TARGETED LOW-RANK REFINEMENT: ENHANCING SPARSE NEURAL NETWORKS WITH PRECISION

Anonymous authors
Paper under double-blind review

ABSTRACT

Pruning is a widely used technique for compressing large neural networks that eliminates weights that have minimal impact on the model's performance. Current pruning methods, exemplified by magnitude pruning, assign an importance score to each weight based on its magnitude and remove weights with scores below a certain threshold. Nonetheless, these methods often create a gap between the original dense and the pruned sparse model, potentially impairing performance. Especially when the sparsity ratio is high, the gap becomes more pronounced. To mitigate this issue, we introduce a method to bridge the gap left by pruning by utilizing a low-rank approximation of the difference between the dense and sparse matrices. Our method entails the iterative refinement of the sparse weight matrix augmented by a low-rank adjustment. This technique captures and retains the essential information often lost during pruning, thereby improving the performance of the pruned model. Furthermore, we offer a comprehensive theoretical analysis of our approach, emphasizing its convergence properties and establishing a solid basis for its efficacy. Experimental results on LLaMa models validate its effectiveness on large language models across various pruning techniques and sparsity levels. Our method shows significant improvements: at 50% sparsity, it reduces perplexity by 53.9% compared to conventional magnitude pruning on LLaMa-7B. Furthermore, to achieve a specific performance target, our approach enables an 8.6% reduction in model parameters while maintaining a sparsity ratio of about 50%.

1 INTRODUCTION

Pruning is a crucial technique in the field of model compression, particularly for large language models (LLMs), which have become the cornerstone of natural language processing tasks (Devlin, 2018; Brown, 2020; Hoffmann et al., 2022). Pruning involves the removal of specific weights or parameters from the neural network that are considered to have minimal impact on the overall performance of the model (LeCun et al., 1989; Hassibi et al., 1993; Han et al., 2015b; Frankle & Carbin, 2018; Frankle et al., 2020). This technique is especially valuable for LLMs, which often contain billions of parameters and require substantial computational resources for both training and inference (Han, 2017; Touvron et al., 2023; Minaee et al., 2024).

One of the most representative pruning techniques is magnitude pruning, which removes weights that have the smallest absolute values. This method is based on the assumption that smaller weights have less effect on the network's overall performance (Han et al., 2015a). Furthermore, as model sizes continue to grow, the number of redundant parameters also increases. For LLMs with billions of parameters, even half of the layers can be dropped without significantly affecting performance (Men et al., 2024; Fan et al., 2024). However, recent research shows that pruning can cause irreparable loss of knowledge and performance drops, especially for difficult tasks, a phenomenon known as the *Junk DNA Hypothesis* (Yin et al., 2024). This consistent degradation in performance is observed across a spectrum of pruning methods, including magnitude pruning, SparseGPT (Frantar & Alistarh, 2023), and Wanda (Sun et al., 2023), and applies to both unstructured pruning and structured N:M pruning.

Conventional approaches to post-pruning recovery face a three-fold challenge: (1) **Computational burden**: Re-training (Frankle & Carbin, 2018; Xia et al., 2023; Kim et al., 2024a) and knowledge distillation (Hinton, 2015; Wan et al., 2024a; Muralidharan et al., 2024) methods are computationally expensive and time-consuming. (2) **Data and model dependency**: These techniques typically require

either extensive datasets or access to a high-performing teacher model, which may not always be feasible. (3) **Sparsity inconsistency**: Recent low-rank approximation methods (Li et al., 2023; Mozaffari et al., 2024; Zhang & Papyan, 2024) often fail to maintain a consistent sparsity pattern, making them unsuitable for structured pruning, which is crucial for hardware efficiency.

In this study, we address the challenge of post-pruning recovery by approximating the dense matrix as the sum of an updated sparse matrix with a maintained sparsity pattern and a low-rank matrix. We propose an iterative refinement process that concurrently updates the sparse matrix and the low-rank component in a data-free manner. This approach effectively recovers crucial information typically lost during pruning. Our method features an adaptive low-rank approximation that dynamically adjusts to complement the sparse matrix, enabling efficient information recovery. Unlike traditional techniques that rely on large datasets or high-performing teacher models, our approach operates directly on model weights, offering a computationally efficient and broadly applicable solution. This approach aims to improve the performance of the pruned model without significantly increasing the parameter count. Our method combines the advantages of both sparse and low-rank approximation, ensuring the model maintains its efficiency while enhancing its accuracy. In addition to our empirical findings, we provide a comprehensive theoretical analysis of the iterative refinement process, which rigorously demonstrates the favorable convergence properties of our method.

Experimental results on the LLaMa models demonstrate the effectiveness of our approach. When applying 50% sparsity, our method achieves a 53.9% reduction in perplexity compared to conventional magnitude pruning. The benefits of our approach become increasingly evident as sparsity increases: at 60% sparsity, we observe a 92.0% decrease in perplexity, while at 70% sparsity, an impressive 99.6% reduction is achieved. These findings highlight the effectiveness of our method, especially in scenarios of high sparsity where traditional approaches typically face considerable challenges.

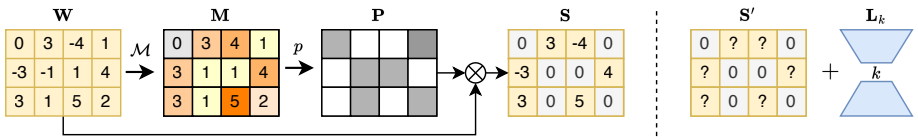
To summarize, the main contributions of this paper are:

- In this work, we bridge the gap between the original dense and pruned sparse model by leveraging a low-rank component. This approach effectively fills the gap left by pruning, enhancing the model's performance with minimal parameter increase.
- We develop an iterative algorithm that incrementally refines the sparse weight matrix and incorporates the low-rank approximation. By prioritizing the preservation of weight components associated with larger singular values, our method allows for a more aggressive reduction of less important components, leading to a more precise approximation.
- We provide a thorough theoretical analysis of our proposed method, which offers a rigorous foundation understanding of the effectiveness, convergence, and stability of our approach.
- We validate the effectiveness of our method on LLMs across various sparsity levels, achieving substantial perplexity reductions compared to baselines. This is particularly noteworthy at high sparsity levels, maintaining robust improvements even as high as 70% sparsity.

2 PRELIMINARY

2.1 THE GENERAL FRAMEWORK OF LAYER-WISE PRUNING

Neural network pruning is a crucial technique for model compression, aiming to reduce parameters while preserving performance. In this section, we first present the general framework for (local) layer-wise pruning, outlining the key steps and components involved and it is illustrated in Fig. 1(a). Let $\boldsymbol{W} \in \mathbb{R}^{m\times n}$ represent a weight matrix, and $\boldsymbol{M} = \mathcal{M}(\boldsymbol{W}, \mathcal{D}) \in \mathbb{R}^{m\times n}$ denote its corresponding importance score matrix. Here, $\mathcal{M} : \mathbb{R}^{m\times n} \times \mathcal{D} \to \mathbb{R}^{m\times n}$ is a metric function that computes the importance scores based on the weight matrix W and an optional dataset $\mathcal{D}$. The most straightforward approach is to use the magnitude of the weights as the importance score, i.e., $\boldsymbol{M}_{ij} = |\boldsymbol{W}_{ij}|$. More sophisticated methods can be employed to capture the importance of each weight more accurately, such as the SparseGPT (Frantar & Alistarh, 2023) and Wanda (Sun et al., 2023). Given a pruning ratio $p \in [0, 1]$, we set threshold h as the p-th percentile of $\boldsymbol{M}$ as the decision boundary for pruning. Then the binary pruning mask $\boldsymbol{P}$ is obtained by $\boldsymbol{P}_{ij} = \mathbb{I}(\boldsymbol{M}_{ij} > h)$. The pruned sparse weight matrix $\boldsymbol{S}$ is obtained by $\boldsymbol{S} = \boldsymbol{W} \odot \boldsymbol{P}$, where $\odot$ denotes element-wise multiplication. This zeroes out less important weights while retaining significant ones. An optional weight update procedure can be applied to refine the pruned weights further. For example, after the pruning process, a re-training



(a) The general framework of neural network pruning. (b) Refine with a low-rank patch.

Figure 1: (a) An overview of the general framework of (local) layer-wise pruning, where $\boldsymbol{W}$, $\boldsymbol{M}$, $\boldsymbol{P}$, and $\boldsymbol{S}$ are the original dense weight matrix, the importance score matrix, the binary pruning mask, and the pruned sparse weight matrix, respectively. Here we show the case of pruning using the magnitude of the weights as the importance score. (b) Our proposed method, where $\boldsymbol{S}$ is updated and a low-rank matrix $\boldsymbol{L}_k$ is introduced to minimize the approximation error.

phase may be implemented to fine-tune the remaining weights and recover some lost performance. During this process, the pruning structure is maintained by enforcing the constraint $\boldsymbol{S}^{(t)} = \boldsymbol{S}^{(t)} \odot \boldsymbol{P}$ at each iteration t. This ensures that the pruned weights remain zero throughout the re-training process, preserving the sparsity achieved through pruning.

