91.26%)	56.16 (68.71%)	20.02 (24.49%)	0.83 (1.02%)
Cross-Entropy (CE)	79.11 (96.79%)	76.51 (93.61%)	70.87 (86.71%)	57.58 (70.45%)
Kullback–Leibler (KL)	80.64 (98.67%)	79.87 (97.72%)	75.70 (92.62%)	63.55 (77.76%)
STARFISH+KL	81.34 (99.52%)	79.87 (97.72%)	77.59 (94.93%)	69.07 (84.51%)
STARFISH	**81.41 (99.61%)**	**80.55 (98.56%)**	**78.95 (96.60%)**	**73.66 (90.13%)**

Figure 4 highlights that STARFISH does not aim to learn the dense model's representations but rather to repair information already retained in the pruned pretrained network. This is particularly striking at 0.8-sparsity, where pruning reduces the model to only 0.83 accuracy, yet applying STARFISH to this severely damaged model dramatically recovers accuracy to 73.28. In contrast, when we consider the smaller model architecture and apply STARFISH as knowledge distillation from the dense model, recovery remains at only 0.099, creating a nearly three-orders-of-magnitude gap. This experiment demonstrates that even when the pruned model's accuracy has collapsed, its surviving trained weights contain the critical information needed to restore accuracy, whereas the target dense representations alone are insufficient for learning with such limited time complexity and calibration set.

5 Experiments

We evaluate STARFISH on ImageNet-1K (Deng et al., 2009) across vision transformers and convolutional networks, showing it outperforms other efficient recovery methods at high sparsity as well as the 0.5-sparsity level. In the following experiments, STARFISH uses up to $9{,}000$ unlabeled images from the ImageNet test split (only 0.7% of the training-set size) and does not use labels or revisit the training set. We show the remarkable abilities of STARFISH across a broad range of pruning settings.

To demonstrate the independence of the STARFISH method from pruning type or method, we compare it with multiple competing methods across various pruning variations. The first variation is

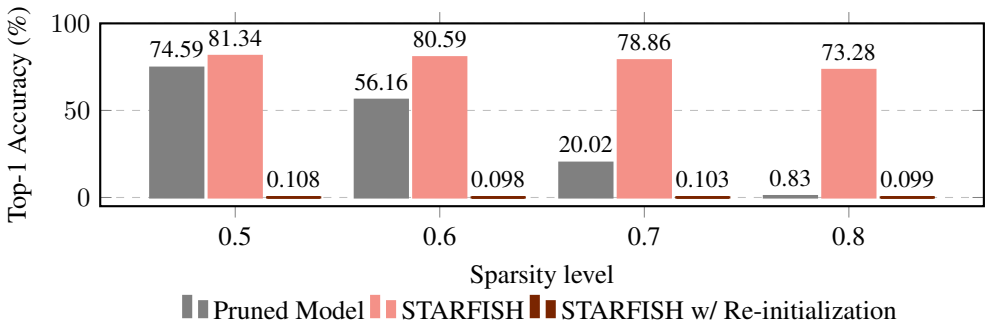


Figure 4: Comparison of top-1 accuracy after STARFISH recovery from pretrained versus reinitialized pruned weights. Reinitialization yields near-chance accuracy across sparsity levels, demonstrating the importance of preserving the pretrained initialization.

the sparsity pattern, including *unstructured* pruning, which removes individual weights; *structured* pruning, which removes groups such as channels, neurons, heads, or blocks; and *semi-structured* pruning, such as N:M sparsity, keeping N weights in each group of M parameters. The second is the pruning scope. In transformers, Q, K, and V denote the Query, Key, and Value projection weights; Out denotes the output projection of the attention block; and MLP denotes the following feed-forward block. When pruning selected components, the reported sparsity refers to those prunable weights.

5.1 High sparsity STARFISH

In this section, we present STARFISH in severe pruning regimes, where recovery is most challenging and the differences between recovery methods are most pronounced. We compare STARFISH across sparsity levels on both MobileNetV1 and DeiT models. The advantage of STARFISH grows with sparsity: the gains become substantially larger once 70-85% of the prunable weights are removed, where other pruning and recovery methods present smaller improvements.

In Figure 1, we compare STARFISH in structured pruning of various DeiT models. STARFISH systematically surpasses the state-of-the-art CORP recovery method (Zhang & Yang, 2026), recovering models pruned using the CORP mask selection method over a range of sparsity levels and scopes. For example, STARFISH recovers a 0.85-sparse DeiT3-H model, and retains 92.08% of the dense model accuracy, successfully increasing the pruned accuracy by 72.36 points. Recovering a 0.75-sparse DeiT-B model, STARFISH achieves accuracy with a staggering 34.33 percentage-point improvement over CORP.

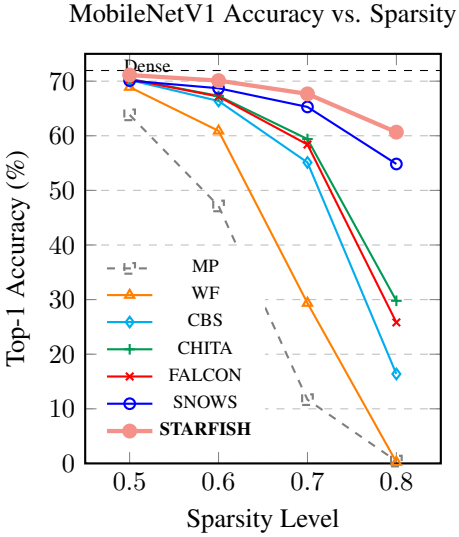


Figure 5: Comparison of top-1 accuracy results of unstructured pruning and recovery methods on MobileNetV1. STARFISH recovery exceeds other methods, within a larger margin at high sparsity levels.

A similar pattern is observed in Figure 5, where we compare STARFISH to several leading unstructured pruning methods on MobileNetV1. STARFISH consistently outperforms the other methods across all settings. STARFISH and SNOWS (Lucas & Mazumder, 2024) use Magnitude Pruning masks, with unrecovered MP as the baseline. Further details are provided in Appendix B.

5.2 HALF SPARSITY STARFISH

In this section, we compare STARFISH with prior recovery methods in the 0.5-sparsity regime, showing that STARFISH achieves the highest recovered accuracy in every setting we evaluate. Across different vision transformers models, we consider both structured pruning as well as 2:4 semi-structured pruning in different pruning scopes.

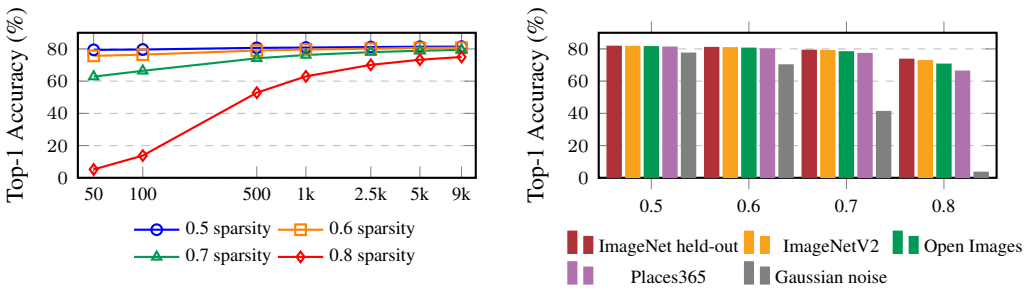
In Panel A of Table 2, we compare STARFISH to the CORP recovery method (Zhang & Yang, 2026), under 0.5-sparsity structured pruning across various DeiT models, with both performed after CORP-pruning. The comparison spans multiple pruning scopes, and the gains are especially large in the more challenging joint MLP+QK setting, where pruning affects the most components. For example, in the DeiT-T MLP+QK setting, pruning reduces accuracy from 72.01 to 7.57, STARFISH recovers the model to 81.95% of the dense accuracy, compared with 50.41% retention for CORP. This is a 31.54 percentage-point improvement in retention.

In Panel B, we compare STARFISH with SNOWS (Lucas & Mazumder, 2024), under 2:4 semi-structured sparsity on ViT models, with both methods performed after Magnitude Pruning. These results show that STARFISH consistently improves recovered accuracy beyond structured pruning and also improves recovery under hardware-friendly semi-structured sparsity. Further implementation details and extended results are in Appendices B and B.1.

5.3 CALIBRATION DATA

As the lightweight STARFISH method requires only a small unlabeled calibration set, Figure 6 studies its size and distribution using an unstructured magnitude-pruned DeiT-B of the QKV+Out+MLP parameters. In Figure 6a, we show that at 0.5-sparsity, only 50 samples are needed for STARFISH to yield 79.42 accuracy. At higher sparsity, more data is needed, although the gains saturate: at 0.8-sparsity, increasing the set from 5,000 to 9,000 improves accuracy by merely 1.69 points.

