Input Compensation for Pruned Models

Anonymous authors
Paper under double-blind review

Abstract

Though foundation models are powerful, they are large and require substantial memory and computation resources for serving. To tackle this issue, many pruning methods have been proposed to reduce the model size, thereby achieving memory and computational efficiency. These methods either identify and retrain the important weights or *adjust the unpruned weights* to compensate for the removed weights. In this paper, we propose a novel approach called input compensation (IC) to boost the performance of pruned models, i.e., *adjust the input* to compensate for the removed weights. We learn a compensation pool to construct input-dependent compensation to reduce the error caused by pruning. Different from existing pruning methods, which are designed in the parameter space, the proposed IC is designed in the input space. Hence, IC is complementary to existing methods and can be integrated with them. Extensive experiments on various tasks, including image classification, language modeling, and image generation, demonstrate that IC is effective in improving the performance of pruned models.

1 Introduction

Foundation models (Baevski et al., 2020; Radford et al., 2021; Touvron et al., 2023b; Podell et al., 2024) have achieved great success in a variety of domains such as computer vision, natural language processing, and speech recognition. As the availability of data and computational resources expands, these models have scaled in both size and performance (Touvron et al., 2023a;b; Meta, 2024). However, the substantial number of parameters in these models require extensive computational resources for serving, posing a significant challenge to deploy them on resource-constraint devices such as smartphones and laptops. To reduce the costs, numerous *model compression* techniques have been proposed to reduce the model size, e.g., distillation (Polino et al., 2018; Wang et al., 2019; Liang et al., 2023), quantization (Lin et al., 2024; Dettmers et al., 2022; Shao et al., 2024; Xiao et al., 2023), and pruning (Han et al., 2015; Frantar & Alistarh, 2023; Zhang et al., 2024; Sun et al., 2024). As quantization needs specialized hardware supports and distillation requires extensive retraining, we focus on pruning, which is a simple and representative technique.

Pruning reduces the model size by removing individual weights or rows/columns according to their importance scores. A pruned model can achieve promising performance with fewer parameters, resulting in a noticeable reduction in memory and computational demands. A simple but effective pruning method is Magnitude Pruning (Han et al., 2015) which removes weights according to their magnitudes. The underlying assumption is that weights with smaller values contribute less to the overall performance. However, this assumption does not always hold and many advanced methods (Sun et al., 2024; Frantar & Alistarh, 2023; Zhang et al., 2024) have been proposed recently.

Current state-of-the-art pruning methods (Frantar & Alistarh, 2023; Das et al., 2023; Zhang et al., 2024; Sun et al., 2024; Dong et al., 2024; An et al., 2024) focus on the *parameter space* to enhance pruning efficacy and can be roughly categorized into two groups: (i) designing an effective score to measure the importance of weight and (ii) adjusting the remaining unpruned weights to reduce the error caused by the pruned weights. For example, Wanda (Sun et al., 2024) designs an importance score to incorporate input activations with weight magnitude to take outlier features into consideration, instead of only weight magnitudes in Magnitude Pruning; SparseGPT (Frantar & Alistarh, 2023) proposes to adjust the unpruned weights by minimizing a reconstruction loss using the Optimal Brain Surgeon framework (Hassibi et al., 1993; Singh & Alistarh, 2020; Frantar et al., 2021). The pruned model can be formulated as $\mathcal{F}(\mathbf{X}; \mathbf{W} \odot \mathbf{M} + \mathbf{\Delta_w})$, where $\mathcal{F}$ is the model, $\mathbf{X}$ is the input, $\mathbf{W}$ is the weight matrix, $\mathbf{M}$ is the weight mask determined by the importance score, $\odot$ is element-wise multiplication, and $\boldsymbol{\Delta}_{\mathbf{w}}$ (called *weight compensation*) is an update matrix for the unpruned weights.

In this paper, we propose a novel method called input compensation (IC) for enhancing pruned models by adjusting the input to compensate for the removed weights. Specifically, the output of the pruned model is determined by $\mathcal{F}(\mathbf{X} + \boldsymbol{\Delta}_{\mathbf{x}}; \hat{\mathbf{W}})$, where $\boldsymbol{\Delta}_{\mathbf{x}}$ is an input compensation for adjusting the original input and $\hat{\mathbf{W}}$ is a sparse weight matrix corresponding to the pruned model. We learn a compensation pool consists of multiple candidate compensations from calibration data and $\boldsymbol{\Delta}_{\mathbf{x}}$ is a weighted combination of the candidate compensations via the attention mechanism (Vaswani et al., 2017).

Different from existing pruning methods, the proposed IC is designed in the *input space*. Hence, IC is complementary to existing methods that operate in the parameter space and can be integrated with them to boost their performance. Extensive experiments on computer vision and natural language processing show that IC brings a large improvement to existing pruning methods.

Our contributions are summarized as follows: (i) We propose IC which is a novel direction to enhance pruned models; (ii) IC is designed in the input space and, thus, is orthogonal to existing pruning methods designed in the parameter space. Hence, IC can be combined with existing pruning methods; (iii) Experimental results on various tasks demonstrate that IC is beneficial to existing pruning methods.