2.2 Singular Value Decomposition

Here we briefly review the Singular Value Decomposition (SVD) of a matrix. Given a matrix $\boldsymbol{W} \in \mathbb{R}^{m \times n}$, its reduced SVD is given by (Olver & Shakiban, 2018):

$$\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top, \tag{1}$$

where $\boldsymbol{U} \in \mathbb{R}^{m \times r}$ and $\boldsymbol{V} \in \mathbb{R}^{r \times n}$ are orthogonal matrices, and $\boldsymbol{\Sigma} \in \mathbb{R}^{r \times r}$ is a diagonal matrix containing the singular values of $\boldsymbol{W}$. The i-th columns of matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are represented by $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$, respectively. Additionally, we use σ_i to denote the i-th diagonal element of the matrix $\boldsymbol{\Sigma}$. For convenience, we extend this notation to represent functions that map $\boldsymbol{W}$ to its SVD components. This allows us to express the decomposition as $\boldsymbol{W} = \boldsymbol{U}(\boldsymbol{W})\boldsymbol{\Sigma}(\boldsymbol{W})\boldsymbol{V}^\top(\boldsymbol{W})$, or alternatively as a sum of outer products: $\boldsymbol{W} = \sum_{i=1}^{r} \sigma_i(\boldsymbol{W})\boldsymbol{u}_i(\boldsymbol{W})\boldsymbol{v}_i^\top(\boldsymbol{W})$. While this slightly abuses the original notation, it provides a concise way to refer to the SVD components of any given matrix.

3 Method

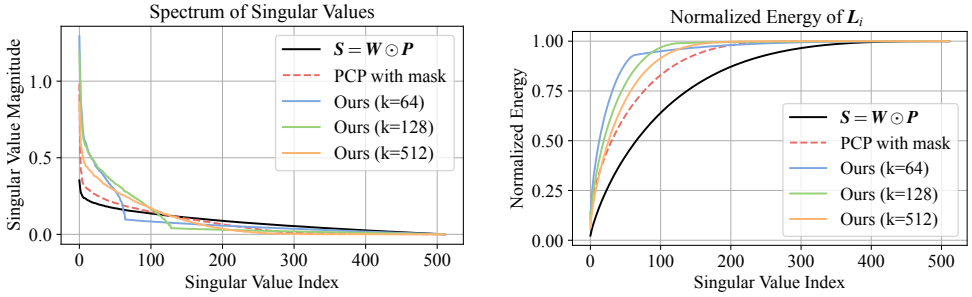
In this study, we focus on the local pruning method and aim to fill the gap between the original dense weight matrix $\boldsymbol{W}$ and the pruned weight $\boldsymbol{S}$ by introducing a low-rank matrix $\boldsymbol{L}_k$ to approximate the difference matrix $\Delta\boldsymbol{W} = \boldsymbol{W} - \boldsymbol{S}$. We illustrate the main idea in Fig. 1(b), where the dense matrix $\boldsymbol{W}$ is factorized into the superposition of a low-rank matrix $\boldsymbol{L}$ and a updated sparse matrix $\boldsymbol{S}'$:

$$\boldsymbol{W} = \boldsymbol{S}' + \boldsymbol{L} \approx \boldsymbol{S}' + \boldsymbol{L}_k. \tag{2}$$

In this formulation, $\boldsymbol{S}'$ maintains the same sparsity pattern as $\boldsymbol{S}$, ensuring that $\boldsymbol{S}' = \boldsymbol{S}' \odot \boldsymbol{P}$ and $\boldsymbol{S} = \boldsymbol{S} \odot \boldsymbol{P}$ are satisfied. It's important to note that both $\boldsymbol{S}'$ and $\boldsymbol{L}$ can have elements of any magnitude without restrictions. $\boldsymbol{L}_k$ is the best rank-k approximation of $\boldsymbol{W}$, which can be obtained using the SVD of $\boldsymbol{L}$ as $\boldsymbol{L}_k = \boldsymbol{U}_{:k}(\boldsymbol{L})\boldsymbol{\Sigma}_{:k}(\boldsymbol{L})\boldsymbol{V}_{:k}^\top(\boldsymbol{L})$. Begin with fixed $\boldsymbol{S}$, we have the baseline:

Baseline 1 (Zero-shot SVD) *The most straightforward method to obtain the low-rank matrix $\boldsymbol{L}_k$ is to directly perform the SVD on $\boldsymbol{W} - \boldsymbol{S}$ without updating $\boldsymbol{S}$. This can be expressed as $\boldsymbol{L}_k = \boldsymbol{U}_{:k}(\boldsymbol{W} - \boldsymbol{S})\boldsymbol{\Sigma}_{:k}(\boldsymbol{W} - \boldsymbol{S})\boldsymbol{V}_{:k}^\top(\boldsymbol{W} - \boldsymbol{S})$. Here, $\boldsymbol{U}_{:k}(\cdot)$, $\boldsymbol{\Sigma}_{:k}(\cdot)$, and $\boldsymbol{V}_{:k}(\cdot)$ represent the first k columns of $\boldsymbol{U}(\cdot)$, the top-left $k \times k$ submatrix of $\boldsymbol{\Sigma}(\cdot)$, and the first k columns of $\boldsymbol{V}(\cdot)$, respectively.*

Parameter efficiency analysis. We analyze the computational efficiency of low-rank refinement by examining its parameter count and FLOPs (floating-point operations). We compare these metrics with the dense model and a pruned model, considering the impact of sparsity. For a weight matrix $\boldsymbol{W} \in \mathbb{R}^{m \times n}$, the dense model has mn parameters and requires $2mn$ FLOPs for a forward pass. With pruning at sparsity ratio p, these reduce to $(1-p)mn$ parameters and $2(1-p)mn$ FLOPs.



(a) Singular value distribution of $\boldsymbol{L}$

(b) Cumulative energy retention in $\boldsymbol{L}_k$

Figure 2: Analysis of the residual matrix $\boldsymbol{L} = \boldsymbol{W} - \boldsymbol{S}'$ and its low-rank approximation $\boldsymbol{L}_k$ using different methods. Results are shown for zero-shot SVD, PCP baseline ($T = 5000$), and our proposed method with varying k $(64, 128, 512)$ and $T = 50$. (a) Singular value spectrum of $\boldsymbol{L}$. (b) Proportion of total energy captured by the top k singular values, calculated as $\sum_{j=1}^{i} \sigma_j^2(\boldsymbol{L}_k) / \sum_{j=1}^{r} \sigma_j^2(\boldsymbol{L})$.

Our method introduces a low-rank matrix $\boldsymbol{L}_k = \boldsymbol{B}\boldsymbol{A}$, which introduces $k(m+n)$ parameters and $2k(m+n)$ FLOPs, slightly reducing the overall sparsity ratio by $k(\frac{1}{m} + \frac{1}{n})$. The choice of k and p presents a trade-off between model size, computational complexity, and performance. In practice, using a rank k of 128 results in only a 4.9% increase in parameters, while possibly reducing perplexity by half on a LLaMa-7B model with both unstructured 50% sparsity and structured 4:8 sparsity.

This optimization problem is generally NP-hard and similar challenges have been addressed in the fields of matrix completion (Chandrasekaran et al., 2011) and robust Principal Component Analysis (PCA) (Candès et al., 2011; Peng et al., 2020) by solving a Principal Component Pursuit (PCP) as:

$$\min_{\boldsymbol{L}, \boldsymbol{S}'} \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{S}'\|_1, \quad \text{s.t. } \boldsymbol{L} + \boldsymbol{S}' = \boldsymbol{W}. \tag{3}$$

Where $\|\cdot\|_*$ denotes the nuclear norm and $\|\cdot\|_1$ denotes the ℓ_1-norm, λ is a hyperparameter that controls the trade-off between the rank of $\boldsymbol{L}$ and the sparsity of $\boldsymbol{S}'$. The nuclear norm serves as a convex approximation for the rank of a matrix, while the ℓ_1-norm acts as a convex proxy for the ℓ_0-norm (which represents the count of non-zero elements in a matrix). In this convex optimization problem, the $\|\boldsymbol{L}\|_*$ term promotes a low-rank solution for $\boldsymbol{L}$, whereas the $\lambda\|\boldsymbol{S}'\|_1$ term promotes sparsity in $\boldsymbol{S}'$. Nevertheless, it's important to note that the solution to Eq.(3) does not necessarily preserve the fixed sparsity pattern of $\boldsymbol{S}'$ that is required to match $\boldsymbol{S}$ in Eq.(2). To address this issue, we propose to incorporate the binary mask $\boldsymbol{P}$ into the optimization process to ensure that the sparsity pattern of $\boldsymbol{S}'$ is fixed as $\boldsymbol{S}$. This is achieved by rewriting Eq.(2) as follows:

$$\boldsymbol{W} = \underbrace{(\overbrace{\boldsymbol{W} \odot \boldsymbol{P}}^{\boldsymbol{S}} - \boldsymbol{Q} \odot \boldsymbol{P})}_{\text{sparse part } \boldsymbol{S}'} + \underbrace{(\boldsymbol{W} \odot (1 - \boldsymbol{P}) + \boldsymbol{Q} \odot \boldsymbol{P})}_{\text{low-rank part } \boldsymbol{L}}. \tag{4}$$

Where $\boldsymbol{Q}$ is a learnable matrix in the same shape as $\boldsymbol{W}$.

Baseline 2 (PCP with mask) *By directly substituting the decomposition from Eq.(4) into the optimization problem presented in Eq.(3) yields:*

$$\min_{\boldsymbol{Q}} \|\boldsymbol{W} \odot (1 - \boldsymbol{P}) + \boldsymbol{Q} \odot \boldsymbol{P}\|_* + \lambda \|\boldsymbol{W} \odot \boldsymbol{P} - \boldsymbol{Q} \odot \boldsymbol{P}\|_1, \textit{and } \boldsymbol{L} = \boldsymbol{W} \odot (1 - \boldsymbol{P}) + \boldsymbol{Q} \odot \boldsymbol{P}. \tag{5}$$

This equation represents a constrained optimization problem where we seek to find the optimal matrix $\boldsymbol{Q}$. The binary mask $\boldsymbol{P}$ plays a crucial role in maintaining the desired sparsity pattern. To solve this optimization problem, we can employ iterative methods such as gradient descent or its variants. Following Candès et al. (2011), we set $\lambda = 1/\sqrt{\max(m, n)}$. However, the nuclear norm minimization approach has a limitation: it applies equal shrinkage to all rank components (Zha et al., 2019). This uniform treatment may not be optimal for our objective in Eq.(2), where we aim to approximate the residual matrix $\boldsymbol{W} - \boldsymbol{S}$ using a low-rank matrix $\boldsymbol{L}_k$ with a rank lower than that of $\boldsymbol{L}$.