Next, Figure 6b shows that STARFISH remains effective without access to the original training data. With 5,000 examples at 0.8-sparsity, ImageNetV2, Open Images, and Places365 achieve 72.40, 70.24, and 65.93, respectively, versus 73.28 for held-out ImageNet. These results demonstrate that STARFISH works effectively with access to other natural images, which are considerably easier to obtain. Additional calibration ablations are provided in Appendices B.4 and B.5.



(a) Effect of calibration-set size.

(b) Effect of calibration-data distribution.

Figure 6: Calibration-data ablations, showing that size and distributional relevance matter increasingly at high sparsity. **Left:** recovery versus calibration-set size at different sparsity levels. **Right:** recovery using 5,000 samples from different distributions.

6 CONCLUSION AND FUTURE DIRECTIONS

We presented STARFISH, a post-pruning recovery method that treats sparse-model repair as internal-state healing. Given a fixed pruning mask, STARFISH updates only the remaining weights so that the sparse model's intermediate representations match those of the dense model across all blocks. This simple objective is modular across pruning methods, sparsity patterns, pruning scopes, sparsity levels, and architectures. STARFISH uses a tiny unlabeled calibration set drawn outside the training set, making recovery efficient and requiring only the dense model, pruning mask, and calibration data.

Table 2: Comparison of top-1 accuracy results of different recovery methods. Retention is recovered accuracy relative to dense accuracy; Improvement is the retention gap between methods, showing that STARFISH achieves large improvement gaps over competing methods across settings. Best result within each model and scope is bolded. **Panel A:** STARFISH vs. CORP under structured DeiT pruning. **Panel B:** STARFISH vs. SNOWS under 2:4 ViT pruning.

Model	Pruning Scope	Method	Calib. Size	Recovered Acc.	Retention (%)	Improvement (Δ%)
Panel A: DeiT / CORP setting (0.5 **structured sparsity)**						
DeiT-T (Dense Acc. 72.01)	MLP (Pruned Acc. 10.93)	CORP	–	53.20	73.88%	**+10.51%**
		STARFISH	1,000	**60.77**	**84.39%**	
	MLP+QK (Pruned Acc. 7.57)	CORP	–	36.30	50.41%	**+31.54%**
		STARFISH	1,000	**59.01**	**81.95%**	
DeiT-S (Dense Acc. 79.72)	MLP (Pruned Acc. 45.51)	CORP	–	65.85	82.60%	**+8.39%**
		STARFISH	2,500	**72.54**	**90.99%**	
	MLP+QK (Pruned Acc. 17.32)	CORP	–	55.50	69.62%	**+20.18%**
		STARFISH	2,500	**71.59**	**89.80%**	
DeiT-B (Dense Acc. 81.73)	MLP (Pruned Acc. 54.56)	CORP	–	68.67	84.02%	**+8.86%**
		STARFISH	5,000	**75.91**	**92.88%**	
	MLP+QK (Pruned Acc. 31.19)	CORP	–	67.00	81.98%	**+10.73%**
		STARFISH	5,000	**75.77**	**92.71%**	
DeiT3-L (Dense Acc. 84.58)	MLP (Pruned Acc. 77.81)	CORP	–	80.90	95.65%	**+2.74%**
		STARFISH	7,000	**83.22**	**98.39%**	
	MLP+QK (Pruned Acc. 72.09)	CORP	–	79.40	93.88%	**+3.89%**
		STARFISH	7,000	**82.69**	**97.77%**	
DeiT3-H (Dense Acc. 84.97)	MLP (Pruned Acc. 81.79)	CORP	–	83.70	98.51%	**+0.82%**
		STARFISH	9,000	**84.40**	**99.33%**	
	MLP+QK (Pruned Acc. 78.27)	CORP	–	82.80	97.45%	**+1.48%**
		STARFISH	9,000	**84.06**	**98.93%**	
Panel B: ViT / SNOWS setting (2:4 **semi-structured sparsity)**						
ViT/B-16 (Dense Acc. 80.40)	QKV (Pruned Acc. 78.92)	SNOWS	5,000	79.45	98.82%	**+0.94%**
		STARFISH	5,000	**80.21**	**99.76%**	
	QKV+Out+MLP (Pruned Acc. 70.86)	SNOWS	5,000	76.57	95.24%	**+3.11%**
		STARFISH	5,000	**79.07**	**98.35%**	
ViT/L-16 (Dense Acc. 84.20)	QKV (Pruned Acc. 75.40)	SNOWS	20,000	81.01	96.21%	**+2.32%**
		STARFISH	7,000	**82.96**	**98.53%**	
	MLP (Pruned Acc. 0.20)	SNOWS	20,000	69.71	82.79%	**+3.27%**
		STARFISH	7,000	**72.46**	**86.06%**	

Empirically, STARFISH consistently improves recovery across all settings tried. In moderate pruning regimes, STARFISH substantially improves dense-accuracy retention; at 0.5-sparsity on DeiT-T, STARFISH recovers 81.95% of the dense accuracy, a 31.54-point retention improvement over competitors. Under severe pruning, STARFISH remains effective: at 0.85-sparsity on DeiT3-H, STARFISH recovers to 78.24 top-1 accuracy, improving the pruned model by 72.36 points.

Future work could extend STARFISH in several directions. One such direction is adaptive block weighting, where the recovery objective could emphasize layers that are more damaged by pruning. Representation mismatch could also be used to guide the pruning process itself, rather than only to repair the model afterward. Finally, this idea could be incorporated directly into training by encouraging a model and its induced sparse counterpart to maintain aligned internal representations. Such a sparsity-aware representation objective may produce dense models whose pruned versions are easier to recover, or even remain performant immediately after pruning.

AI USE STATEMENT

In this work, we used generative AI tools to assist with identifying potentially relevant related work, in addition to literature searches conducted by the authors. We used it for language editing, including grammar, spelling and clarity, and for implementing standard methods used in the experimental code. All AI-found literature suggestions were manually checked by the authors to see if relevant before cited. All AI-written code was reviewed and tested by the authors, and all AI-assisted edits were manually reviewed and edited accordingly. We did not use AI to develop the research ideas or conceptual framework of this work, formulate or refine hypotheses, design the proposed method or experiments, formulate mathematical claims, develop or write proofs, or interpret the experimental results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the details needed to reproduce STARFISH and our experiments throughout the paper and appendix. The STARFISH objective and recovery procedure are specified in Section 3, while Appendix B provides the data splits, model checkpoints, trainable parameters, representation specifications, recovery hyperparameters, calibration-set sizes, batch sizes, random-seed protocol, and details of the pruning masks and comparison settings. All reported results are averaged over three independent random seeds. Source code will be made publicly available upon acceptance.

REFERENCES

Riade Benbaki, Wenyu Chen, Xiang Meng, Hussein Hazimeh, Natalia Ponomareva, Zhe Zhao, and Rahul Mazumder. Fast as chita: Neural network pruning with combinatorial optimization. In *International Conference on Machine Learning*, pp. 2031–2049. PMLR, 2023.

Shaowu Chen, Wei Ma, Binhua Huang, Qingyuan Wang, Guoxin Wang, Weize Sun, Lei Huang, and Deepu John. Optimal brain connection: Towards efficient structural pruning. *arXiv preprint arXiv:2508.05521*, 2025.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pp. 1597–1607. PMLR, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

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.

Linus Ericsson, Henry Gouk, Chen Change Loy, and Timothy M Hospedales. Self-supervised representation learning: Introduction, advances, and challenges. *IEEE Signal Processing Magazine*, 39(3):42–62, 2022.

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

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*, 2019.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pp. 10323–10337. PMLR, 2023.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in Neural Information Processing Systems*, 28, 2015.

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

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pp. 3519–3529. PMLR, 2019.

Aditya Kusupati, Vivek Ramanujan, Raghav Somani, Mitchell Wortsman, Prateek Jain, Sham Kakade, and Ali Farhadi. Soft threshold weight reparameterization for learnable sparsity. In *International conference on machine learning*, pp. 5544–5555. PMLR, 2020.

Denis Kuznedelev, Eldar Kurtić, Elias Frantar, and Dan Alistarh. Cap: Correlation-aware pruning for highly-accurate sparse vision models. *Advances in Neural Information Processing Systems*, 36: 28805–28831, 2023.

Woosuk Kwon, Sehoon Kim, Michael W Mahoney, Joseph Hassoun, Kurt Keutzer, and Amir Gholami. A fast post-training pruning framework for transformers. *Advances in Neural Information Processing Systems*, 35:24101–24116, 2022.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in Neural Information Processing Systems*, 2:598–605, 1989.

Ryan Lucas and Rahul Mazumder. Preserving deep representations in one-shot pruning: A hessian-free second-order optimization framework. *arXiv preprint arXiv:2411.18376*, 2024.