2 Related Work

Foundation Models are large pre-trained models designed to serve as base models for various downstream tasks. These models are typically trained on a large amount of data and contain massive of parameters. Notable examples include Large Language Models (LLMs) like LLaMA series (Touvron et al., 2023a;b; Meta, 2024), which have promising performance in natural language processing tasks such as text generation (Li et al., 2024; Zhang et al., 2023), understanding (Guo et al., 2024; Fan & Hunter, 2023), and reasoning (Wei et al., 2022; Yu et al., 2024). In the realm of computer vision (CV), models like CLIP (Contrastive Language-Image Pretraining) (Radford et al., 2021) use multimodal learning to bridge textual and visual information, enhancing various CV tasks such as image classification (Radford et al., 2021), image captioning and visual question answering (Li et al., 2022; 2023a). Additionally, diffusion models like DDPM (Ho et al., 2020), Stable Diffusion (Rombach et al., 2022), and SDXL (Podell et al., 2024) have revolutionized image generation by employing a process of gradually transforming noise into images, showing the diverse applications of foundation models in creative applications.

Model Compression. Though foundation models are powerful, their massive of parameters usually require extensive computational and memory resources. Many recent efforts have been devoted to reducing the cost via model compression (Frantar & Alistarh, 2022; Xu et al., 2024; Wang et al., 2024). The most popular methods for model compression are pruning, quantization, and distillation. *Pruning* (Han et al., 2015; Zhang et al., 2024; Sun et al., 2024; Dong et al., 2024; Das et al., 2023; An et al., 2024; Frantar & Alistarh, 2023) discards parts of the model that are less important or redundant. *Quantization* (Lin et al., 2024; Dettmers et al., 2022; Shao et al., 2024; Xiao et al., 2023; Yao et al., 2022; Kim et al., 2024) is a technique to reduce the computational complexity and memory footprint of a neural network by converting the model's parameters (weights and activations) from higher-precision representations (such as 32-bit floating-point) to lower-precision ones (such as 8-bit integers). The primary goal of quantization and pruning is to make the model more compressed without significantly sacrificing its performance. *Distillation* (Polino et al., 2018; Wang et al., 2019; Liang et al., 2023) trains a smaller and more efficient model to replicate the behavior of a larger and more complex model, thereby retaining much of its performance while significantly reducing computational resources. Quantization demands specialized hardware (e.g., NVIDIA TensorRT[1]) that supports lower precision arithmetic, while distillation requires an expensive training phase to transfer knowledge from a large teacher model to a small student model. In this paper, we focus on pruning, which is a simple and widely used method.

[1] https://github.com/NVIDIA/TensorRT

Pruning aims to remove less important weights without significant performance degradation. Several important metrics have been designed recently. The simplest one is based on the parameter magnitude, i.e., Magnitude Pruning (Han et al., 2015). Wanda (Sun et al., 2024) further incorporates weight magnitude with their input activations to consider outlier features when calculating importance scores, while RIA (Zhang et al., 2024) uses relative importance as a pruning metric. Taylor pruning (Molchanov et al., 2022) designs a score based on the weight multiplied by its gradient, while Diff-Pruning (Fang et al., 2023) further uses Taylor expansion over pruned timesteps to identify and discard unimportant parameters. In addition to designing importance scores to find less useful parameters, one can *update the unpruned weights* to compensate for the error caused by the pruned weights. For example, SparseGPT (Frantar & Alistarh, 2023) and OBC (Frantar & Alistarh, 2022) propose to update the unpruned weights by minimizing a reconstruction loss by the Optimal Brain Surgeon framework (Hassibi et al., 1993; Singh & Alistarh, 2020; Frantar et al., 2021). Different from SparseGPT and OBC, we propose input compensation by *adjusting the inputs* to reduce the error caused by pruning.

Prompting (Radford et al., 2019; Brown et al., 2020; Liu et al., 2022; Ding et al., 2022) is a popular method used in transformer-based models which inserts additional tokens that instruct the model to generate a specific kind of response. These tokens can be either discrete tokens (e.g., "The topic is" for topic classification (Zhang et al., 2022a; Hou et al., 2022; Jiang et al., 2023), "Let's think step by step" for reasoning tasks (Kojima et al., 2022)) or learnable continuous vectors (e.g., prompt tuning (Lester et al., 2021; Liu et al., 2021; Zhang et al., 2022b) or prefix learning (Li & Liang, 2021; Liu et al., 2023)). Unlike prompting that inserts extra tokens into the inputs, our input compensation edits the inputs directly. Furthermore, compensations are input-dependent, while prompts are usually input-independent (Ding et al., 2022; Lester et al., 2021; Liu et al., 2021; Zhang et al., 2022b; Bahng et al., 2022).

In control systems, the idea of **input compensation** (Kuo & Golnaraghi, 1995; Franklin et al., 2002) is practically used to adjust the control signal to reduce the influence of disturbance. The goal is to adjust the input such that the overall system achieves desired behavior, such as better stability, faster response, or improved accuracy. For example, in feedforward compensation (Campos & Lewis, 1999; Krstic, 2009), if a disturbance is known ahead of time (e.g., wind gusts affecting an airplane), this information can be incorporated into the control signal so that the system compensates for it before it affects the output. In model pruning, the pruned weights can be viewed as disturbances and we use input compensation to enhance pruned models.