To address this limitation, we propose a forward-only method that prioritizes the preservation of low-rank components associated with larger singular values, while allowing for a more aggressive

reduction of components with smaller singular values. This approach offers greater flexibility and precision in low-rank approximation compared to the PCP baseline, aligning more closely with our goal of achieving an efficient low-rank approximation. We begin by setting $\boldsymbol{S}^{(0)} = \boldsymbol{W} \odot \boldsymbol{P}$, and then iteratively refine $\boldsymbol{S}^{(t+1)}$ using the following update rule:

$$\boldsymbol{S}^{(t+1)} = \boldsymbol{S}^{(t)} + \boldsymbol{P} \odot \left\{ \sum_{i=r^{(t)}+1}^{r} \sigma_i \left(\boldsymbol{L}^{(t)} \right) \boldsymbol{u}_i \left(\boldsymbol{L}^{(t)} \right) \boldsymbol{v}_i^{\top} \left(\boldsymbol{L}^{(t)} \right) \right\} \tag{6}$$

$$= \boldsymbol{S}^{(t)} + \boldsymbol{P} \odot \left\{ \boldsymbol{U}_{r^{(t)}:} \left(\boldsymbol{L}^{(t)} \right) \boldsymbol{\Sigma}_{r^{(t)}:} \left(\boldsymbol{L}^{(t)} \right) \boldsymbol{V}_{r^{(t)}:}^{\top} \left(\boldsymbol{L}^{(t)} \right) \right\}. \tag{7}$$

In this equation, $\boldsymbol{L}^{(t)}$ represents the difference $\boldsymbol{W} - \boldsymbol{S}^{(t)}$, and $r^{(t)}$ is defined as $\lfloor 1 + \frac{k-1}{T-1} t \rfloor$, where T denotes the total number of iterations, k is the target rank of $\boldsymbol{L}_k$, and t ranges from 0 to $T-1$. We summarize the above algorithm in Algorithm. 1. In each iteration, our algorithm focuses on eliminating the least significant components from the residual matrix $\boldsymbol{L}^{(t)}$, which are associated with smaller singular values, while preserving those with larger singular values. This approach allows for a progressive refinement of the sparse matrix $\boldsymbol{S}$. As we advance through the iterations, we gradually incorporate more subtle details from the original dense weight matrix $\boldsymbol{W}$ into the updated sparse term $\boldsymbol{S}$. Incrementally increasing the rank $r^{(t)}$ from 2 to k over T iterations enables a more nuanced exploration of the weight space. Furthermore, the consistent application of the binary mask $\boldsymbol{P}$ throughout this optimization process ensures the preservation of the desired sparsity pattern.

Algorithm 1 Iterative Weight Update

1: **Inputs:** Dense weight matrix $\boldsymbol{W}$, binary mask $\boldsymbol{P}$, target rank k, number of iterations T
2: Initialize $\boldsymbol{S}^{(0)} \leftarrow \boldsymbol{W} \odot \boldsymbol{P}$
3: **for** $t = 0$ to $T-1$ **do**
4: $\quad \boldsymbol{L}^{(t)} \leftarrow \boldsymbol{W} - \boldsymbol{S}^{(t)}$
5: $\quad$ Compute SVD: $\boldsymbol{L}^{(t)} = \boldsymbol{U}^{(t)} \boldsymbol{\Sigma}^{(t)} \boldsymbol{V}^{(t)^{\top}}$
6: $\quad r^{(t)} \leftarrow \lfloor 1 + \frac{k-1}{T-1} t \rfloor$
7: $\quad \boldsymbol{S}^{(t+1)} \leftarrow \boldsymbol{S}^{(t)} + \boldsymbol{P} \odot \left\{ \boldsymbol{U}^{(t)}_{r^{(t)}:} \boldsymbol{\Sigma}^{(t)}_{r^{(t)}:} \boldsymbol{V}^{(t)^{\top}}_{r^{(t)}:} \right\}$
8: **end for**
9: $\boldsymbol{L}^{(T)} \leftarrow \boldsymbol{W} - \boldsymbol{S}^{(T)}$
10: **Returns:** $\boldsymbol{S}^{(T)}$, $\boldsymbol{L}^{(T)}$

In Figure 2, we show the singular value spectrum of $\boldsymbol{L}$ and the cumulative energy retention in $\boldsymbol{L}_k$ for different methods using a 512×512 weight block extracted from a fine-tuned LLaMa-7B model. Subfigure 2a demonstrates that the proposed method with different target rank values k (64, 128, and 512), consistently produces a more pronounced decay in singular values compared to both the zero-shot SVD and the PCP baseline. Subfigure 2b shows that the cumulative energy retention in $\boldsymbol{L}$, calculated as $E(i) = \sum_{j=1}^{i} \sigma_j^2(\boldsymbol{L}) / \sum_{j=1}^{i} \sigma_j^2(\boldsymbol{L})$ or $\|\boldsymbol{L}_i\|_F^2 / \|\boldsymbol{L}\|_F^2$, increases more rapidly with truncated rank i for the proposed method compared to the PCP baseline 2.

From Algorithm 1 and observations from Figure 2, we can summarize the technical contributions of our proposed method as follows: (1) **Iterative refinement with adaptive rank increase.** Our method iteratively refines the sparse matrix $\boldsymbol{S}$ while gradually increasing the rank $r^{(t)}$ from 2 to k over T iterations. This enables progressive elimination of less significant components from the residual matrix $\boldsymbol{L}^{(t)}$. (2) **Sparsity-preserving.** Throughout the optimization process, our method consistently applies the binary mask $\boldsymbol{P}$, ensuring the preservation of the desired sparsity pattern. Sparsity-preserving enables compatibility with both unstructured and structured pruning techniques, facilitating hardware-friendly implementations, particularly for structured sparsity patterns. (3) **Efficient information capture.** Our approach creates low-rank approximations that more efficiently capture the essential information of the original weight matrix compared to zero-shot SVD.

3.1 Theoretical Analysis

In this section, we delve into the theoretical underpinnings of our proposed method, offering a more rigorous analysis of its properties and performance. We present key theorems that elucidate the behavior of our algorithm, focusing on two critical aspects: sparsity preservation and convergence.

First, we demonstrate that our method maintains the desired sparsity pattern throughout the iterative process, ensuring that the final solution adheres to the specified binary mask. This property is crucial for applications where specific weight connections must remain zero.

Theorem 1 (Sparsity Preservation) *For all iterations t, the sparsity pattern of $\boldsymbol{S}^{(t)}$ is preserved and matches the binary mask $\boldsymbol{P}$, i.e., $\boldsymbol{S}^{(t)} = \boldsymbol{S}^{(t)} \odot \boldsymbol{P}$ for all t. Refer to Proof. 5.*

Second, we prove the convergence of our algorithm, showing that it approaches a well-defined solution as the number of iterations increases. This convergence guarantee provides theoretical justification for the stability and reliability of our method.

Theorem 2 (Convergence) *For any weight matrix $\boldsymbol{W} \in \mathbb{R}^{m\times n}$ and binary mask $\boldsymbol{P} \in \{0,1\}^{m\times n}$, the iterative weight update algorithm converges to a solution $(\boldsymbol{S}^*, \boldsymbol{L}^*)$ as $T \to \infty$, such that $\boldsymbol{W} = \boldsymbol{S}^* + \boldsymbol{L}^*$, $\boldsymbol{S}^* = \boldsymbol{S}^* \odot \boldsymbol{P}$. Moreover, $\lim_{T\to\infty} \|\boldsymbol{S}^{(T)} - \boldsymbol{S}^*\|_F = 0$ and $\lim_{T\to\infty} \|\boldsymbol{L}^{(T)} - \boldsymbol{L}^*\|_F = 0$.*

Theorem 3 (Asymptotic Convergence) *There exists a time step T_0 such that for all $t > T_0$, the Frobenius norm of the error decreases monotonically, i.e., for some*

$$\left\|\boldsymbol{W} - \left(\boldsymbol{S}^{(t+1)} + \boldsymbol{L}_k^{(t+1)}\right)\right\|_F \leq \left\|\boldsymbol{W} - \left(\boldsymbol{S}^{(t)} + \boldsymbol{L}_k^{(t)}\right)\right\|_F. \tag{8}$$

A corollary of Theorem. 3 is that if we choose to fix the $r^{(t)}$ to be k for all t, the Frobenius norm of the error decreases monotonically. Because in this case, the term $\|\boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}\|_F^2$ vanishes, and Eq.(43) becomes $\left\|\boldsymbol{E}^{(t+1)}\right\|_F^2 \leq \left\|\boldsymbol{E}^{(t)}\right\|_F^2 - \left\|\boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\|_F^2$. Formally, we have the following corollary:

Corollary 1 (Monotonic Improvement) *If we choose to fix the $r^{(t)}$ to be k for all t, the Frobenius norm of the error decreases monotonically, i.e.,*

$$\left\|\boldsymbol{E}^{(t+1)}\right\|_F - \left\|\boldsymbol{E}^{(t)}\right\|_F \leq -\left\|\boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\|_F \leq 0, \tag{9}$$

the equality holds if and only if $\boldsymbol{E}^{(t)} = \boldsymbol{0}$.