Eran Malach, Gilad Yehudai, Shai Shalev-Schwartz, and Ohad Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*, pp. 6682–6691. PMLR, 2020.

Xiang Meng, Wenyu Chen, Riade Benbaki, and Rahul Mazumder. Falcon: Flop-aware combinatorial optimization for neural network pruning. In *International Conference on Artificial Intelligence and Statistics*, pp. 4384–4392. PMLR, 2024.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation. *Advances in Neural Information Processing Systems*, 37:41076–41102, 2024.

Pravin Nair. Softmax is $1/2$-lipschitz: A tight bound across all ℓ_p norms. *arXiv preprint arXiv:2510.23012*, 2025.

Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11893–11902, 2020.

Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. *arXiv preprint arXiv:2003.02389*, 2020.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets, 2015. URL `https://arxiv.org/abs/1412.6550`.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Sidak Pal Singh and Dan Alistarh. Woodfisher: Efficient second-order approximation for neural network compression. *Advances in Neural Information Processing Systems*, 33:18098–18109, 2020.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations (ICLR)*, 2020.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *European Conference on Computer Vision*, pp. 516–533. Springer, 2022.

Xin Yu, Thiago Serra, Srikumar Ramalingam, and Shandian Zhe. The combinatorial brain surgeon: Pruning weights that cancel one another in neural networks. In *International Conference on Machine Learning*, pp. 25668–25683. PMLR, 2022.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations (ICLR)*, 2017.

Boxiang Zhang and Baijian Yang. Corp: Closed-form one-shot representation-preserving structured pruning for vision transformers. *arXiv preprint arXiv:2602.05243*, 2026.

Hanxiao Zhang, Yifan Zhou, and Guo-Hua Wang. Dense vision transformer compression with few samples. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15825–15834. IEEE, 2024.

Chuanyang Zheng, Kai Zhang, Zhi Yang, Wenming Tan, Jun Xiao, Ye Ren, Shiliang Pu, et al. Savit: Structure-aware vision transformer pruning via collaborative optimization. *Advances in Neural Information Processing Systems*, 35:9010–9023, 2022.

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.

A Representation alignment (Section 4)

A.1 Representation bound for KL divergence

We state the definitions and the theorem from Section 4. For an input $\mathbf{x}_i \in S_{\text{cal}}$, we denote by $\mathbf{h}_i, \widetilde{\mathbf{h}}_i \in \mathbb{R}^{d_{\text{out}}}$ the last hidden representations (i.e., the input of the classification head) of the dense and recovered pruned models, respectively. Let $A \in \mathbb{R}^{d_{\text{out}} \times K}$ denote the dense classification head, where K is the number of classes, and let $\widetilde{A} \in \mathbb{R}^{d_{\text{out}} \times K}$ denote the recovered pruned head. We denote the output logits vectors and their difference by

$$\mathbf{z}_i = \mathbf{h}_i^\top A \qquad \widetilde{\mathbf{z}}_i = \widetilde{\mathbf{h}}_i^\top \widetilde{A} \qquad \Delta\mathbf{z}_i = \widetilde{\mathbf{z}}_i - \mathbf{z}_i,$$

and the corresponding predictive distribution vectors by

$$\mathbf{p}_i = \mathrm{softmax}(\mathbf{z}_i), \qquad \widetilde{\mathbf{p}}_i = \mathrm{softmax}(\widetilde{\mathbf{z}}_i).$$

Theorem 1 (Representation and head recovery controls predictive KL)**.** *For a calibration set $S_{cal} = \{\mathbf{x}_i\}_{i=1}^n$, we denote* [1]

$$M_i = \frac{\|(\widetilde{\mathbf{h}}_i - \mathbf{h}_i)^\top A\|_2}{\|(\widetilde{\mathbf{h}}_i - \mathbf{h}_i)^\top\|_2}, \qquad \widetilde{M}_i = \frac{\|\widetilde{\mathbf{h}}_i^\top (\widetilde{A} - A)\|_2}{\|\widetilde{\mathbf{h}}_i^\top\|_2}, \qquad \zeta_i = \sup_{t \in [0,1]} \frac{\Delta\mathbf{z}_i^\top \nabla^2 \psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i}{\|\Delta\mathbf{z}_i\|_2^2} \tag{2}$$

where $\psi(u) = \log \sum_{k=1}^K e^{u_k}$ is the log-sum-exp function. Then,

$$\frac{1}{n}\sum_{i=1}^n \mathrm{KL}(\mathbf{p}_i \,\|\, \widetilde{\mathbf{p}}_i) \le \frac{1}{2n}\sum_{i=1}^n \zeta_i \left(M_i \|(\mathbf{h}_i - \widetilde{\mathbf{h}}_i)^\top\|_2 + \widetilde{M}_i \|\widetilde{\mathbf{h}}_i^\top\|_2 \right)^2.$$

In particular, if the classification head is not pruned, so that $\widetilde{A} = A$, then $\widetilde{M}_i = 0$ and

$$\frac{1}{n}\sum_{i=1}^n \mathrm{KL}(\mathbf{p}_i \,\|\, \widetilde{\mathbf{p}}_i) \le \frac{1}{2n}\sum_{i=1}^n \zeta_i M_i^2 \|(\mathbf{h}_i - \widetilde{\mathbf{h}}_i)^\top\|_2^2.$$

Proof. We start by showing that $\nabla\psi = \mathrm{softmax}$.
Let $\mathbf{u} \in \mathbb{R}^K$. By the chain rule, for each $j \in [K]$ we have

$$\frac{\partial \psi}{\partial u_j}(\mathbf{u}) = \frac{1}{\sum_{k=1}^K e^{u_k}} \cdot \frac{\partial}{\partial u_j}\left(\sum_{k=1}^K e^{u_k}\right) = \frac{e^{u_j}}{\sum_{k=1}^K e^{u_k}}.$$

which is exactly $\mathrm{softmax}(\mathbf{u})_j$. Recall that the KL divergence between distributions $\mathbf{p}$ and $\mathbf{q}$ over $[K]$ is defined as

$$\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}) = \sum_{k \in [K]} p_k \log \frac{p_k}{q_k}$$

[1] If a denominator in the definitions in (2) is zero, then so is the numerator and we define the corresponding ratio to be zero.

and the Bregman divergence of a differentiable function f is defined as

$$D_f(x, y) = f(x) - f(y) - \langle \nabla f(y), x - y \rangle$$

Let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^K$. We define

$$\mathbf{p} = \nabla\psi(\mathbf{u}), \qquad \mathbf{q} = \nabla\psi(\mathbf{v}),$$

the softmax distributions. We aim to prove that

$$\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}) = D_\psi(\mathbf{v}, \mathbf{u})$$

Indeed, since $\nabla\psi(\mathbf{u}) = \mathrm{softmax}(\mathbf{u}) = \mathbf{p}$, we have $p_k = e^{u_k} / \sum_{j\in[K]} e^{u_j}$. Taking logarithms,

$$\log p_k \;=\; \log \frac{e^{u_k}}{\sum_{j\in[K]} e^{u_j}} \;=\; u_k - \log \sum_{j\in[K]} e^{u_j} \;=\; u_k - \psi(\mathbf{u}).$$

Likewise,

$$\log q_k \;=\; v_k - \psi(\mathbf{v}).$$

We have,

$$\begin{aligned}
\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}) \;&=\; \sum_{k=1}^{K} p_k \big(\log \frac{p_k}{q_k}\big) \\
&= \sum_{k=1}^{K} p_k \big(\log p_k - \log q_k\big) \\
&= \sum_{k=1}^{K} p_k \Big[\big(u_k - \psi(\mathbf{u})\big) - \big(v_k - \psi(\mathbf{v})\big)\Big] \\
&= \sum_{k=1}^{K} p_k \,(u_k - v_k) \;+\; \big(\psi(\mathbf{v}) - \psi(\mathbf{u})\big) \sum_{k=1}^{K} p_k.
\end{aligned}$$

Since p is a distribution on $[K]$, we have $\sum_{k=1}^{K} p_k = 1$. Thus

$$\begin{aligned}
\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}) \;&=\; \sum_{k=1}^{K} p_k \,(u_k - v_k) \;+\; \psi(\mathbf{v}) - \psi(\mathbf{u}) \\
&= \;\psi(\mathbf{v}) \;-\; \psi(\mathbf{u}) \;-\; \sum_{k=1}^{K} p_k \,(v_k - u_k)
\end{aligned}$$

By definition $\mathbf{p} = \nabla\psi(\mathbf{u})$, hence

$$\sum_{k=1}^{K} p_k \,(v_k - u_k) \;=\; \langle \nabla\psi(\mathbf{u}), \mathbf{v} - \mathbf{u} \rangle.$$