3 Preliminary on Model Pruning

Let $\mathbf{W} \in \mathbb{R}^{d_i \times d_o}$ be a weight matrix of a model $\mathcal{F}$ and $\mathbf{S}$ be a scoring matrix whose $\mathbf{S}_{i,j}$ measures the importance of $\mathbf{W}_{i,j}$. To prune $p\%$ parameters of $\mathbf{W}$, we determine a threshold β satisfies $\frac{\#\{\mathbf{S}_{i,j}:|\mathbf{S}_{i,j}|<\beta\}}{\#\{\mathbf{S}_{i,j}\}} = p\%$. Using the threshold, we construct a binary weight mask $\mathbf{M}$ whose $\mathbf{M}_{i,j} = 1$ if $|\mathbf{S}_{i,j}| \geq \beta$ else 0 and prune the model as $\mathbf{W} \odot \mathbf{M}$. To improve the performance of the pruned model, one can adjust the unpruned weights to compensate for the removed weights. Generally, the pruned model can be formulated as:

$$\mathcal{F}(\mathbf{X}; \mathbf{W} \odot \mathbf{M} + \mathbf{\Delta_w}), \tag{1}$$

where $\mathbf{\Delta_w}$ (called *weight compensation*) is an update matrix for the unpruned weights. Various pruning methods have been proposed to design an effective scoring metric or learn an effective weight compensation $\Delta_{\mathbf{w}}$, e.g., Han et al. (2015); Zhang et al. (2024); Sun et al. (2024); Dong et al. (2024); Das et al. (2023); An et al. (2024) for the former, and Frantar & Alistarh (2023; 2022) for the latter. We briefly review three representative pruning methods.

Magnitude Pruning (Han et al., 2015) is the simplest technique whose score matrix is defined as $\mathbf{S}_{i,j} = |\mathbf{W}_{i,j}|$, i.e., removing the weights whose magnitudes are below a predefined threshold. In practice, magnitude pruning is performed in a layer-wise manner: for each layer, a layer-dependent threshold is determined based on the local distribution of weights. Though Magnitude pruning has stood out as a strong baseline for pruning models (Blalock et al., 2020), it has a major limitation: it ignores the importance of input activation, which plays an equally importance role as weight magnitudes in determining the output of linear layers (e.g., fully connected layers, attention layers).

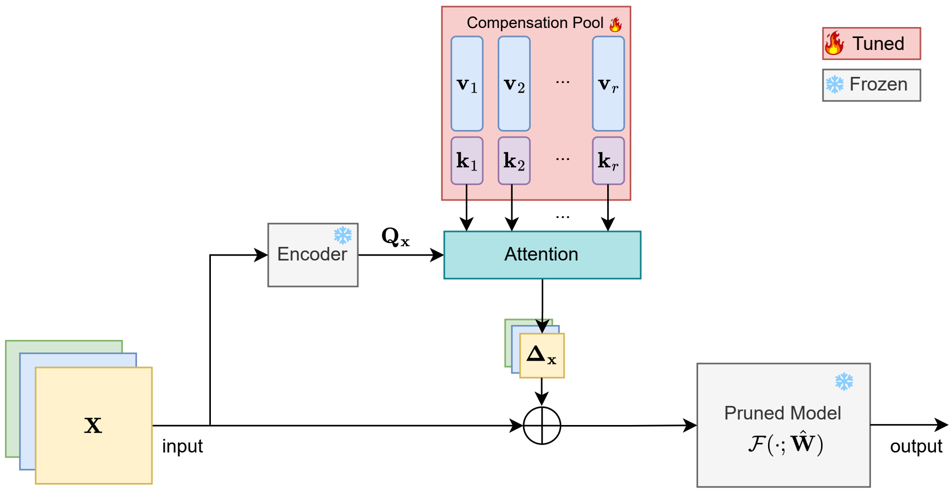


Figure 1: Input compensation for pruned models.

Wanda (Sun et al., 2024) addresses this limitation by incorporating both weights and inputs into defining the weight importance. Specifically, let $\mathbf{X} \in \mathbb{R}^{N \times d_i}$ (where N is the sequence length) be the input activation of a calibration sample. Consider a linear layer $\mathbf{Y} = \mathbf{X}\mathbf{W}$, Wanda defines the importance of $\mathbf{W}_{i,j}$ as $\mathbf{S}_{i,j} = |\mathbf{W}_{i,j}| \cdot \|\mathbf{X}_{:,i}\|_2$.

SparseGPT (Frantar & Alistarh, 2023) introduces a more sophisticated pruning approach by incrementally pruning each column of $\mathbf{W}$, followed by adjusting the remaining weights to compensate for those that have been pruned by the Optimal Brain Surgeon framework (Hassibi et al., 1993; Singh & Alistarh, 2020; Frantar et al., 2021). The score matrix is determined by $\mathbf{S}_{i,j} = \frac{|\mathbf{W}_{i,j}|^2}{[\mathbf{H}^{-1}]_{i,i}}$ and $\mathbf{H} = \mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}$ (λ is a small positive constant) is the Hessian matrix of the reconstruction loss.