However, our empirical observations suggest that fixing $r^{(t)}$ to be k throughout the process is not optimal in practice, and gradually increasing $r^{(t)}$ as the iterations progress leads to faster convergence.

Theorem 4 (Error Bound) *At each iteration t, the Frobenius norm of the difference between the original weight matrix $\boldsymbol{W}$ and its approximation $\boldsymbol{S}^{(t)} + \boldsymbol{L}_k^{(t)}$ is bounded by:*

$$\left\|\boldsymbol{W} - \left(\boldsymbol{S}^{(t)} + \boldsymbol{L}_k^{(t)}\right)\right\|_F = \sqrt{\sum_{i=k+1}^{r} \sigma_i^2\left(\boldsymbol{L}^{(t)}\right)} \leq \sqrt{(r-k)}\sigma_{k+1}\left(\boldsymbol{L}^{(t)}\right), \tag{10}$$

where $\sigma_i\left(\boldsymbol{L}^{(t)}\right)$ are the singular values of $\boldsymbol{L}^{(t)} = \boldsymbol{W} - \boldsymbol{S}^{(t)}$, and $r = rank\left(\boldsymbol{L}^{(t)}\right)$.

For a more comprehensive treatment, including detailed proofs of these theorems and additional supporting lemmas, we direct the reader to Appendix A. This appendix contains the full mathematical derivations and supplementary results that underpin our theoretical analysis.

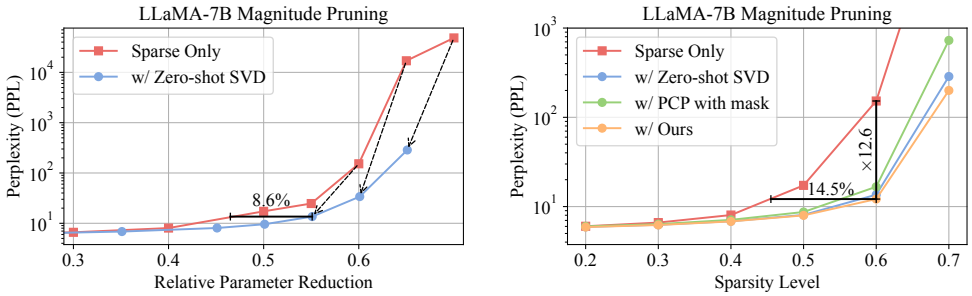
4 Experiment

4.1 Experimental Setup

We conducted our experiments using LLaMa models, evaluating their performance on the WikiText-2. Our primary metric for assessment was perplexity, where lower values indicate better performance. We tested various sparsity levels, including unstructured sparsity and structured N:M sparsity.

4.2 Effectiveness of Low-Rank Refinement

First, we examine the benefits of low-rank refinement on the LLaMa-7B model by evaluating the WikiText-2 validation perplexity. Figure 3 illustrates the perplexity evaluation for various sparsity levels using magnitude pruning and low-rank refinement methods, with the target rank k set to 128. In subfigure 3a, we compare the perplexity of sparsity-only pruning and zero-shot SVD



(a) Low-rank refinement advantage.

(b) Method comparison.

Figure 3: Perplexity evaluation for different sparsity levels and methods, $k = 128$ for low-rank refinement. (a) Low-rank refinement advantage over sparsity-only pruning. (b) Comparison of our proposed iterative weight update method with other baseline methods.

Table 1: Comparison of WikiText validation perplexity (↓ is better) across various sparsity levels on LLaMa-7B. All methods and sparsity levels use a target rank of $k = 128$ (4.9% more parameters).

	SPARSITY LEVEL (LLaMa-7B)				
METHOD	**50%**	**60%**	**70%**	**4:8**	**2:4**
Dense			5.68		
Magnitude	17.29 (0%)	152.36 (0%)	48427.85 (0%)	16.83 (0%)	42.53 (0%)
w/ Zero-shot SVD	8.06 (-53.4%)	13.59 (-91.1%)	283.74 (-99.4%)	9.29 (-44.8%)	12.72 (-70.0%)
w/ PCP with mask	8.70 (-49.7%)	16.67 (-89.1%)	727.54 (-98.5%)	10.60 (-37.0%)	16.62 (-60.9%)
w/ **Ours**	**7.97 (-53.9%)**	**12.14 (-92.0%)**	**200.09 (-99.6%)**	**8.86 (-47.4%)**	**10.74 (-74.7%)**
Wanda	7.26 (0%)	10.69 (0%)	84.69 (0%)	8.57 (0%)	11.53 (0%)
w/ Zero-shot SVD	7.09 (-2.3%)	9.60 (-10.2%)	35.65 (-57.9%)	8.14 (-5.0%)	10.48 (-9.1%)
w/ PCP with mask	7.28 (+0.3%)	10.19 (-4.7%)	47.11 (-44.4%)	8.63 (-0.7%)	11.22 (-2.7%)
w/ **Ours**	**6.92 (-4.7%)**	**8.97 (-16.1%)**	**32.90 (-61.2%)**	**7.74 (-9.7%)**	**9.18 (-20.4%)**

refinement to highlight the benefits of the low-rank refinement strategy. By incorporating a low-rank structure into the pruned weights, the parameter count increases, so the x-axis represents the parameter reduction relative to the dense model, and we use a dashed black arrow to indicate the correspondence between the pruned model and the model with refinement. The results indicate that we gain 8.6% parameter reduction at approximately 50% sparsity, highlighting the superiority of low-rank refinement compared to sparsity-only pruning techniques. In subfigure 3b, we compare our proposed method with other baseline methods. Our proposed method consistently outperforms the other two baseline methods, with the benefits becoming more pronounced at higher sparsity levels. It is evident that PCP with mask underperforms compared to zero-shot SVD and our iterative weight update method, although it still surpasses sparsity-only pruning. This can be attributed to PCP's uniform shrinkage of all singular values, including smaller ones, along with our choice of a very low target rank $k = 128$. As illustrated in Figure 4a, PCP also yields larger singular values for indices ranging from 10^2 to nearly 10^3, demonstrating its less discriminative approach to value reduction.

To further analyze the performance of low-rank refinement with different sparsity types and levels, we provide a more detailed comparison of WikiText-2 validation perplexity across various sparsity levels and pruning methods in Tables 1 and 2. The tables include results for different sparsity levels, ranging from 50% to 70%, as well as structured sparsity patterns like 4:8 and 2:4. For each sparsity level, we compare the performance of low-rank refinement incorporated with magnitude pruning and Wanda pruning (Sun et al., 2023). At 50% sparsity, our method reduces perplexity by 53.9% compared to sparse-only magnitude pruning. This improvement becomes more significant at higher sparsity levels, reaching a 92.0% reduction at 60% sparsity and a 99.6% reduction at 70% sparsity. Our proposed method consistently achieves lower perplexity values across all sparsity levels. This detailed comparison highlights the effectiveness of our method in maintaining low perplexity even at higher sparsity levels, demonstrating its robustness and superiority over other baseline methods.

Table 2: Comparison of WikiText validation perplexity (↓ is better) across various sparsity levels on LLaMa-13B. All methods and sparsity levels use a target rank of $k = 128$ (3.8% more parameters).

	SPARSITY LEVEL (LLaMa-13B)				
METHOD	**50%**	**60%**	**70%**	**4:8**	**2:4**
Dense			4.57		
Magnitude	5.98 (0%)	9.91 (0%)	408.75 (0%)	6.76 (0%)	8.32 (0%)
w/ Zero-shot SVD	5.73 (-4.2%)	**8.83 (-10.9%)**	163.96 (-59.9%)	6.58 (-2.7%)	8.86 (-6.5%)
w/ **Ours**	**5.65 (-5.5%)**	**8.83 (-10.9%)**	**99.27 (-75.7%)**	**6.40 (-5.3%)**	**7.76 (-6.7%)**

Table 3: Performance comparison on LLaMa models across several benchmark datasets ($k = 128$).

	METHOD	**TruthfulQA**	**GSM8K**	**ARC-c**	**MMLU**	**AVG.**
7B	Dense	34.1	10.3	44.7	32.1	30.3 (0%)
	Magnitude 50%	**35.3**	1.0	33.5	24.6	23.6 (-22.1%)
	w/ Zero-shot SVD	34.3	1.5	36.9	**26.0**	24.7 (-18.5%)
	w/ **Ours**	34.2	**3.4**	**41.5**	**26.0**	**26.3 (-13.2%)**
13B	Dense	36.9	23.4	49.1	52.1	40.4 (0%)
	Magnitude 2:4	**38.4**	1.7	34.9	27.8	25.7 (-36.4%)
	w/ Zero-shot SVD	37.6	1.8	32.2	27.0	24.7 (-38.9%)
	w/ **Ours**	36.9	**9.4**	**36.7**	**41.9**	**31.2 (-22.8%)**

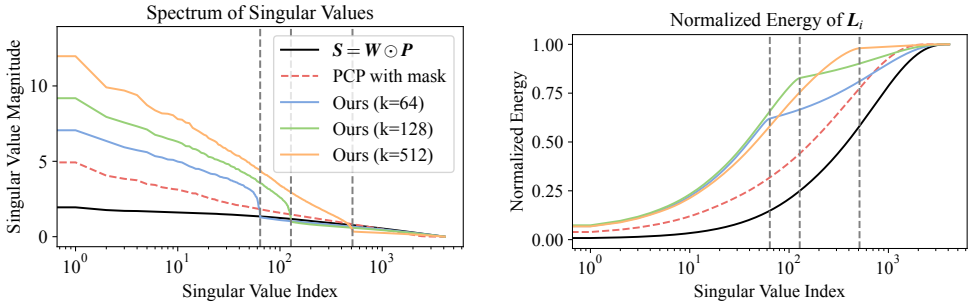
In Table 3, we evaluate the performance of our method on several benchmark datasets. We compare the performance of our method with that of dense models, sparsity-only pruned models, and low-rank refined using zero-shot SVD. Our method consistently outperforms both magnitude pruning and zero-shot SVD across most tasks and model sizes. For the 7B model, it achieves a 13.2% reduction in average performance compared to 22.1% for magnitude pruning and 18.5% for zero-shot SVD.