Substituting, we finally get

$$\begin{aligned}
\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}) \;&=\; \psi(\mathbf{v}) - \psi(\mathbf{u}) \;-\; \langle \nabla\psi(\mathbf{u}), \mathbf{v} - \mathbf{u} \rangle \\
&=\; D_\psi(\mathbf{v}, \mathbf{u})
\end{aligned}$$

We substitute $\mathbf{z}_i$ for $\mathbf{u}$ and $\widetilde{\mathbf{z}}_i$ for $\mathbf{v}$ and since $\nabla\psi(\mathbf{z}_i) = \mathbf{p}_i$ and $\nabla\psi(\widetilde{\mathbf{z}}_i) = \widetilde{\mathbf{p}}_i$ we get

$$\mathrm{KL}(\mathbf{p}_i \,\|\, \widetilde{\mathbf{p}}_i) \;=\; D_\psi(\widetilde{\mathbf{z}}_i, \mathbf{z}_i) \tag{3}$$

The multivariate version of Taylor's theorem tells us that for a twice continuously-differentiable function $f : \mathbb{R}^K \to \mathbb{R}$ at point $\mathbf{y} \in \mathbb{R}^K$ we have $\forall \mathbf{h} \in \mathbb{R}^K$

$$f(\mathbf{y} + \mathbf{h}) = f(\mathbf{y}) + \langle \nabla f(\mathbf{y}), \mathbf{h} \rangle + \int_0^1 (1 - t)\, \mathbf{h}^\top \nabla^2 f(\mathbf{y} + t\mathbf{h}) \mathbf{h}\, dt.$$

We apply this formula to the function ψ, since it is twice continuously-differentiable, with

$$\mathbf{y} = \mathbf{z}_i, \qquad \mathbf{h} = \Delta\mathbf{z}_i.$$

Thus $\mathbf{y} + \mathbf{h} = \widetilde{\mathbf{z}}_i$, and we have

$$\psi(\widetilde{\mathbf{z}}_i) = \psi(\mathbf{z}_i) + \langle \nabla\psi(\mathbf{z}_i), \Delta\mathbf{z}_i \rangle + \int_0^1 (1-t)\, \Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i \, dt.$$

Rearranging gives

$$\psi(\widetilde{\mathbf{z}}_i) - \psi(\mathbf{z}_i) - \langle \nabla\psi(\mathbf{z}_i), \Delta\mathbf{z}_i \rangle = \int_0^1 (1-t)\, \Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i \, dt.$$

From what we showed in (3), we get

$$\mathrm{KL}(\mathbf{p}_i \,\|\, \widetilde{\mathbf{p}}_i) \;=\; \psi(\widetilde{\mathbf{z}}_i) - \psi(\mathbf{z}_i) \;-\; \langle \nabla\psi(\mathbf{z}_i), \Delta\mathbf{z}_i \rangle \tag{4}$$

$$\;=\; \int_0^1 (1-t)\, \Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i \, dt. \tag{5}$$

By the definition of the supremum, for every $t \in [0,1]$ we have

$$\frac{\Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i}{\|\Delta\mathbf{z}_i\|_2^2} \le \zeta_i.$$

Multiplying both sides by $\|\Delta\mathbf{z}_i\|_2^2$ gives

$$\Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i \le \zeta_i \|\Delta\mathbf{z}_i\|_2^2.$$

Therefore,

$$\begin{aligned}
&\int_0^1 (1-t)\, \Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i \, dt \\
&\qquad \le \int_0^1 (1-t)\, \zeta_i \|\Delta\mathbf{z}_i\|_2^2 \, dt \\
&\qquad = \zeta_i \|\Delta\mathbf{z}_i\|_2^2 \int_0^1 (1-t)\, dt \\
&\qquad = \frac{\zeta_i}{2} \|\Delta\mathbf{z}_i\|_2^2.
\end{aligned}$$

In particular, if we use the fact that the softmax function is $\frac{1}{2}$ Lipschitz (Nair, 2025), we get that its Jacobian is bounded by $\frac{1}{2}$. But its Jacobian is the Hessian of the log-sum-exp function, thus for every $t \in [0,1]$,

$$\Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i \le \frac{1}{2} \|\Delta\mathbf{z}_i\|_2^2.$$

Hence

$$\zeta_i \le \frac{1}{2},$$

and consequently

$$\int_0^1 (1-t)\, \Delta\mathbf{z}_i^\top \nabla^2\psi(\mathbf{z}_i + t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i \, dt \le \frac{1}{4} \|\Delta\mathbf{z}_i\|_2^2.$$

From (4) we get

$$\mathrm{KL}(\mathbf{p}_i \,\|\, \widetilde{\mathbf{p}}_i) \;\le\; \frac{1}{2}\zeta_i \|\Delta\mathbf{z}_i\|_2^2 \;\le\; \frac{1}{4} \|\Delta\mathbf{z}_i\|_2^2. \tag{6}$$

It remains to relate the logit error to representation and head recovery. Since

$$\Delta z_i \;=\; \widetilde{\mathbf{h}}_i^\top \widetilde{A} \;-\; \mathbf{h}_i^\top A \;=\; (\widetilde{\mathbf{h}}_i - \mathbf{h}_i)^\top A \;+\; \widetilde{\mathbf{h}}_i^\top (\widetilde{A} - A),$$

the triangle inequality gives

$$\|\Delta\mathbf{z}_i\|_2 \le \|(\widetilde{\mathbf{h}}_i - \mathbf{h}_i)^\top A\|_2 + \|\widetilde{\mathbf{h}}_i^\top (\widetilde{A} - A)\|_2.$$

We note that we can use the sub-multiplicativity of the norm and get

$$\|\Delta\mathbf{z}_i\|_2 \le \|A\|_2 \|(\widetilde{\mathbf{h}}_i - \mathbf{h}_i)\|_2 + \|(\widetilde{A} - A)\|_2 \|\widetilde{\mathbf{h}}_i\|_2.$$

But using the definitions in (2) we can get

$$\|\Delta\mathbf{z}_i\|_2 \le M_i \|(\mathbf{h}_i - \widetilde{\mathbf{h}}_i)\|_2 + \widetilde{M}_i \|\widetilde{\mathbf{h}}_i\|_2. \tag{7}$$

Substituting into (6) and averaging over the calibration examples concludes the proof. □

We note that using the two global bounds introduced in the proof gives us instead the following bound:

$$\frac{1}{n}\sum_{i=1}^{n}\mathrm{KL}(\mathbf{p}_i \,\|\, \widetilde{\mathbf{p}}_i) \le \frac{1}{4n}\sum_{i=1}^{n}\left(\|A\|_2\|(\mathbf{h}_i-\widetilde{\mathbf{h}}_i)^\top\|_2 + \|\widetilde{A}-A\|_2\|\widetilde{\mathbf{h}}_i^\top\|_2\right)^2.$$

The global bound is valid, but can be overly conservative in practice. Indeed, the proof above keeps two example-dependent quantities: M_i, which measures the gain of the classification head in the realized representation-error direction, and ζ_i, which measures the curvature of the log-sum-exp function along the corresponding path. Both quantities admit worst-case global upper bounds, namely $M_i \le \|A\|_2$ and $\zeta_i \le \frac{1}{2}$. Substituting these worst-case constants yields the global bound shown in Figure 7. Although this bound still upper-bounds the empirical KL divergence, it is much looser than the local, batch-dependent bound.

Figure 8 visualizes the empirical distributions of M_i and ζ_i over the calibration examples and compares them to their corresponding global upper bounds. The figure shows that the realized values are substantially smaller than their worst-case bounds: the head gain M_i is typically well below $\|A\|_2$, and the local curvature ζ_i is far below the worst-case smoothness constant $\frac{1}{2}$.

In Figures 3, 7, and 8, we empirically approximate ζ_i by taking the maximum over $\frac{\Delta\mathbf{z}_i^\top\nabla^2\psi(\mathbf{z}_i+t\Delta\mathbf{z}_i)\Delta\mathbf{z}_i}{\|\Delta\mathbf{z}_i\|_2^2}$ for 20 points in $[0,1]$.