4 Methodology

4.1 Input Compensation (IC)

Different from existing pruning methods, which primarily focus on learning a good scoring metric $\mathbf{S}$ or weight compensation $\mathbf{\Delta}_\mathbf{w}$ in the parameter space, we propose a novel direction to enhance model pruning by adjusting the input to compensate for the removed weights. Formally, let $\hat{\mathbf{W}}$ be a pruned model. Our objective is to determine an input compensation $\mathbf{\Delta}_\mathbf{x}$ for the input such that its output approximates that of the dense model, i.e.,

$$\mathcal{F}(\mathbf{X} + \mathbf{\Delta}_\mathbf{x}; \hat{\mathbf{W}}) \approx \mathcal{F}(\mathbf{X}; \mathbf{W}). \tag{2}$$

The compensation $\mathbf{\Delta}_\mathbf{x}$ depends on the input $\mathbf{X}$. Obviously, learning $\mathbf{\Delta}_\mathbf{x}$ from scratch for each sample is inefficient. To deal with this issue, we begin by developing a learning framework for IC within the context of a simple linear layer and subsequently extend this approach to more complex, general models.

Linear Layer. Recent studies (Yu et al., 2017; Li et al., 2023b; Ding et al., 2023) have shown that the weight matrix $\mathbf{W}$ of neural networks can be approximated by a combination of a sparse matrix $\mathbf{S} \in \mathbb{R}^{d_i \times d_o}$ (assume rank$(\mathbf{S}) = d_o$) and a low-rank matrix $\mathbf{A}\mathbf{B}^\top$ (where $\mathbf{A} \in \mathbb{R}^{d_i \times r}$ and $\mathbf{B} \in \mathbb{R}^{d_o \times r}$, r is the rank). Hence, for a linear layer, the output is approximated as

$$\mathbf{Y} = \mathbf{X}\mathbf{W} \approx \mathbf{X}(\mathbf{S}+\mathbf{A}\mathbf{B}^\top) = \mathbf{X}\mathbf{S} + \mathbf{X}\mathbf{A}\underbrace{\mathbf{B}^\top(\mathbf{S}^\top\mathbf{S})^{-1}\mathbf{S}^\top}_{\equiv\hat{\mathbf{B}}}\mathbf{S} = \Bigg(\mathbf{X} + \underbrace{\mathbf{X}\mathbf{A}\hat{\mathbf{B}}^\top}_{\text{i.e., } \mathbf{\Delta}_\mathbf{x}}\Bigg)\mathbf{S}. \tag{3}$$

Let $\mathbf{a}_i$ and $\hat{\mathbf{b}}_i$ be the ith column of $\mathbf{A}$ and $\hat{\mathbf{B}}$, respectively. The ith row of $\mathbf{\Delta_x}$ is computed as $\sum_{j=1}^{r}(\mathbf{x}_i^\top \mathbf{a}_j)\hat{\mathbf{b}}_j$, which is similar to the attention mechanism (Vaswani et al., 2017): $\{\mathbf{x}_i\}$ are the query, $\{\mathbf{a}_j\}$ are the keys, and $\{\hat{\mathbf{b}}_j\}$ are the values.

General Models. Building on insights from the linear layer, we propose a general IC framework based on the attention mechanism (Vaswani et al., 2017). Figure 1 provides an overview of the IC framework, which contains a frozen encoder $\mathcal{E}(\cdot)$ and a learnable compensation pool $(\mathbf{K}, \mathbf{V})$ (where $\mathbf{K} \in \mathbb{R}^{d_e \times r}$ and $\mathbf{V} \in \mathbb{R}^{r \times d_i}$). The encoder, which can either be a sub-module of the pruned model or an identity function, maps $\mathbf{X}$ into an embedding $\mathbf{Q_x} = \mathcal{E}(\mathbf{X}) \in \mathbb{R}^{N \times d_e}$, while the compensation pool consists of r candidate compensations. The input compensation is then constructed as:

$$\mathbf{\Delta_x} = \mathrm{softmax}\left(\frac{\mathbf{Q_x K}}{\sqrt{d_e}}\right)\mathbf{V}. \tag{4}$$

The input is adjusted by adding $\mathbf{\Delta_x}$, and the compensation pool is optimized by minimizing the following supervised loss:

$$\min_{\mathbf{K},\mathbf{V}} \sum_{(\mathbf{X},\mathbf{Y})\in\mathcal{D}} \ell(\mathcal{F}(\mathbf{X}+\mathbf{\Delta_x}; \hat{\mathbf{W}}), \mathbf{Y}), \tag{5}$$

where $\ell(\cdot,\cdot)$ is the supervised loss function. In cases where labels for $\mathbf{X}$ are unavailable, we can learn the pool by minimizing the reconstruction loss:

$$\min_{\mathbf{K},\mathbf{V}} \sum_{(\mathbf{X},\cdot)\in\mathcal{D}} \|\mathcal{F}(\mathbf{X}+\mathbf{\Delta_x}; \hat{\mathbf{W}}) - \mathcal{F}(\mathbf{X}; \mathbf{W})\|^2. \tag{6}$$