4.3 Iterative Weight Update Analysis

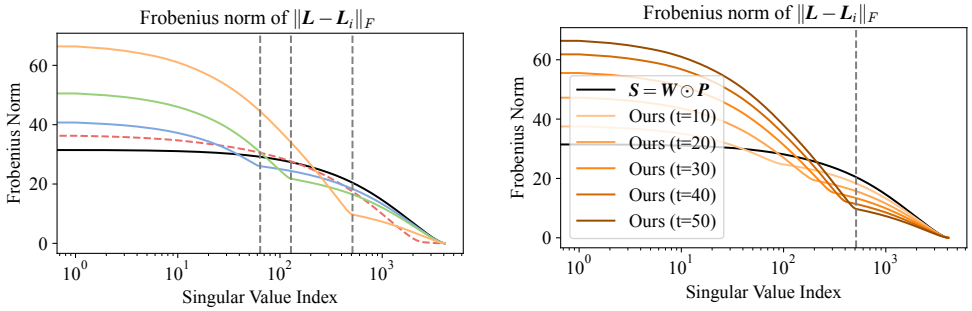
Here we analyze the behavior of our proposed iterative weight update algorithm (Algorithm 1). Our goal is to empirically validate the theoretical properties established in Section 3, specifically regarding convergence, error reduction, and emperically investigate the lower bound of rank$(\boldsymbol{L})$.

Convergence and Error Analysis. In Figure 4, we visualize several key properties of the residual matrix $\boldsymbol{L} = \boldsymbol{W} - \boldsymbol{S}'$ and its low-rank approximation $\boldsymbol{L}_k$, we use a log scale for the x-axis to better visualize the decay of singular values at low ranks. Throughout the analysis, we use magnitude pruning with a sparsity level of 50%. (a) *Singular Value Spectrum*. We first visualize the singular value spectrum of $\boldsymbol{L}$ for different methods and hyperparameter configurations for our method. It is clear that the iterative weight update method exhibits a more pronounced decay in singular values compared to both zero-shot SVD and the PCP baseline. (b) *Energy Retention*. Here we show the cumulative energy retention of $\boldsymbol{L}$, calculated as $\sum_{j=1}^{i} \sigma_j^2(\boldsymbol{L}_k) / \sum_{j=1}^{r} \sigma_j^2(\boldsymbol{L}) = \|\boldsymbol{L}_i\|_F^2 / \|\boldsymbol{L}\|_F^2$. Across various target rank values, our method captures a higher proportion of energy within the top k singular values, suggesting that our low-rank approximations preserve more information from the original matrix, thereby enhancing performance. (c) *Error Analysis*. This subfigure presents the Frobenius norm of the error $\boldsymbol{E}_i = \boldsymbol{W} - (\boldsymbol{S}^{(t)} + \boldsymbol{L}_i^{(t)})$, noting that $\boldsymbol{E}_i = \boldsymbol{L} - \boldsymbol{L}_i$ as well. It is observed that at the desired target rank, our method achieves a lower Frobenius norm error than both zero-shot SVD and the PCP baseline. (d) *Convergence Analysis*. We illustrate how the approximation error between the dense matrix $\boldsymbol{W}$ and the combined approximation $\boldsymbol{S}^{(t)} + \boldsymbol{L}_k^{(t)}$ diminishes over iterations, as predicted by Theorem 2 and Theorem 3. Specifically, it shows the Frobenius norm $\|\boldsymbol{E}_i^{(t)}\|_F = \|\boldsymbol{L}^{(t)} - \boldsymbol{L}_i^{(t)}\|_F$ at various iterations for $k = 512$ and $T = 50$.

Investigating the Lower Bound of rank$(\boldsymbol{L})$ An intriguing question arises: *Is it possible to fully compress the information contained in $\boldsymbol{W} \odot (1 - \boldsymbol{P})$ into $\boldsymbol{S}'$? In other words, does the rank of $\boldsymbol{L}$*

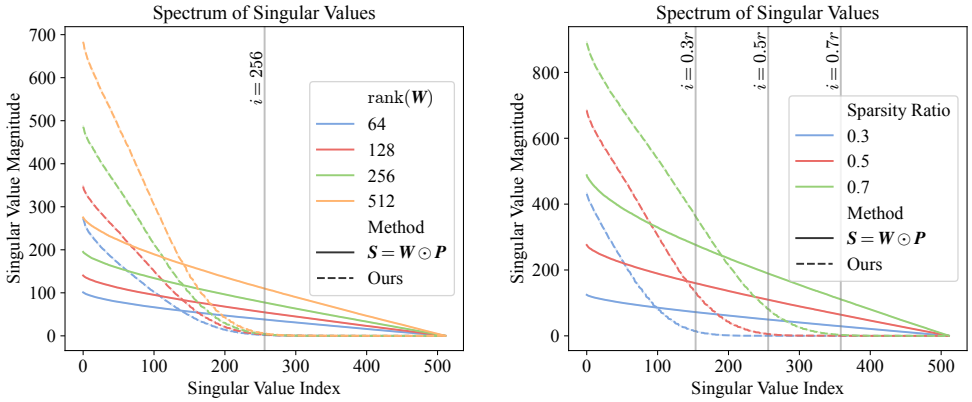


(a) Singular values distribution of $\boldsymbol{L}$. (b) Cumulative energy retention of $\boldsymbol{L}$.



(c) Frobenius norm of $\|\boldsymbol{E}_i\|_F = \|\boldsymbol{L} - \boldsymbol{L}_i\|_F$. (d) $\|\boldsymbol{E}_i^{(t)}\|_F$ at different time steps t ($k = 512$).

Figure 4: Analysis of the residual matrix $\boldsymbol{L} = \boldsymbol{W} - \boldsymbol{S}'$ and its low-rank approximation $\boldsymbol{L}_k$ using different methods. Results are shown for zero-shot SVD, PCP baseline ($T = 100$), and our proposed method with varying k $(64, 128, 512)$ and $T = 50$. We show the x-axis in log scale and vertical dashed lines at $i = 64, 128, 512$ for better visualization. Subfigures (a), (b), and (c) have a shared legend. (a) Singular value spectrum of $\boldsymbol{L}$. (b) Proportion of total energy captured by the top k singular values, calculated as $\sum_{i=1}^{k} \sigma_i^2(\boldsymbol{L}_k) / \sum_{i=1}^{r} \sigma_i^2(\boldsymbol{L})$. (c) Frobenius norm of the dropped matrix $\|\boldsymbol{L} - \boldsymbol{L}_i\|_F$. (d) $\|\boldsymbol{L} - \boldsymbol{L}_i\|_F$ at different number of iterations t. Here $T = 50$ and $k = 512$ and the vertical dashed line at $i = 512$ indicates the target rank.



(a) Spectrum of $\boldsymbol{L}$ across various ranks of $\boldsymbol{W}$. (b) Spectrum of $\boldsymbol{L}$ across various sparsity levels.

Figure 5: Singular value spectrum analysis of $\boldsymbol{L}$ across different conditions. We compare our method with the zero-shot SVD method $\boldsymbol{L} = \boldsymbol{W} \odot (1 - \boldsymbol{P})$. (a) This subfigure shows the spectrum of $\boldsymbol{L}$ across different ranks of the original weight matrix $\boldsymbol{W}$ (64, 128, 256, 512). (b) This subfigure shows the spectrum of $\boldsymbol{L}$ across different sparsity levels (0.3, 0.5, 0.7) of magnitude pruning.

have a lower bound? To address this, we conduct an empirical investigation by applying our method to various target ranks k, using both synthetic matrices and real-world weight matrices extracted from the LLaMa-7B model. We first construct a series of synthetic matrices $\boldsymbol{W}$ of known rank r by multiplying two Gaussian random matrices: $\boldsymbol{U} \in \mathbb{R}^{m \times r}$ and $\boldsymbol{V} \in \mathbb{R}^{n \times r}$, such that $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{V}^T$.

Here, $m = n = 512$. We subsequently apply our method along with zero-shot SVD to $\boldsymbol{W}$, varying the rank r and sparsity level p, and visualize the spectrum of $\boldsymbol{L} = \boldsymbol{W} - \boldsymbol{S}'$ in Figure 5.