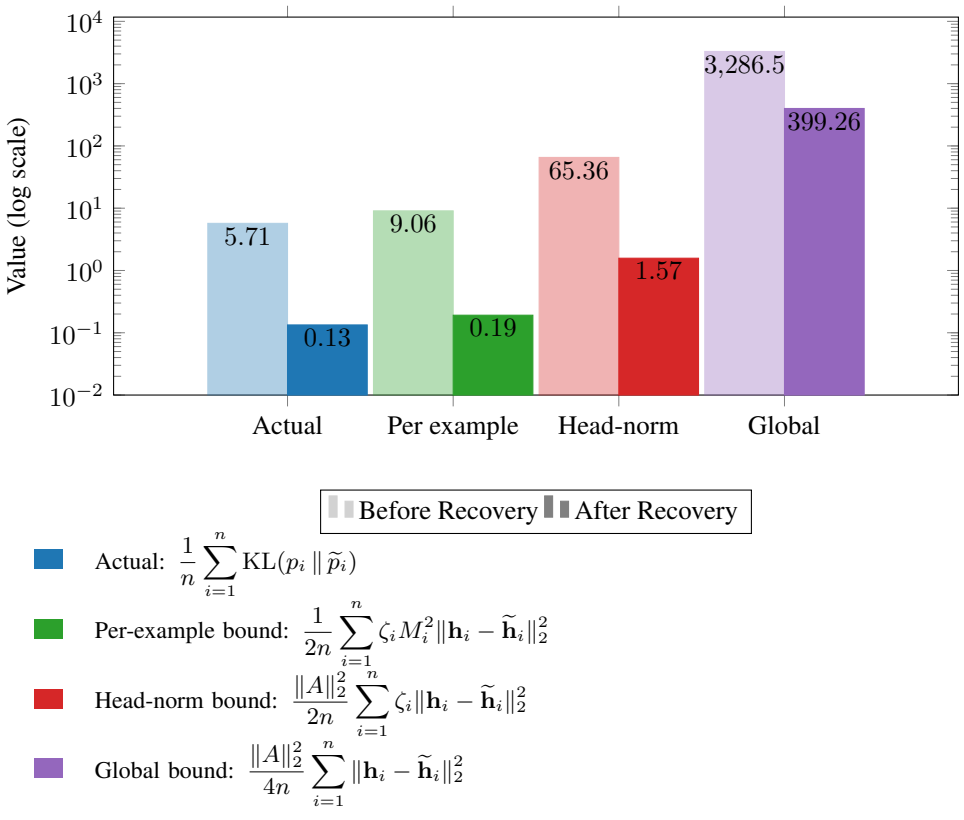


Figure 7: Empirical verification of the KL divergence bound. Comparing both the global bounds as well as the data-dependent bounds. It can be seen that STARFISH recovery improves all bounds, and that the per-example one remains the tightest.

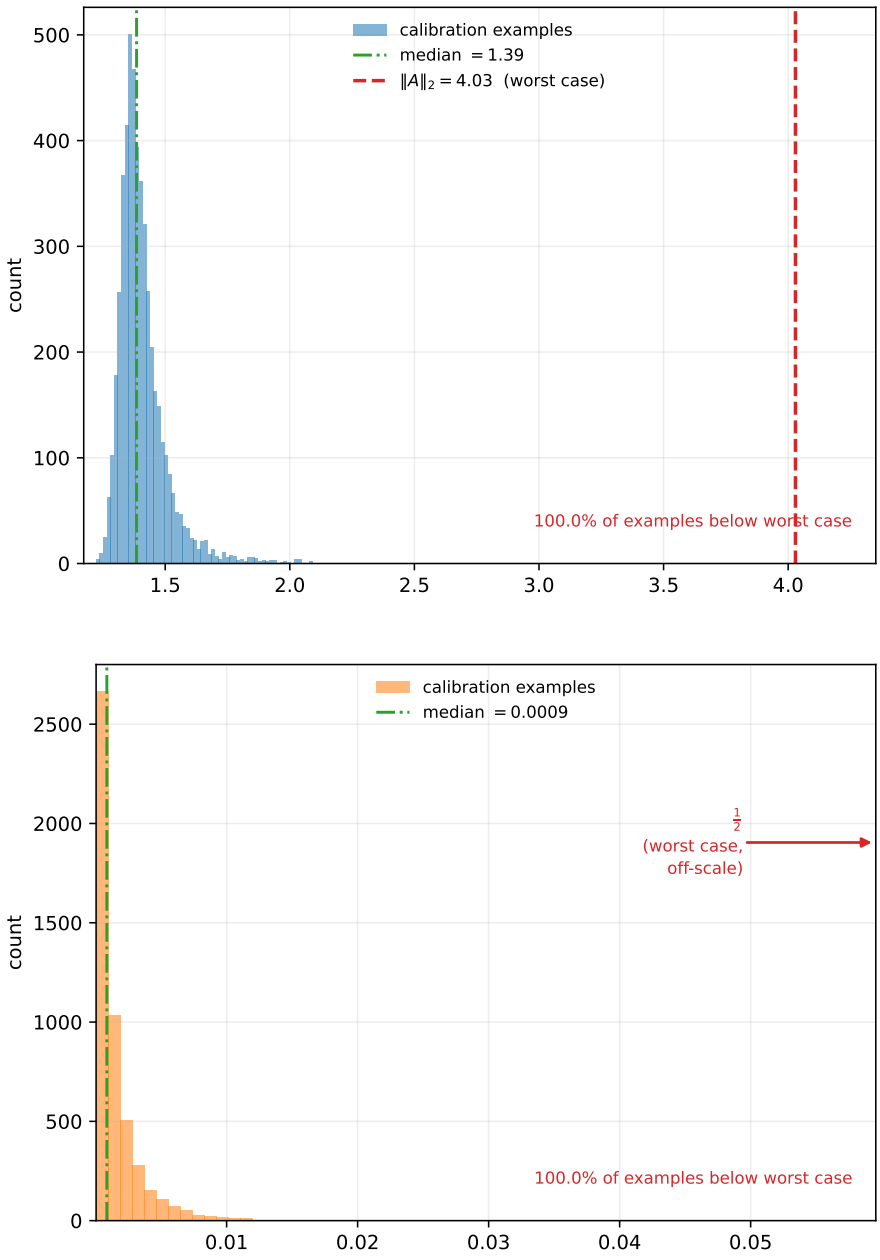


Figure 8: Visualization of the data-dependent constants in the local KL bound compared to their global worst-case upper bounds. **Top**: Distribution of the example-wise head gains M_i over the calibration examples, compared to the global bound $\|A\|_2$. **Bottom**: Distribution of the example-wise curvature constants ζ_i, compared to the global smoothness bound $\frac{1}{2}$. In both cases, the realized data-dependent quantities are substantially smaller than their worst-case bounds, explaining why the local batch-dependent KL bound in Figure 7 is much tighter than the corresponding global bound.

A.2 Recovery objective and pretrained initialization: further experiment and details

In Section 4.1, we compared STARFISH to standard output-level recovery objectives on DeiT-B under unstructured pruning. Here, we provide an additional comparison to CE in structured pruning of various DeiT models and pruning scopes.

Figure 9 compares STARFISH to CE (i.e., standard fine-tuning) after applying CORP structured pruning to several DeiT architectures and pruning scopes. Across all models and sparsity levels, STARFISH outperforms CE. The gap is especially pronounced at the hardest settings, the joint QK+MLP setting in the two leftmost graphs, where the label-based CE provides only a weak recovery signal.

The CE and KL baselines reported in Table 1 and fig. 9 use images from the ImageNet training set. We use the same recovery budget as STARFISH, including the same number of epochs and batch size, which are disclosed in Appendix B. For these experiments, for each recovery objective and sparsity level, we tune the learning rate and weight decay independently, sweeping learning rates over $10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 6 \times 10^{-4}$ and weight decay over $0, 10^{-4}, 10^{-2}$. In all cases, we use a cosine learning-rate schedule with a final learning rate of 10^{-6}. We apply the same tuning procedure to the comparisons in both Table 1 and fig. 9,selecting the best configuration. The reported results are averaged over three independent seeds.

The experiments reported in Table 1 and fig. 4 reports recovery after unstructured pruning on the QKV+Out+MLP parameters of a DeiT-B model. The STARFISH+KL objective combines the two losses with equal weights of 0.5 and uses a calibration set drawn from the test set. For the re-initialization experiment, the only difference between the two methods is whether we reinitialize the surviving weights or keep the pruned models weights before we apply the STARFISH recovery method. In both cases we preserve the pruning mask.

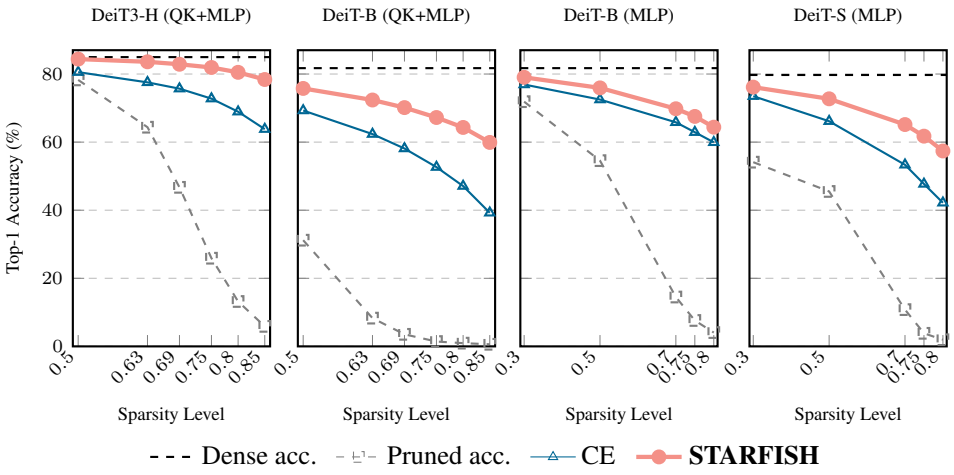


Figure 9: Comparison of the top-1 accuracy results after the STARFISH and fine-tuning healing processes, on different structured pruning scopes and sparsity levels for the DeiT-B, DeiT3-H, and DeiT-S architectures trained on ImageNet, performed following CORP pruning. One can see the STARFISH method achieves high results in all settings, especially in high-sparsity levels.