4.2 Application in LLMs

For NLP tasks, inputs are sequences of discrete tokens, making direct modification of inputs infeasible. To deal with this issue, we propose adjusting the input embeddings. Figure 7 in Appendix B provides an illustration of IC for LLMs. Let $\mathbf{H_x} \in \mathbb{R}^{N \times d_e}$ be the embeddings extracted by the input embedding layer of the pruned LLM. Similar to Eq.(4), we construct the input compensation for LLMs as $\mathbf{\Delta_x} = \mathrm{softmax}\left(\frac{\mathbf{H_x K}}{\sqrt{d_e}}\right)\mathbf{V}$. The input embeddings are then adjusted as $\mathbf{H} + \mathbf{\Delta_x}$ and we learn the compensation pool by minimizing the reconstruction loss of the last hidden states:

$$\min_{\mathbf{K},\mathbf{V}} \sum_{(\mathbf{X},\cdot)\in\mathcal{D}} \|\mathcal{F}(\mathbf{H_x}+\mathbf{\Delta_x}; \hat{\mathbf{W}}) - \mathcal{F}(\mathbf{H_x}; \mathbf{W})\|^2. \tag{7}$$

5 Experiments

5.1 Experiments on Image Classification

Datasets. We conduct image classification experiments on ten datasets: CIFAR100 (Krizhevsky & Hinton, 2009), Flowers (Nilsback & Zisserman, 2008), Food (Bossard et al., 2014), EuroSAT (Helber et al., 2019), SUN (Xiao et al., 2016), DTD (Cimpoi et al., 2014), UCF (Soomro et al., 2012), SVHN (Netzer et al., 2011), OxfordPets (Jawahar et al., 2012) (denoted by Pets), and RESISC45 (Cheng et al., 2017) (denoted by RESISC). A summary of the datasets is in Table 9 of Appendix A.

Implementation Details. We adopt CLIP ViT-B/32 and ViT-B/16 (Radford et al., 2021) as the base models, whose pruned image encoder is used as the encoder of IC. We initialize the $\mathbf{K}$ and $\mathbf{V}$ by the standard normal distribution and train the compensation pool for 30 epochs using the SGD optimizer with a learning rate of 40 and momentum of 0.9. The mini-batch size is 128. Following (Bahng et al., 2022), $\mathbf{v}_i$ is learnable padding pixels on all sides, where the padding size is set to 30. The rank r is chosen as 32 and a sensitivity analysis is provided in Section 6. We evaluate two types of sparsity: unstructured sparsity and structured 4:8 sparisty (Mishra et al., 2021), i.e., at most 4 out of every 8 contiguous weights to be non-zero.

Baselines. The proposed IC can be integrated into any existing pruning methods. To verify its effectiveness, we consider three pruning methods: (i) Magnitude Pruning (Han et al., 2015) which

Table 1: Testing accuracy on image classification tasks using CLIP ViT-B/32.

	Sparsity	CIFAR100	Flowers	Food	EuroSAT	SUN	UCF	SVHN	Pets	DTD	RESISC	**Avg**
Dense	0%	88.3	97.8	89.1	98.8	73.9	86.4	97.1	92.0	74.4	96.0	89.4
Magnitude	50%	33.9	26.1	34.2	45.6	30.8	35.4	45.3	38.7	27.9	55.4	37.3
Magnitude + IC	50%	73.0	62.9	72.4	96.5	48.9	63.1	94.4	69.2	44.1	87.1	**71.2**
Wanda	50%	75.0	56.4	74.1	95.2	50.8	59.7	91.8	57.6	43.4	84.4	68.9
Wanda + IC	50%	80.1	76.4	80.4	97.9	54.7	69.1	96.1	77.5	49.8	91.6	**77.4**
SparseGPT	50%	83.3	69.1	81.6	97.9	58.0	68.5	93.7	59.4	48.2	89.8	74.9
SparseGPT + IC	50%	82.9	76.2	83.1	98.2	57.2	71.0	96.7	79.7	53.8	92.9	**79.2**
Magnitude	4:8	49.0	25.9	36.5	45.1	32.8	37.8	60.8	45.3	27.1	60.2	42.1
Magnitude + IC	4:8	72.9	62.4	72.1	96.5	48.2	62.9	94.3	68.3	44.8	87.4	**71.0**
Wanda	4:8	60.9	30.5	59.2	83.1	37.2	43.2	74.4	47.0	30.9	68.7	53.5
Wanda + IC	4:8	76.9	71.4	77.3	97.2	50.2	64.2	95.2	75.3	51.1	89.5	**74.8**
SparseGPT	4:8	80.2	55.7	79.6	96.6	52.3	61.4	85.8	58.5	42.3	86.6	69.9
SparseGPT + IC	4:8	81.8	72.6	81.6	98.1	51.7	65.8	96.6	78.1	49.1	92.2	**76.8**

Table 2: Testing accuracy on image classification tasks using CLIP ViT-B/16.