In both subfigures in Figure 5, the solid lines represent the zero-shot SVD method ($\boldsymbol{L} = \boldsymbol{W} \odot (1 - \boldsymbol{P})$), while the dashed lines represent the proposed method. The proposed method consistently shows a steeper decay in singular values compared to the zero-shot SVD method, indicating better compression of information. In subfigure 5a, we observe that for diverse ranks of $\boldsymbol{W}$ and a constant sparsity level $p = 0.5$, the rank of $\boldsymbol{L}$ remains invariant. Conversely, subfigure 5b shows that fixing the rank of $\boldsymbol{W}$ at $r = 512$ and varying the sparsity level p from 0.3 to 0.7, reveals a positive correlation between the rank of $\boldsymbol{L}$ and increasing sparsity. More specifically, the rank of $\boldsymbol{L}$ is around $p \times \min(m, n)$. Empirically, we come to the conclusion that the rank of $\boldsymbol{L}$ is independent of rank($\boldsymbol{W}$) but dependent on the sparsity level p. This is consistent with the intuitive explanation that as the sparsity level increases, the pruned matrix contains more information, and thus the rank of $\boldsymbol{L}$ is higher. Similar observations can be made in the LLaMa-7B model, as shown in Figure 2.

5 Related Work

LLM Compression. LLMs have become essential in natural language processing tasks, but their substantial size poses challenges in terms of computational resources and efficiency. Various techniques have been proposed to compress these models while maintaining their performance, including pruning, low-rank compression, quantization, and knowledge distillation (Cheng et al., 2017; Choudhary et al., 2020; Haroush et al., 2020). In this study, we focus on the pruning technique, and the low-rank approximations are utilized to recover the lost performance. Pruning LLMs with billions of parameters differs significantly from pruning smaller models (Gale et al., 2019; Frankle et al., 2020; Kurtic & Alistarh, 2022), as current pruning techniques often necessitate extensive re-training after the pruning process, which is prohibitively costly for LLMs (Komatsuzaki et al., 2022; Chung et al., 2024; Snell et al., 2024). Pruning methods are often categorized into unstructured and structured pruning (Liu et al., 2017; Fan et al., 2019; He & Xiao, 2023). Structured pruning techniques, including layer pruning (Chen & Zhao, 2018; Kim et al., 2024b), channel pruning (He et al., 2017; Zhuang et al., 2018), and N:M pruning (Sun et al., 2021), focus on eliminating entire neurons, layers, or N out of M elements in a regular pattern. Unstructured pruning removes individual weights without considering the underlying structure, which can often lead to better performance but is less hardware-friendly. Low-rank compression reduces the dimensions of the weight matrix by focusing on larger singular values in both the column and row spaces (Cheng et al., 2005; Idelbayev & Carreira-Perpinán, 2020). While quantization (Xiao et al., 2023; Lin et al., 2024) and knowledge distillation (Wan et al., 2024b) have been used to compress LLMs, they are orthogonal to our approach.

Post-Pruning Performance Recovery. To recover the performance after pruning step, several post-pruning recovery techniques have been explored such as re-training the pruned model and using knowledge distillation (Muralidharan et al., 2024). However, these approaches can be computationally expensive and time-consuming. Recent research has focused on incorporating low-rank approximations to recover the lost performance. By adding a low-rank component to the pruned model, it is possible to approximate the original dense model more closely with minimal computational overhead (Li et al., 2023; Mozaffari et al., 2024; Zhang & Papyan, 2024).

6 Conclusion

In this work, we present an iterative weight update algorithm for low-rank refinement of sparse-pruned models. Our approach aims to bridge the performance gap between dense and pruned sparse models in large language models. The proposed method offers a computationally efficient solution that does not rely on extensive datasets or high-performing teacher models, making it a practical choice for improving sparse model performance. A notable advantage of our method is its sparsity-preserving property, which allows for the concurrent update of the sparse matrix while maintaining its sparsity pattern and incorporating a low-rank component. This approach effectively recovers crucial information lost during pruning, leading to performance recovery, especially at high sparsity ratios.

These experimental results highlight the potential of combining low-rank and sparsity in LLMs. Future work could explore methods for automatically determining the optimal rank for the low-rank component based on the specific characteristics of each layer or the overall model architecture.

REFERENCES

Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

Venkat Chandrasekaran, Sujay Sanghavi, Pablo A Parrilo, and Alan S Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.

Shi Chen and Qi Zhao. Shallowing deep networks: Layer-wise pruning based on feature representations. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):3048–3056, 2018.

Hongwei Cheng, Zydrunas Gimbutas, Per-Gunnar Martinsson, and Vladimir Rokhlin. On the compression of low rank matrices. *SIAM Journal on Scientific Computing*, 26(4):1389–1404, 2005.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017.

Tejalal Choudhary, Vipul Mishra, Anurag Goswami, and Jagannathan Sarangapani. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, 53:5113–5155, 2020.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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

Siqi Fan, Xin Jiang, Xiang Li, Xuying Meng, Peng Han, Shuo Shang, Aixin Sun, Yequan Wang, and Zhongyuan Wang. Not all layers of llms are necessary during inference. *arXiv preprint arXiv:2403.02181*, 2024.

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 Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pp. 3259–3269. PMLR, 2020.

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.

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

Song Han. *Efficient methods and hardware for deep learning*. PhD thesis, Stanford University, 2017.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015a.

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, 2015b.

Matan Haroush, Itay Hubara, Elad Hoffer, and Daniel Soudry. The knowledge within: Methods for data-free model compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8494–8502, 2020.

Babak Hassibi, David G Stork, and Gregory J Wolff. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pp. 293–299. IEEE, 1993.

Yang He and Lingao Xiao. Structured pruning for deep convolutional neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2023.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397, 2017.

Geoffrey Hinton. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Yerlan Idelbayev and Miguel A Carreira-Perpinán. Low-rank compression of neural nets: Learning the rank of each layer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8049–8059, 2020.

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. Shortened llama: A simple depth pruning for large language models. *arXiv preprint arXiv:2402.02834*, 2024a.

Jinuk Kim, Marwa El Halabi, Mingi Ji, and Hyun Oh Song. Layermerge: Neural network depth compression through layer pruning and merging. *arXiv preprint arXiv:2406.12837*, 2024b.

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. Sparse upcycling: Training mixture-of-experts from dense checkpoints. *arXiv preprint arXiv:2212.05055*, 2022.

Eldar Kurtic and Dan Alistarh. Gmp*: Well-tuned gradual magnitude pruning can outperform most bert-pruning methods. *arXiv preprint arXiv:2210.06384*, 2022.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Yixiao Li, Yifan Yu, Qingru Zhang, Chen Liang, Pengcheng He, Weizhu Chen, and Tuo Zhao. Losparse: Structured compression of large language models based on low-rank and sparse approximation. In *International Conference on Machine Learning*, pp. 20336–20350. PMLR, 2023.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6: 87–100, 2024.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pp. 2736–2744, 2017.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*, 2024.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey. *arXiv preprint arXiv:2402.06196*, 2024.

Mohammad Mozaffari, Amir Yazdanbakhsh, Zhao Zhang, and Maryam Mehri Dehnavi. Slope: Double-pruned sparse plus lazy low-rank adapter pretraining of llms. *arXiv preprint arXiv:2405.16325*, 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. *arXiv preprint arXiv:2407.14679*, 2024.

Peter J. Olver and Chehrzad Shakiban. *Applied Linear Algebra*. Undergraduate Texts in Mathematics. Springer International Publishing, Cham, 2018. ISBN 978-3-319-91040-6 978-3-319-91041-3. doi: 10.1007/978-3-319-91041-3. URL http://link.springer.com/10.1007/978-3-319-91041-3.

Chong Peng, Yongyong Chen, Zhao Kang, Chenglizhao Chen, and Qiang Cheng. Robust principal component analysis: A factorization-based approach with linear complexity. *Information Sciences*, 513:581–599, 2020.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.

Wei Sun, Aojun Zhou, Sander Stuijk, Rob Wijnhoven, Andrew O Nelson, Henk Corporaal, et al. Dominosearch: Find layer-wise fine-grained n: M sparse schemes from dense neural networks. *Advances in neural information processing systems*, 34:20721–20732, 2021.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge fusion of large language models. *arXiv preprint arXiv:2401.10491*, 2024a.

Fanqi Wan, Ziyi Yang, Longguang Zhong, Xiaojun Quan, Xinting Huang, and Wei Bi. Fusechat: Knowledge fusion of chat models. *arXiv preprint arXiv:2402.16107*, 2024b.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

Lu Yin, Ajay Kumar Jaiswal, Shiwei Liu, Souvik Kundu, and Zhangyang Wang. Junk DNA hypothesis: Pruning small pre-trained weights irreversibly and monotonically impairs "difficult" downstream tasks in llms. In *Forty-first International Conference on Machine Learning*, 2024.

Zhiyuan Zha, Bihan Wen, Jiachao Zhang, Jiantao Zhou, and Ce Zhu. A comparative study for the nuclear norms minimization methods. In *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 2050–2054. IEEE, 2019.

Stephen Zhang and Vardan Papyan. OATS: Outlier-Aware Pruning Through Sparse and Low Rank Decomposition, September 2024. URL http://arxiv.org/abs/2409.13652. arXiv:2409.13652 [cs].

Zhuangwei Zhuang, Mingkui Tan, Bohan Zhuang, Jing Liu, Yong Guo, Qingyao Wu, Junzhou Huang, and Jinhui Zhu. Discrimination-aware channel pruning for deep neural networks. *Advances in neural information processing systems*, 31, 2018.

The appendix is organized into multiple sections, with each section offering additional details.

A Theoretical Analysis

In this section, we provide the proofs for the theorems presented in the main text and additional lemmas.

A.1 Additional Lemmas

In this subsection, we present additional lemmas that are useful for the proofs of the theorems in the main text.