B Experimental details for Section 5

Data splits. Unless otherwise stated, all calibration images used for STARFISH recovery are sampled from the unlabeled ImageNet-1K test split, streamed from Hugging Face.[2] No labels are used during STARFISH recovery. Evaluation is performed on the ImageNet-1K validation split, also streamed from Hugging Face.[2] Thus, the calibration images used for representation alignment are disjoint from the labeled validation images used for reporting top-1 accuracy.

Model checkpoints. All dense model checkpoints are loaded from publicly available pretrained weights. For ViT/B-16 and ViT/L-16, we use the torchvision checkpoints `ViT_B_16_Weights.IMAGENET1K_V1` and

[2] `https://huggingface.co/datasets/ILSVRC/imagenet-1k`

`ViT_L_16_Weights.IMAGENET1K_SWAG_LINEAR_V1`, respectively.[3][4] For DeiT and DeiT3 models, we use the pretrained checkpoints provided through `timm` by calling `timm.create_model(model_name, pretrained=True)`.[5] Specifically, we use the non-distilled DeiT variants `deit_tiny_patch16_224`, `deit_small_patch16_224`, and `deit_base_patch16_224`, and the DeiT3 variants `deit3_large_patch16_224`, and `deit3_huge_patch14_224`. For MobileNetV1, we use the dense STR (Kusupati et al., 2020) checkpoint provided by the WoodFisher repository.[6]

Trainable parameters during recovery. During STARFISH recovery, the pruning mask is kept fixed and we update only the surviving weights in the tensors that were designated as prunable by the corresponding pruning method. All parameters outside the prunable set are frozen at their dense-model values, and all masked-out weights remain fixed at zero throughout recovery. Equivalently, if $\mathcal{P}$ denotes the set of prunable parameter tensors and $\mathbf{m}$ is the binary pruning mask, then optimization is performed only over entries θ_j such that $\theta_j \in \mathcal{P}$ and $m_j = 1$. Entries with $m_j = 0$ are not updated, and parameters not belonging to $\mathcal{P}$ are excluded from the optimizer. In implementation, we enforce this constraint by applying the mask after each optimizer step, ensuring that the sparsity pattern is unchanged throughout recovery.

Representation specifications. For ViTs, that is the post-block token matrix – taken after the attention and MLP sublayers, including their residual additions. For MobileNetV1, it is the output of the depthwise-separable convolution block, taken after the point-wise convolution, BatchNorm, and ReLU.

Recovery optimization. For DeiT-T, DeiT-S, DeiT-B, ViT/B-16, and MobileNetV1, STARFISH recovery is run for 10 epochs. For DeiT3-L, DeiT3-H, and ViT/L-16, STARFISH is run for 5 epochs. Recovery uses AdamW with initial learning rate 6×10^{-4}, decayed to 10^{-6} with a cosine learning-rate schedule. All results reported in the paper are averaged over three independent random seeds. Across seeds, randomness comes from the sampled calibration subset and the ordering of calibration batches. If not otherwise stated, we used $1{,}000$ examples for DeiT-T, $2{,}500$ examples for DeiT-S, $5{,}000$ examples for DeiT-B and ViT/B-16, $7{,}000$ examples for DeiT3-L and ViT/L-16 and $9{,}000$ examples for DeiT3-H. We used a batch size of 32 for all models except MobilenetV1, for which we used a batch size of 64.

CORP masks. For comparisons with CORP, we use the CORP recovery results as reported in the original CORP paper. To evaluate STARFISH under the same pruning setting, we need CORP-style pruning masks. Since the official CORP code was not publicly available at the time of our experiments, we implemented only the CORP mask-selection procedure and used it to generate masks for STARFISH recovery. The mask-selection calibration set has the same size as the calibration set used later for STARFISH recovery, but is sampled from the ImageNet-1K training split streamed from Hugging Face.[2] After mask selection, STARFISH recovery is performed using the unlabeled ImageNet-1K test-split calibration images described above. The pruned accuracies of our CORP-mask implementation and the STARFISH recovery results are averaged over three independent random seeds.

ImageNet-mini evaluation for SNOWS comparisons. All SNOWS results reported in prior work are evaluated on ImageNet-1K-mini rather than on the full ImageNet-1K validation split. Therefore, for the comparisons against SNOWS in Panel B of Table 2, we also evaluate STARFISH on ImageNet-1K-mini, using the same evaluation setting in order to ensure a fair comparison. All other ImageNet-1K results in the paper are evaluated on the full ImageNet-1K validation split unless explicitly stated otherwise.

[3] `https://docs.pytorch.org/vision/stable/models/generated/torchvision.models.vit_b_16.html`

[4] `https://docs.pytorch.org/vision/stable/models/generated/torchvision.models.vit_l_16.html`

[5] `https://huggingface.co/docs/timm/reference/models`

[6] `https://github.com/IST-DASLab/WoodFisher/tree/main/checkpoints`

B.1 Extended 0.5-sparsity results

We report extended results for Table 2 in Table 3, which additionally includes the QK-only pruning setting in Panel A. STARFISH also outperforms the competing method in this setting, despite QK-only pruning being the least damaging setting, as reflected by the higher pruned-model accuracy.

B.2 Ablation study: cosine-similarity is better than MSE

STARFISH uses cosine similarity to align the dense and pruned intermediate representations. A natural alternative is to replace the cosine loss with a mean-squared-error (MSE) loss between the same representations. In Table 4, we compare these two objectives for recovering a pruned DeiT-B model pruned under unstructured sparsity on QKV+Out+MLP. Empirically, cosine alignment works better than MSE alignment. The difference is most pronounced at 0.8-sparsity, with a difference in final accuracy of 2.7 points. Accordingly, we use cosine alignment as the default STARFISH objective throughout the paper.

Table 4: Comparison of top-1 accuracy results of using cosine similarity vs. MSE as an objective for recovering a pruned DeiT-B at different sparsity levels. We show the cosine similarity loss presents better recovery results

	Dense Acc.: 81.73%			
Method/Sparsity	**0.5**	**0.6**	**0.7**	**0.8**
Pruned Acc.	74.59	56.16	20.02	0.83
MSE	81.15	80.18	78.17	70.56
Cosine Similarity	**81.34**	**80.59**	**78.86**	**73.28**

B.3 Ablation study: alignment throughout the network

STARFISH aligns the dense and pruned representations after every block. We next ablate whether recovery can instead rely on a single representation target. We use the same DeiT-B setting with unstructured magnitude pruning of QKV+Out+MLP.

First, we compare STARFISH with an objective that matches only the final representation using Frobenius MSE. As shown in Table 5, STARFISH consistently achieves higher recovered accuracy, and the gap grows substantially with sparsity. At 0.5 sparsity, the difference is only 0.41 percentage points, whereas at 0.8 sparsity STARFISH reaches 73.28 compared with 62.09 for final-representation matching, an improvement of 11.19 points. This suggests that under severe pruning, supervision of the final representation alone is insufficient to repair errors accumulated throughout the network.

Table 5: Top-1 accuracy when recovering a pruned DeiT-B using STARFISH alignment throughout the network versus matching only the final representation using Frobenius MSE. The benefit of intermediate alignment increases with sparsity.

	Dense Acc.: 81.73%			
Method/Sparsity	**0.5**	**0.6**	**0.7**	**0.8**
Pruned Acc.	74.59	56.16	20.02	0.83
STARFISH	**81.34**	**80.59**	**78.86**	**73.28**
Final-representation Frobenius MSE	80.93	79.93	77.71	62.09

We next isolate the importance of aligning multiple intermediate representations by restricting the STARFISH objective to a single late block. We separately align Blocks 9, 10, and 11, rather than averaging the alignment loss over all blocks. Table 6 shows that aligning all blocks consistently outperforms each single-block alternative. The difference again becomes larger at high sparsity: at 0.8 sparsity, STARFISH obtains 73.28 accuracy, whereas the best single-block variant, Block 10,

reaches 71.34. Thus, the gain does not arise solely from choosing one particularly informative late representation, supervision distributed throughout the network provides additional recovery signal.