	Sparsity	CIFAR100	Flowers	Food	EuroSAT	SUN	UCF	SVHN	Pets	DTD	RESISC	**Avg**
Dense	0%	90.1	98.7	91.9	98.8	75.1	87.8	97.7	93.8	76.1	96.7	90.7
Magnitude	50%	76.9	56.5	78.3	90.7	51.2	65.6	95.3	62.9	42.8	82.1	70.2
Magnitude + IC	50%	82.9	86.7	84.7	97.6	60.1	75.1	97.1	82.5	61.1	92.8	**82.1**
Wanda	50%	84.1	78.1	85.5	97.6	59.5	68.9	96.9	72.7	51.8	91.2	78.6
Wanda + IC	50%	86.2	82.8	87.8	98.4	63.8	75.5	97.6	83.6	63.5	94.7	**83.4**
SparseGPT	50%	87.2	80.2	88.1	98.0	63.8	73.8	97.0	75.6	56.4	93.7	81.4
SparseGPT + IC	50%	86.1	86.0	87.9	98.4	64.4	76.2	97.6	85.2	66.4	95.0	**84.3**
Magnitude	4:8	75.8	52.0	75.4	89.6	50.0	61.4	77.4	68.8	41.4	79.1	67.1
Magnitude + IC	4:8	81.5	84.2	83.2	97.5	57.6	72.5	96.9	81.6	54.4	91.9	**80.1**
Wanda	4:8	78.6	63.2	81.4	96.0	50.6	61.3	78.2	69.5	42.8	87.7	70.9
Wanda + IC	4:8	84.7	82.1	86.8	98.4	60.6	74.5	97.4	82.0	61.3	94.3	**82.2**
SparseGPT	4:8	85.1	74.0	87.0	95.6	60.4	69.8	72.6	78.2	50.5	93.7	76.7
SparseGPT + IC	4:8	84.7	84.9	87.1	98.3	60.9	74.9	97.5	83.6	62.2	94.4	**82.9**

discards weights based on their magnitudes; (ii) Wanda (Sun et al., 2024) designs a scoring metric as the weight magnitudes multiplied by the corresponding input activations on a per-output basis; (iii) SparseGPT (Frantar & Alistarh, 2023) which adjusts the unpruned weights by solving a layer-wise reconstruction problem using a second-order optimizer. SparseGPT is a weight compensation method, while Magnitude and Wanda design a scoring metric for pruning without updating weights. For all methods, the base models are fully finetuned on the training set of all tasks before pruning.

Results. Tables 1 and 2 show the testing accuracy on ten image classification tasks using CLIP ViT-B/32 and ViT-B/16, respectively. As can be seen, IC consistently brings large improvements to existing pruning methods in both unstructured (sparsity=50%) and structured (sparsity=4:8) cases. Specifically, compared with Magnitude, Magnitude + IC achieves improvements of 28% and 12% on ViT-B/32 and ViT-B/16, respectively; Compared with Wanda, Wanda + IC has improvements of about 5%; Compared with SparseGPT, SparseGPT + IC performs better by an improvement of 4% on ViT-B/32. The large improvements contributed by IC verify that the learned compensation pool is effective in constructing input compensation for the pruned models. Moreover, SparseGPT + IC consistently performs the best, demonstrating that combining both weight compensation and input compensation is more desirable. We can also observe that unstructured pruning (sparsity=50%) achieves higher accuracy than structured pruning (sparsity=4:8), which is aligned with findings in previous works (Sun et al., 2024; Frantar & Alistarh, 2023; Zhang et al., 2024). Figure 2 shows an input image and its

Figure 2: An input image (left) and its compensation (right).

Table 3: WikiText validation perplexity of pruned LLaMA family of models.

	Sparsity	LLaMA-1 (7B)	LLaMA-2 (7B)	LLaMA-3.1 (8B)
Dense	0%	5.68	5.12	5.84
Magnitude	70%	48431.68	52457.06	3483566.50
Magnitude + IC	70%	**19677.83**	**8585.07**	**33193.79**
Wanda	70%	85.02	74.42	99.72
Wanda + IC	70%	**56.47**	**67.04**	**80.12**
SparseGPT	70%	26.79	24.65	38.80
SparseGPT + IC	70%	**17.68**	**18.25**	**27.48**

compensation constructed by SparseGPT + IC when using CLIP-ViT-B/32. The compensation pool is shared across all ten tasks; thus, the additional parameters are very small (only 2.3M).

5.2 Experiments on Natural Language Processing

Models and Datasets. We evaluate IC on the LLaMA model family, i.e., LLaMA-1 (Touvron et al., 2023a), LLaMA-2 (Touvron et al., 2023b), and LLaMA-3.1 (Meta, 2024). Following (Sun et al., 2024; Frantar & Alistarh, 2023), 128 sequences sampled from the first shard of the C4 dataset (Raffel et al., 2020) are used as training data. We evaluate the pruned models on two types of tasks: (i) language modeling task which evaluates the perplexity on the held-out validation data of WikiText-2 (Merity et al., 2016); and (ii) seven zero-shot tasks include BoolQ (Clark et al., 2019), RTE (Wang, 2018), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC-easy/challenging (Clark et al., 2018), and OpenbookQA (Mihaylov et al., 2018)) from the EleutherAI LM Harness package (Gao et al., 2024).

Implementation Details. We randomly initialize $\mathbf{K}$ and $\mathbf{V}$ by a normal distribution with zero mean and standard deviation 0.01, where the rank r is set to 32. We train $\mathbf{K}$ and $\mathbf{V}$ using the AdamW optimizer (Loshchilov & Hutter, 2019) with a learning rate of 0.001 and a linear warmup scheduler over 20 epochs. The mini-batch size is set to 1, with a gradient accumulation of 2. The input embedding layer is used as the encoder of IC. As LLMs contain billions of parameters, to make pruned models more compressed, we follow Yin et al. (2024) and focus on the unstructured sparsity of 70% case.