Lemma 1 (Mask Norm Inequality I) *For any matrix $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ and binary mask $\boldsymbol{P} \in \{0,1\}^{m\times n}$, the following inequality holds:*

$$\|\boldsymbol{P} \odot \boldsymbol{M}\|_F \leq \|\boldsymbol{M}\|_F \tag{11}$$

Proof 1 (Proof of Lemma 1) *Let m_{ij} and p_{ij} be the elements of $\boldsymbol{M}$ and $\boldsymbol{P}$ respectively. By definition of the Frobenius norm and element-wise multiplication:*

$$\|\boldsymbol{P} \odot \boldsymbol{M}\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}(p_{ij}m_{ij})^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}^2 m_{ij}^2 \tag{12}$$

Since $\boldsymbol{P}$ is a binary mask, $p_{ij} \in \{0,1\}$, which means $p_{ij}^2 = p_{ij}$. Therefore:

$$\|\boldsymbol{P} \odot \boldsymbol{M}\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}m_{ij}^2 \leq \sum_{i=1}^{m}\sum_{j=1}^{n}m_{ij}^2 = \|\boldsymbol{M}\|_F^2 \tag{13}$$

Taking the square root of both sides preserves the inequality:

$$\|\boldsymbol{P} \odot \boldsymbol{M}\|_F \leq \|\boldsymbol{M}\|_F \tag{14}$$

□

Proof 2 (Proof of Lemma 2) *Let m_{ij} and p_{ij} be the elements of $\boldsymbol{M}$ and $\boldsymbol{P}$ respectively. By definition of the Frobenius inner product and element-wise multiplication:*

$$\langle \boldsymbol{M}, \boldsymbol{P} \odot \boldsymbol{M}\rangle_F = \sum_{i=1}^{m}\sum_{j=1}^{n}m_{ij}(p_{ij}m_{ij}) = \sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}m_{ij}^2 \leq \sum_{i=1}^{m}\sum_{j=1}^{n}m_{ij}^2 = \langle \boldsymbol{M}, \boldsymbol{M}\rangle_F \tag{15}$$

□

Lemma 2 (Mask Norm Inequality II) *For any matrix $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ and binary mask $\boldsymbol{P} \in \{0,1\}^{m\times n}$, the following inequality holds:*

$$\langle \boldsymbol{M}, \boldsymbol{P} \odot \boldsymbol{M}\rangle_F \leq \langle \boldsymbol{M}, \boldsymbol{M}\rangle_F \tag{16}$$

Lemma 3 (Mask Norm Equality I) *For any matrix $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ and binary mask $\boldsymbol{P} \in \{0,1\}^{m\times n}$, the following equality holds:*

$$\langle \boldsymbol{M}, \boldsymbol{P} \odot \boldsymbol{M}\rangle_F = \|\boldsymbol{P} \odot \boldsymbol{M}\|_F^2 \tag{17}$$

Proof 3 (Proof of Lemma 3) *Because $\boldsymbol{P}$ is a binary mask, we have $p_{ij}^2 = p_{ij}$. Therefore, we can rewrite the Frobenius inner product as:*

$$\langle \boldsymbol{M}, \boldsymbol{P} \odot \boldsymbol{M}\rangle_F = \sum_{i=1}^{m}\sum_{j=1}^{n}m_{ij}(p_{ij}m_{ij}) = \sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}^2 m_{ij}^2 = \|\boldsymbol{P} \odot \boldsymbol{M}\|_F^2 \tag{18}$$

□

Lemma 4 (Mask Norm Equality II) *For any matrix $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m\times n}$ and binary mask $\boldsymbol{P} \in \{0,1\}^{m\times n}$, the following equality holds:*

$$\langle \boldsymbol{P} \odot \boldsymbol{A}, \boldsymbol{P} \odot \boldsymbol{B}\rangle_F = \langle \boldsymbol{A}, \boldsymbol{P} \odot \boldsymbol{B}\rangle_F = \langle \boldsymbol{P} \odot \boldsymbol{A}, \boldsymbol{B}\rangle_F \tag{19}$$

Proof 4 (Proof of Lemma 4) *By definitions of Frobenius inner product and element-wise multiplication, we have:*

$$\langle \boldsymbol{P} \odot \boldsymbol{A}, \boldsymbol{P} \odot \boldsymbol{B}\rangle_F = \sum_{i=1}^{m}\sum_{j=1}^{n}(p_{ij}a_{ij})(p_{ij}b_{ij}) = \sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}^2 a_{ij}b_{ij} \tag{20}$$

$$\langle \boldsymbol{A}, \boldsymbol{P} \odot \boldsymbol{B}\rangle_F = \sum_{i=1}^{m}\sum_{j=1}^{n}a_{ij}(p_{ij}b_{ij}) = \sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}a_{ij}b_{ij} \tag{21}$$

$$\langle \boldsymbol{P} \odot \boldsymbol{A}, \boldsymbol{B}\rangle_F = \sum_{i=1}^{m}\sum_{j=1}^{n}(p_{ij}a_{ij})b_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}a_{ij}b_{ij} \tag{22}$$

For binary mask, $p_{ij}^2 = p_{ij}$, so the three expressions are equivalent. □

A.2 Proofs of Theorems

Proof 5 (Proof of Theorem 1) *We prove this by induction. For $t = 0$, $\boldsymbol{S}^{(0)} = \boldsymbol{W} \odot \boldsymbol{P}$, so the property holds. Assume the property holds for iteration t, i.e., $\boldsymbol{S}^{(t)} = \boldsymbol{S}^{(t)} \odot \boldsymbol{P}$. We need to prove that it holds for iteration $t+1$, i.e., $\boldsymbol{S}^{(t+1)} = \boldsymbol{S}^{(t+1)} \odot \boldsymbol{P}$. Therefore, we have:*

$$\begin{aligned}
\boldsymbol{S}^{(t+1)} &= \boldsymbol{S}^{(t)} + \boldsymbol{P} \odot \left\{\boldsymbol{U}_{r^{(t)}:}\left(\boldsymbol{L}^{(t)}\right)\boldsymbol{\Sigma}_{r^{(t)}:}\left(\boldsymbol{L}^{(t)}\right)\boldsymbol{V}^{\top}_{r^{(t)}:}\left(\boldsymbol{L}^{(t)}\right)\right\} \\
&= (\boldsymbol{S}^{(t)} \odot \boldsymbol{P}) + \boldsymbol{P} \odot (\textit{SVD terms}) \quad \textit{(applying induction hypothesis)} \\
&= \boldsymbol{P} \odot (\boldsymbol{S}^{(t)} + \textit{SVD terms}) = \boldsymbol{P} \odot \boldsymbol{S}^{(t+1)}
\end{aligned}$$

Thus, we have shown that if the property holds for t, it also holds for $t+1$. Combined with the base case, this completes the induction proof, showing that $\boldsymbol{S}^{(t)} = \boldsymbol{S}^{(t)} \odot \boldsymbol{P}$ for all t. □

Proof 6 (Proof of Theorem 2) *Let $\boldsymbol{S}^*$ and $\boldsymbol{L}^*$ be the limits of $\boldsymbol{S}^{(T)}$ and $\boldsymbol{L}^{(T)}$ as $T \to \infty$, respectively. We will show that these limits exist and satisfy the stated properties. First, note that for any t, $\boldsymbol{S}^{(t)} = \boldsymbol{S}^{(t)} \odot \boldsymbol{P}$ by construction of the algorithm. Let $\epsilon_t = \|\boldsymbol{S}^{(t+1)} - \boldsymbol{S}^{(t)}\|_F$. We need to show that $\lim_{T\to\infty}\epsilon_{T-1} = 0$. At each iteration t, we have:*

$$\boldsymbol{S}^{(t+1)} = \boldsymbol{S}^{(t)} + \boldsymbol{P} \odot \left\{\boldsymbol{U}_{r^{(t)}:}\left(\boldsymbol{L}^{(t)}\right)\boldsymbol{\Sigma}_{r^{(t)}:}\left(\boldsymbol{L}^{(t)}\right)\boldsymbol{V}^{\top}_{r^{(t)}:}\left(\boldsymbol{L}^{(t)}\right)\right\} \tag{23}$$

$$= \boldsymbol{S}^{(t)} + \boldsymbol{P} \odot \left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{r^{(t)}:} \tag{24}$$

where $(\cdot)_{r^{(t)}:}$ denotes the truncated SVD reconstruction from $r^{(t)}$ onwards. From the properties of SVD, we can express the difference between the truncated SVDs as:

$$\left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{r^{(t)}:} - \left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{k:} = \sum_{i=r^{(t)}+1}^{k}\sigma_i^{(t)}\boldsymbol{u}_i^{(t)}{\boldsymbol{v}_i^{(t)}}^{\top}, \tag{25}$$

where $\sigma_i^{(t)}$, $\boldsymbol{u}_i^{(t)}$, and $\boldsymbol{v}_i^{(t)}$ are the i-th singular value and corresponding left and right singular vectors of $\boldsymbol{W} - \boldsymbol{S}^{(t)}$. The Frobenius norm of this difference is:

$$\left\|\sum_{i=r^{(t)}+1}^{k}\sigma_i^{(t)}\boldsymbol{u}_i^{(t)}{\boldsymbol{v}_i^{(t)}}^{\top}\right\|_F^2 = \sum_{i=r^{(t)}+1}^{k}{\sigma_i^{(t)}}^2. \tag{26}$$

While we cannot directly guarantee that this sum approaches zero as t increases, we can bound it as:

$$\sum_{i=r^{(t)}+1}^{k}{\sigma_i^{(t)}}^2 \le \left(k - r^{(t)}\right)\left(\sigma_{r^{(t)}+1}^{(t)}\right)^2. \tag{27}$$