Table 6: Top-1 accuracy when applying representation alignment throughout the network or restricting the recovery objective to a single late block. Aligning all blocks consistently provides the strongest recovery.

	Dense Acc.: 81.73%			
Method/Sparsity	**0.5**	**0.6**	**0.7**	**0.8**
Pruned Acc.	74.59	56.16	20.02	0.83
STARFISH	**81.34**	**80.59**	**78.86**	**73.28**
Block 11	81.00	80.05	78.00	70.38
Block 10	81.03	80.14	78.15	71.34
Block 9	81.09	80.13	77.86	70.62

B.4 Ablation study: calibration data source

STARFISH only requires unlabeled calibration inputs, and does not rely on access to the original training set. To verify that the recovery is not limited to the source of the calibration data when sampled from the same distribution, we compare using calibration images sampled from the training set versus the test set, while keeping the calibration size and recovery procedure fixed.

Table 7 reports the results for both structured DeiT pruning, pruned using CORP, and 2:4 semi-structured ViT pruning, pruned using MP, at 0.5-sparsity. The recovered accuracies obtained from train and test calibration images are very similar, with slightly better results for using the test set as the calibration source.

Table 7: Top-1 accuracy after STARFISH recovery when the unlabeled calibration examples are sampled from the training set versus a held-out test set at 0.5-sparsity. **Panel A:** Structured pruning results on DeiT models. **Panel B:** 2:4 semi-structured pruning results on ViT models. Best result within each model is bolded. Showing the calibration set from the unlabeled test set improves recovery.

Model	Pruning Scope	Calib. Set Source	Calib. Size	Recovered Acc.
Panel A: DeiT setting (0.5 structured sparsity)				
DeiT-T (Dense Acc. 72.01)	MLP+QK (Pruned Acc. 7.57)	Train	1,000	58.75
		Test		**59.01**
DeiT-S (Dense Acc. 79.72)	MLP+QK (Pruned Acc. 17.32)	Train	2,500	71.25
		Test		**71.59**
DeiT-B (Dense Acc. 81.73)	MLP+QK (Pruned Acc. 31.19)	Train	5,000	74.95
		Test		**75.77**
DeiT3-L (Dense Acc. 84.58)	MLP+QK (Pruned Acc. 72.09)	Train	7,000	82.11
		Test		**82.69**
DeiT3-H (Dense Acc. 84.97)	MLP+QK (Pruned Acc. 78.27)	Train	9,000	83.90
		Test		**84.06**
Panel B: ViT setting (2:4 semi-structured sparsity)				
ViT/B-16 (Dense Acc. 80.40)	QKV+Out+MLP (Pruned Acc. 70.86)	Train	5,000	79.05
		Test		**79.07**
ViT/L-16 (Dense Acc. 84.20)	MLP (Pruned Acc. 0.20)	Train	7,000	70.88
		Test		**72.46**

B.5 Ablation study: calibration fidelity and diversity.

We further study how the fidelity and diversity of the calibration inputs affect the STARFISH recovery method.

In Table 8 we recover a pruned DeiT-B model pruned using unstructured magnitude pruning on QKV+Out+MLP and vary only the conditions of the calibration set used. Gaussian blurring uses $\sigma = 3$ with a 19×19 kernel. The low-resolution condition reduces the shorter image edge to 56 pixels before applying the standard preprocessing. The redundant condition repeats 500 unique images ten times, preserving $5{,}000$ optimization examples while reducing sample diversity. The degradation caused by lower fidelity or diversity is modest at 0.5 sparsity but substantially larger at 0.8 sparsity. These results indicate that severe pruning benefits more strongly from informative and diverse inputs.

Table 8: Effect of calibration-input quality and diversity on STARFISH recovery. All conditions contain $5{,}000$ optimization examples and results are three-seed mean top-1 accuracies for unstructured-pruned DeiT-B. We show that using clean images becomes more important as the difficulty of recovery arises.

	Dense Acc.: 81.73%			
Calibration condition/Sparsity	**0.5**	**0.6**	**0.7**	**0.8**
Pruned Acc.	74.59	56.16	20.02	0.83
Clean	**81.34**	**80.59**	**78.86**	**73.28**
Gaussian-blurred	80.87	79.72	76.86	67.44
Low resolution	79.43	77.26	71.61	55.73
Redundant	80.74	79.69	77.13	67.81

B.6 Runtime

STARFISH provides a direct runtime–accuracy tradeoff when using an even smaller calibration set. In Table 9, we recover DeiT3-L and DeiT3-H models pruned to 0.8-sparsity using CORP structured masks on the QK+MLP pruning scope, and vary only the number of calibration examples used by STARFISH. It is no surprise that, for smaller calibration sets, runtime scales approximately linearly with the calibration set size, since the number of recovery epochs and batch size are fixed across all runs.

Even very small calibration sets already provide substantial recovery. For example, on DeiT3-H, STARFISH with only 1,000 calibration examples runs in about 16 minutes and recovers almost 86% of the original dense model accuracy. Using larger calibration sets further improves recovery: with 9,000 examples, STARFISH reaches 94.55% accuracy retention in the same DeiT3-H setting. This demonstrates that STARFISH can be used either as a very fast recovery method with modest calibration data, or as a stronger recovery method when more calibration time is available. The runtime results were measured using a single NVIDIA A10 GPU with 24GB memory.

Table 9: Runtime and final top-1 accuracy for STARFISH recovery on DeiT3-L and DeiT3-H under different calibration set sizes. Runtime is reported as mean $\pm$ sample standard deviation over three runs. It can be seen that runtime scales almost linearly, and that even using small calibration sets STARFISH recovers to a substantial accuracy.

Model	Calib. Size	Runtime	Recovered Acc.	Retention (%)
DeiT3-L (Dense Acc. 84.58)	500	3m 10s $\pm$ 10s	54.98	65.00%
	1,000	5m 41s $\pm$ 12s	63.03	74.52%
	7,000	34m 32s $\pm$ 14s	75.4	89.15%
DeiT3-H (Dense Acc. 84.97)	500	8m 24s $\pm$ 2s	68.63	80.77%
	1,000	16m 7s $\pm$ 9s	73.06	85.98%
	9,000	2h 14m 57s $\pm$ 22s	80.34	94.55%

B.7 RAW DATA

For completeness, we report the raw numerical values used in the plots in the main paper. The corresponding trends are discussed in the main text; here, the tables are intended only to make the plotted results explicit and reproducible. Tables 11 and 12 provide the raw data for the structured DeiT pruning experiments shown in Figure 1, Table 10 provides the raw data for the MobileNetV1 sparsity experiment shown in Figure 5, and Tables 13 and 14 provide the raw data for the calibration data ablations in Figure 6.

Table 10: Raw MobileNetV1 top-1 accuracy values across sparsity levels for the experiment shown in Figure 5. Δ denotes the change in top-1 accuracy relative to the corresponding dense baseline.

Method	Dense Acc.	0.5-Sparsity		0.6-Sparsity		0.7-Sparsity		0.8-Sparsity	
		Acc.	Δ	Acc.	Δ	Acc.	Δ	Acc.	Δ
MP	71.95	63.83	-8.12	47.15	-24.80	11.68	-60.27	0.43	-71.52
WF	71.95	68.91	-3.04	60.90	-11.05	29.36	-42.59	0.24	-71.71
CBS	71.95	70.21	-1.74	66.37	-5.58	55.11	-16.84	16.38	-55.57
CHITA	71.95	70.42	-1.53	67.30	-4.65	59.40	-12.55	29.78	-42.17
FALCON	71.95	70.35	-1.60	67.18	-4.77	58.40	-13.55	25.82	-46.13
SNOWS	70.89	70.10	-0.79	68.70	-2.19	65.28	-5.61	54.83	-16.06
STARFISH	71.95	**71.12**	-0.83	**70.10**	-1.85	**67.67**	-4.28	**60.66**	-11.29

Table 3: Comparison of top-1 accuracy results of different recovery methods. Retention is recovered accuracy relative to dense accuracy; Improvement is the retention gap between methods, showing that STARFISH achieves large improvement gaps over competing methods across settings. Best result within each model and scope is bolded. **Panel A:** STARFISH vs. CORP under structured DeiT pruning. **Panel B:** STARFISH vs. SNOWS under 2:4 ViT pruning.