Results on Language Modelling Task. Table 3 shows the WikiText validation perplexity. As can be seen, IC consistently brings a significant improvement to existing pruning methods, verifying the effectiveness of compensating inputs for pruned LLMs. For example, SparseGPT + IC achieves a perplexity improvement of 6.0 over SparseGPT on all three LLaMA family of models, while Wanda + IC outperforms Wanda by a large margin of 7.0 on all three LLMs. Although Magnitude performs much worse, Magnitude + IC still effectively reduces the perplexity by over 60%.

Results on Zero-shot Tasks. Table 4 shows the testing accuracy of seven zero-shot tasks on the LLaMA family of models. As can be seen, IC consistently brings a noticeable improvement (averaged over all tasks) to all existing pruning methods. For example, Wanda + IC outperforms Wanda on LLaMA-3.1-8B, LLaMA-2-7B, and LLaMA-1-7B by margins of 1.09%, 1.73%, and 0.4%, respectively, indicating that the learned compensation pool can be effectively used to construct input compensation for pruned models without any weight update. Moreover, SparseGPT + IC consistently achieves the highest accuracy for all models, showing that learning $\mathbf{\Delta_x}$ and $\mathbf{\Delta_w}$ are complementary and thus can be combined together for boosting performance.

5.3 Experiments on Image Generation

Experimental Setting. We evaluate IC on Denoising Diffusion Probability Models (DDPM) (Ho et al., 2020). Following (Fang et al., 2023), the CIFAR-10 dataset (with the image size of 32×32) (Krizhevsky & Hinton, 2009) and the off-the-shelf DDPM from (Ho et al., 2020) are used. $\mathbf{K}$ is initialized with zero and $\mathbf{V}$ is initialized randomly by a normal distribution with a standard deviation

Table 4: Testing accuracy of zero-shot tasks using LLaMA family of models.

		Sparsity	BoolQ	RTE	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	**Avg**
LLaMA-1 (7B)	Dense	0%	75.08	66.79	56.96	70.01	75.29	41.89	34.40	60.06
	Magnitude	70%	38.29	52.71	25.62	51.14	26.64	19.71	11.60	32.24
	Magnitude + IC	70%	55.99	52.35	25.33	48.38	25.93	21.93	15.00	**34.99**
	Wanda	70%	57.16	54.87	28.73	50.91	32.15	18.86	13.80	36.64
	Wanda + IC	70%	59.60	53.07	28.80	52.01	34.55	18.86	12.40	**37.04**
	SparseGPT	70%	63.43	56.32	33.89	58.96	44.07	23.63	17.80	42.58
	SparseGPT + IC	70%	66.06	54.87	37.47	60.06	48.40	25.51	18.20	**44.37**
LLaMA-2 (7B)	Dense	0%	77.71	62.82	57.16	69.14	76.30	43.43	31.40	59.71
	Magnitude	70%	37.95	53.07	25.95	49.25	27.74	22.78	16.80	33.36
	Magnitude + IC	70%	42.57	52.35	25.77	49.33	25.84	22.10	16.20	**33.45**
	Wanda	70%	46.09	52.71	27.86	51.14	30.05	18.09	11.80	33.96
	Wanda + IC	70%	58.01	52.71	27.91	50.28	29.80	19.54	11.60	**35.69**
	SparseGPT	70%	65.75	53.07	33.47	57.06	43.73	22.35	17.40	41.83
	SparseGPT + IC	70%	65.11	52.71	36.50	57.85	49.33	24.49	17.60	**43.37**
LLaMA-3.1 (8B)	Dense	0%	82.08	68.95	60.01	73.56	81.48	51.28	33.20	64.37
	Magnitude	70%	37.83	52.71	26.16	49.33	26.09	20.14	14.60	32.41
	Magnitude + IC	70%	37.83	53.79	25.71	49.88	25.21	22.78	15.20	**32.91**
	Wanda	70%	56.27	52.71	27.51	47.83	32.20	17.66	13.00	35.31
	Wanda + IC	70%	61.74	52.71	27.75	49.25	33.25	17.92	12.20	**36.40**
	SparseGPT	70%	67.71	52.71	33.60	56.20	43.14	21.08	16.40	41.55
	SparseGPT + IC	70%	67.71	54.15	34.25	57.62	46.63	22.78	15.60	**42.68**

of 0.01, where the rank r is set to 128. We train $\mathbf{K}$ and $\mathbf{V}$ using the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 0.002 over 100K steps. The mini-batch size is set to 128. The identity function is used as the encoder of IC to keep more original image information, which is crucial for image generation. Following (Fang et al., 2023), we focus on the sparsity of 30% case and compare IC with three pruning methods: Magnitude Pruning (Han et al., 2015), Taylor Pruning (Molchanov et al., 2022), and Diff-Pruning (Fang et al., 2023).

Table 5: FID of pruned DDPMs on CIFAR-10.