As $t \to T-1$, $r^{(t)} \to k$, so $k - r^{(t)} \to 0$. Thus, for any $\epsilon > 0$, there exists a T_0 such that for all $t > T_0$, we have $\left(k - r^{(t)}\right)\left(\sigma^{(t)}_{r^{(t)}+1}\right)^2 < \epsilon^2$. Now, considering the effect of the binary mask $\boldsymbol{P}$, according to Lemma. 1, we have:

$$\left\|\boldsymbol{P} \odot \left(\left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{r^{(t)}:} - \left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{k:}\right)\right\|_F \leq \left\|\left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{r^{(t)}:} - \left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{k:}\right\|_F < \epsilon \tag{28}$$

Setting $\delta = \epsilon$, we have shown that for any $\delta > 0$, there exists a T_0 such that for all $t > T_0$:

$$\left\|\boldsymbol{P} \odot \left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{r^{(t)}:} - \boldsymbol{P} \odot \left(\boldsymbol{W} - \boldsymbol{S}^{(t)}\right)_{k:}\right\|_F < \delta \tag{29}$$

This implies that for $t > T_0$:

$$\epsilon_t = \|\boldsymbol{S}^{(t+1)} - \boldsymbol{S}^{(t)}\|_F < \delta \tag{30}$$

Since δ can be arbitrarily small, we conclude that $\lim_{T\to\infty} \epsilon_{T-1} = 0$. This shows that the sequence $\{\boldsymbol{S}^{(T)}\}$ is Cauchy and therefore converges to some limit $\boldsymbol{S}^*$. Since $\boldsymbol{W} = \boldsymbol{S}^{(t)} + \boldsymbol{L}^{(t)}$ for all t, and $\boldsymbol{S}^{(T)}$ converges to $\boldsymbol{S}^*$, it follows that $\boldsymbol{L}^{(T)}$ must converge to $\boldsymbol{L}^* = \boldsymbol{W} - \boldsymbol{S}^*$. Finally, since $\boldsymbol{S}^{(t)} = \boldsymbol{S}^{(t)} \odot \boldsymbol{P}$ for all t, we have $\boldsymbol{S}^* = \boldsymbol{S}^* \odot \boldsymbol{P}$. Thus, the algorithm converges to the solution $(\boldsymbol{S}^*, \boldsymbol{L}^*)$ as $T \to \infty$, satisfying all the stated properties. □

Proof 7 (Proof of Theorem 3) *Let $\boldsymbol{E}^{(t)} = \boldsymbol{W} - \left(\boldsymbol{S}^{(t)} + \boldsymbol{L}^{(t)}_k\right) = \left(\boldsymbol{L}^{(t)}\right)_{k:}$ be the error at iteration t. From the update rule in Eq.(7), we have:*

$$\boldsymbol{E}^{(t+1)} = \boldsymbol{W} - \left(\boldsymbol{S}^{(t+1)} + \boldsymbol{L}^{(t+1)}_k\right) \tag{31}$$

$$= \boldsymbol{W} - \left(\boldsymbol{S}^{(t)} + \boldsymbol{P} \odot \left(\boldsymbol{L}^{(t)}\right)_{r^{(t)}:} + \boldsymbol{L}^{(t+1)}_k\right) \tag{32}$$

$$= \boldsymbol{L}^{(t)} - \boldsymbol{P} \odot \left(\boldsymbol{L}^{(t)}\right)_{r^{(t)}:} - \boldsymbol{L}^{(t+1)}_k \tag{33}$$

Let's denote $\boldsymbol{A} = \boldsymbol{P} \odot \left(\boldsymbol{L}^{(t)}\right)_{r^{(t)}:}$. Then $\boldsymbol{E}^{(t+1)} = (\boldsymbol{L}^{(t)} - \boldsymbol{A}) - \boldsymbol{L}^{(t+1)}_k$. By construction, $\boldsymbol{L}^{(t+1)}_k$ is the best rank-k approximation of $\boldsymbol{L}^{(t+1)} = \boldsymbol{L}^{(t)} - \boldsymbol{A}$. Therefore:

$$\left\|\boldsymbol{E}^{(t+1)}\right\|_F = \left\|\left(\boldsymbol{L}^{(t)} - \boldsymbol{A}\right) - \boldsymbol{L}^{(t+1)}_k\right\|_F \leq \left\|\left(\boldsymbol{L}^{(t)} - \boldsymbol{A}\right) - \boldsymbol{L}^{(t)}_k\right\|_F \tag{34}$$

Now, $\boldsymbol{L}^{(t)} - \boldsymbol{L}^{(t)}_k = \boldsymbol{E}^{(t)}$, so we have:

$$\left\|\boldsymbol{E}^{(t+1)}\right\|^2_F \leq \left\|\left(\boldsymbol{L}^{(t)} - \boldsymbol{A}\right) - \boldsymbol{L}^{(t)}_k\right\|^2_F = \left\|(\boldsymbol{L}^{(t)} - \boldsymbol{L}^{(t)}_k) - \boldsymbol{A}\right\|^2_F \tag{35}$$

$$= \left\|\boldsymbol{E}^{(t)} - \boldsymbol{A}\right\|^2_F = \left\|\boldsymbol{E}^{(t)}\right\|^2_F + \|\boldsymbol{A}\|^2_F - 2\left\langle \boldsymbol{E}^{(t)}, \boldsymbol{A}\right\rangle_F \tag{36}$$

According to Lemma 2 and $\boldsymbol{A} = \boldsymbol{P} \odot \left(\boldsymbol{L}^{(t)}_{r^{(t)}:k} + \boldsymbol{E}^{(t)}\right)$, we have:

$$\left\langle \boldsymbol{E}^{(t)}, \boldsymbol{A}\right\rangle_F = \left\langle \boldsymbol{E}^{(t)}, \boldsymbol{P} \odot \left(\boldsymbol{L}^{(t)}_{r^{(t)}:k} + \boldsymbol{E}^{(t)}\right)\right\rangle_F \tag{37}$$

$$= \left\langle \boldsymbol{E}^{(t)}, \boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}\right\rangle_F + \left\langle \boldsymbol{E}^{(t)}, \boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\rangle_F \tag{38}$$

$$= \left\langle \boldsymbol{E}^{(t)}, \boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}\right\rangle_F + \left\|\boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\|^2_F \tag{39}$$

The last equality follows from Lemma 3. Substituting this back into our earlier inequality Eq.(36):

$$\left\|\boldsymbol{E}^{(t+1)}\right\|^2_F \leq \left\|\boldsymbol{E}^{(t)}\right\|^2_F + \|\boldsymbol{A}\|^2_F - 2\left\langle \boldsymbol{E}^{(t)}, \boldsymbol{A}\right\rangle_F \tag{40}$$

$$\leq \left\|\boldsymbol{E}^{(t)}\right\|^2_F + \|\boldsymbol{A}\|^2_F - 2\left\langle \boldsymbol{E}^{(t)}, \boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}\right\rangle_F - 2\left\|\boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\|^2_F \tag{41}$$

$$= \left\|\boldsymbol{E}^{(t)}\right\|^2_F + \left\|\boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}\right\|^2_F + \left\|\boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\|^2_F + \cancel{2\left\langle \boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}, \boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\rangle_F}$$

$$\cancel{-2\left\langle \boldsymbol{E}^{(t)}, \boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}\right\rangle_F} - 2\left\|\boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\|^2_F \quad \textit{(by Lemma. 4)} \tag{42}$$

$$= \left\|\boldsymbol{E}^{(t)}\right\|^2_F + \left\|\boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k}\right\|^2_F - \left\|\boldsymbol{P} \odot \boldsymbol{E}^{(t)}\right\|^2_F \tag{43}$$

As $t \to T-1$, $r^{(t)} \to k$, so $\left\| \boldsymbol{P} \odot \boldsymbol{L}^{(t)}_{r^{(t)}:k} \right\|_F^2 \to 0$. Therefore, there exists a time step T_0 such that for all $t > T_0$, $\left\| \boldsymbol{E}^{(t+1)} \right\|_F^2 - \left\| \boldsymbol{E}^{(t)} \right\|_F^2 \leq - \left\| \boldsymbol{P} \odot \boldsymbol{E}^{(t)} \right\|_F^2 \leq 0$. □

Proof 8 (Proof of Theorem 4) *At iteration t, we have $\boldsymbol{W} = \boldsymbol{S}^{(t)} + \boldsymbol{L}^{(t)}$ by construction. Let $\boldsymbol{L}^{(t)} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ be the SVD of $\boldsymbol{L}^{(t)}$. Then $\boldsymbol{L}^{(t)}_k = \boldsymbol{U}_{:k}\boldsymbol{\Sigma}_{:k}\boldsymbol{V}^\top_{:k}$ is the best rank-k approximation of $\boldsymbol{L}^{(t)}$. The error can be expressed as:*

$$\left\| \boldsymbol{W} - \left(\boldsymbol{S}^{(t)} + \boldsymbol{L}^{(t)}_k \right) \right\|_F = \left\| \left(\boldsymbol{S}^{(t)} + \boldsymbol{L}^{(t)} \right) - \left(\boldsymbol{S}^{(t)} + \boldsymbol{L}^{(t)}_k \right) \right\|_F \tag{44}$$

$$= \left\| \boldsymbol{L}^{(t)} - \boldsymbol{L}^{(t)}_k \right\|_F = \| \boldsymbol{U}_{k:}\boldsymbol{\Sigma}_{k:}\boldsymbol{V}^\top_{k:} \|_F \tag{45}$$

$$= \sqrt{\sum_{i=k+1}^{r} \sigma_i^2 \left(\boldsymbol{L}^{(t)} \right)} \leq \sqrt{(r-k)}\sigma_{k+1} \left(\boldsymbol{L}^{(t)} \right) \tag{46}$$

□