Model	Pruning Scope	Method	Calib. Size	Recovered Acc.	Retention (%)	Improvement (Δ%)
Panel A: DeiT / CORP setting (0.5 **structured sparsity)**						
DeiT-T (Dense Acc. 72.01)	MLP (Pruned Acc. 10.93)	CORP	–	53.20	73.88%	**+10.51%**
		STARFISH	1,000	**60.77**	**84.39%**	
	QK (Pruned Acc. 62.31)	CORP	–	64.20	89.15%	**+9.43%**
		STARFISH	1,000	**70.99**	**98.58%**	
	MLP+QK (Pruned Acc. 7.57)	CORP	–	36.30	50.41%	**+31.54%**
		STARFISH	1,000	**59.01**	**81.95%**	
DeiT-S (Dense Acc. 79.72)	MLP (Pruned Acc. 45.51)	CORP	–	65.85	82.60%	**+8.39%**
		STARFISH	2,500	**72.54**	**90.99%**	
	QK (Pruned Acc. 54.14)	CORP	–	72.50	90.94%	**+8.33%**
		STARFISH	2,500	**79.14**	**99.27%**	
	MLP+QK (Pruned Acc. 17.32)	CORP	–	55.50	69.62%	**+20.18%**
		STARFISH	2,500	**71.59**	**89.80%**	
DeiT-B (Dense Acc. 81.73)	MLP (Pruned Acc. 54.56)	CORP	–	68.67	84.02%	**+8.86%**
		STARFISH	5,000	**75.91**	**92.88%**	
	QK (Pruned Acc. 73.40)	CORP	–	80.80	98.86%	**+0.87%**
		STARFISH	5,000	**81.51**	**99.73%**	
	MLP+QK (Pruned Acc. 31.19)	CORP	–	67.00	81.98%	**+10.73%**
		STARFISH	5,000	**75.77**	**92.71%**	
DeiT3-L (Dense Acc. 84.58)	MLP (Pruned Acc. 77.81)	CORP	–	80.90	95.65%	**+2.74%**
		STARFISH	7,000	**83.22**	**98.39%**	
	QK (Pruned Acc. 82.97)	CORP	–	83.60	98.84%	**+0.99%**
		STARFISH	7,000	**84.44**	**99.83%**	
	MLP+QK (Pruned Acc. 72.09)	CORP	–	79.40	93.88%	**+3.89%**
		STARFISH	7,000	**82.69**	**97.77%**	
DeiT3-H (Dense Acc. 84.97)	MLP (Pruned Acc. 81.79)	CORP	–	83.70	98.51%	**+0.82%**
		STARFISH	9,000	**84.40**	**99.33%**	
	QK (Pruned Acc. 83.23)	CORP	–	84.20	99.09%	**+0.75%**
		STARFISH	9,000	**84.83**	**99.84%**	
	MLP+QK (Pruned Acc. 78.27)	CORP	–	82.80	97.45%	**+1.48%**
		STARFISH	9,000	**84.06**	**98.93%**	
Panel B: ViT / SNOWS setting (2:4 **semi-structured sparsity)**						
ViT/B-16 (Dense Acc. 80.40)	QKV (Pruned Acc. 78.92)	SNOWS	5,000	79.45	98.82%	**+0.94%**
		STARFISH	5,000	**80.21**	**99.76%**	
	QKV+Out+MLP (Pruned Acc. 70.86)	SNOWS	5,000	76.57	95.24%	**+3.11%**
		STARFISH	5,000	**79.07**	**98.35%**	
ViT/L-16 (Dense Acc. 84.20)	QKV (Pruned Acc. 75.40)	SNOWS	20,000	81.01	96.21%	**+2.32%**
		STARFISH	7,000	**82.96**	**98.53%**	
	MLP (Pruned Acc. 0.20)	SNOWS	20,000	69.71	82.79%	**+3.27%**
		STARFISH	7,000	**72.46**	**86.06%**	

Table 11: Raw top-1 accuracy values for the structured MLP-pruning DeiT experiments shown in the right two panels of Figure 1. We report the pruned accuracy, recovered accuracy, accuracy retention relative to the dense model, and the retention improvement of STARFISH over CORP, in the settings in which CORP is reported. Best reported recovered result within each model and sparsity level is bolded.

Model	Sparsity Level	Pruned Acc.	Method	Calib. Size	Recovered Acc.	Retention (%)	Improvement (Δ%)
DeiT-S (Dense Acc. 79.72)	0.30	54.11	CORP	–	73.88	92.67%	**+2.73%**
			STARFISH	2,500	**76.06**	**95.41%**	
	0.50	45.51	CORP	–	65.85	82.60%	**+8.39%**
			STARFISH	2,500	**72.54**	**90.99%**	
	0.70	10.80	CORP	–	49.98	62.69%	**+19.04%**
			STARFISH	2,500	**65.16**	**81.74%**	
	0.75	3.84	CORP	–	–	–	–
			STARFISH	2,500	**61.68**	**77.37%**	
	0.80	1.95	CORP	–	–	–	–
			STARFISH	2,500	**57.40**	**72.00%**	
DeiT-B (Dense Acc. 81.73)	0.30	71.88	CORP	–	75.65	92.56%	**+4.07%**
			STARFISH	5,000	**78.98**	**96.64%**	
	0.50	54.56	CORP	–	68.67	84.02%	**+8.86%**
			STARFISH	5,000	**75.91**	**92.88%**	
	0.70	14.50	CORP	–	56.46	69.08%	**+16.25%**
			STARFISH	5,000	**69.74**	**85.33%**	
	0.75	7.61	CORP	–	–	–	–
			STARFISH	5,000	**67.15**	**82.16%**	
	0.80	4.06	CORP	–	–	–	–
			STARFISH	5,000	**64.28**	**78.65%**	

Table 12: Raw top-1 accuracy values for the structured QK+MLP-pruning DeiT experiments shown in the left two panels of Figure 1. We report the pruned accuracy, recovered accuracy, accuracy retention relative to the dense model, and the retention improvement of STARFISH over CORP, in the settings in which CORP is reported. Best reported recovered result within each model and sparsity level is bolded.

Model	Sparsity Level	Pruned Acc.	Method	Calib. Size	Recovered Acc.	Retention (%)	Improvement (Δ%)
DeiT-B (Dense Acc. 81.73)	0.50	31.19	CORP	–	67.00	81.98%	**+10.73%**
			STARFISH	5,000	**75.77**	**92.71%**	
	0.63	8.30	CORP	–	55.90	68.40%	**+20.05%**
			STARFISH	5,000	**72.29**	**88.45%**	
	0.69	3.54	CORP	–	46.60	57.02%	**+28.78%**
			STARFISH	5,000	**70.12**	**85.79%**	
	0.75	1.42	CORP	–	32.70	40.01%	**+42.00%**
			STARFISH	5,000	**67.03**	**82.01%**	
	0.80	0.95	CORP	–	–	–	–
			STARFISH	5,000	**64.03**	**78.34%**	
	0.85	0.62	CORP	–	–	–	–
			STARFISH	5,000	**59.70**	**73.05%**	
DeiT3-H (Dense Acc. 84.97)	0.50	78.27	CORP	–	82.80	97.45%	**+1.48%**
			STARFISH	9,000	**84.06**	**98.93%**	
	0.63	64.38	CORP	–	79.10	93.09%	**+5.00%**
			STARFISH	9,000	**83.35**	**98.09%**	
	0.69	46.72	CORP	–	75.10	88.38%	**+9.05%**
			STARFISH	9,000	**82.79**	**97.43%**	
	0.75	25.90	CORP	–	67.20	79.09%	**+17.16%**
			STARFISH	9,000	**81.78**	**96.25%**	
	0.80	13.19	CORP	–	–	–	–
			STARFISH	9,000	**80.34**	**94.55%**	
	0.85	5.88	CORP	–	–	–	–
			STARFISH	9,000	**78.24**	**92.08%**	

Table 13: Raw top-1 accuracy values for the calibration set size experiment shown in Figure 6a. Results are reported for STARFISH recovery using different calibration-set sizes on a DeiT-B model after magnitude-based unstructured pruning of QKV+Out+MLP.

	Dense Acc.: 81.73%			
Calibration set size/Sparsity	**0.5**	**0.6**	**0.7**	**0.8**
Pruned Acc.	74.59	56.16	20.02	0.83
50	79.42	75.68	62.79	5.20
100	79.62	76.44	66.40	13.79
500	80.65	78.99	74.14	52.82
1,000	80.83	79.58	76.22	62.95
2,500	81.14	80.15	77.99	70.08
5,000	81.34	80.59	78.86	73.28
9,000	81.43	80.75	79.37	74.97

Table 14: Raw top-1 accuracy values for the calibration distribution experiment shown in Figure 6b. Results are reported for STARFISH recovery using different calibration-set sizes on a DeiT-B model after magnitude-based unstructured pruning of QKV+Out+MLP.

	Dense Acc.: 81.73%				
Sparsity	**ImageNet held-out**	**ImageNetV2**	**Open Images**	**Places365**	**Gaussian noise**
0.5	81.34	81.26	81.09	80.82	77.06
0.6	80.59	80.43	80.16	79.72	69.78
0.7	78.86	78.67	77.92	76.88	40.86
0.8	73.28	72.40	70.24	65.93	3.16