	Sparsity	FID
Dense	0%	4.19
Magnitude	30%	5.48
Magnitude + IC	30%	**5.31**
Taylor Pruning	30%	5.56
Taylor Pruning + IC	30%	**5.21**
Diff-Pruning	30%	5.29
Diff-Pruning + IC	30%	**5.15**

Results. Table 5 shows the Frechet Inception Distance (FID) (Heusel et al., 2017). As can be seen, IC consistently improves the existing pruning methods, demonstrating the effectiveness of compensating inputs for pruned LLMs. For instance, Taylor Pruning+IC achieves an FID improvement of 0.35 compared to Taylor Pruning. Similarly, Diff-Pruning+IC outperforms Diff-Pruning by 0.14.

6 ANALYSIS

In this section, we conduct empirical analyses to investigate the key components of IC, including rank r, sparsity, sparse retraining, and input-dependent compensation. We adopt the experimental setting used in Section 5.1 with CLIP ViT-B/32.

Sensitivity of Rank. We conduct experiments to study the sensitivity of rank r to the performance of Magnitude + IC, where r is chosen from $\{2, 4, 8, 16, 32, 64, 128\}$. Table 6 shows the testing accuracy and number of parameters of $(\mathbf{K}, \mathbf{V})$ with different ranks. As can be seen, a very small rank (e.g., 2) is not desirable. When the rank is small (≤ 16), increasing the rank leads to better performance

Table 6: Testing accuracy of Magnitude + IC (sparsity=50%) with different ranks on image classification tasks using CLIP ViT-B/32.

Rank	#Params	CIFAR100	Flowers	Food	EuroSAT	SUN	UCF	SVHN	Pets	DTD	RESISC	**Avg**
2	0.14M	70.9	51.7	69.5	94.9	47.1	58.2	93.8	57.2	38.1	83.5	66.5
4	0.29M	73.5	56.6	71.6	95.8	47.9	60.3	94.2	61.9	40.6	85.7	68.8
8	0.58M	73.9	62.0	72.2	96.4	49.4	62.7	94.3	65.8	42.7	86.4	70.6
16	1.15M	73.5	62.0	72.9	96.9	49.6	63.8	94.4	69.8	44.0	87.2	**71.4**
32	2.30M	73.0	62.9	72.4	96.5	48.9	63.1	94.4	69.2	44.1	87.1	71.2
64	4.60M	73.1	64.3	73.1	96.8	50.8	64.4	94.4	65.5	42.4	87.9	71.3
128	9.20M	73.5	56.7	72.6	96.6	49.8	62.0	94.2	62.7	40.4	87.0	69.5

while the number of parameters is still negligible (≤ 1.15M). A very large rank (e.g., 128) contains more parameters but does not contribute to better performance. In practice, we can choose the rank $\in [16, 32]$.

Sensitivity of Sparsity. We study the performance of Magnitude + IC with different sparsities. Figure 3 shows the trend of testing accuracy (averaged over ten tasks) w.r.t. sparsity (the detailed results are shown in Table 7 of Appendix A). As can be seen, when the sparsity is high ($\geq 40\%$), Magnitude + IC significantly outperforms Magnitude; When the sparsity is low ($\leq 20\%$), Magnitude + IC and Magnitude perform comparably. In practice, a high sparsity is more desirable for pruning in order to reduce the model size; Thus, IC is practically useful for enhancing pruned models.

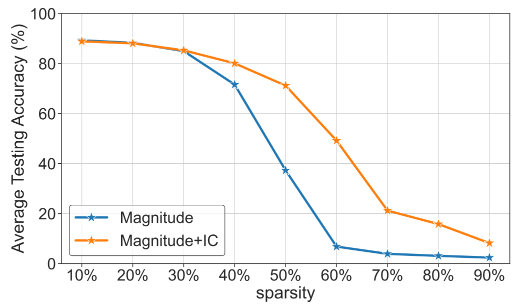
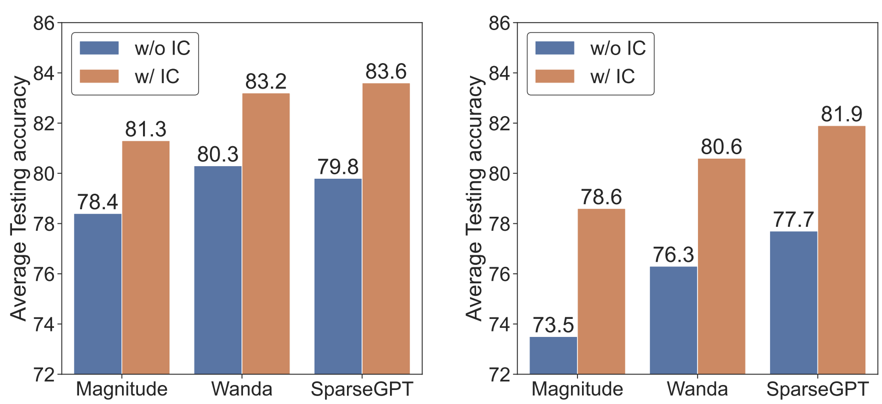


Figure 3: Performance of Magnitude and Magnitude + IC with different sparsities on image classification tasks using CLIP ViT-B/32.

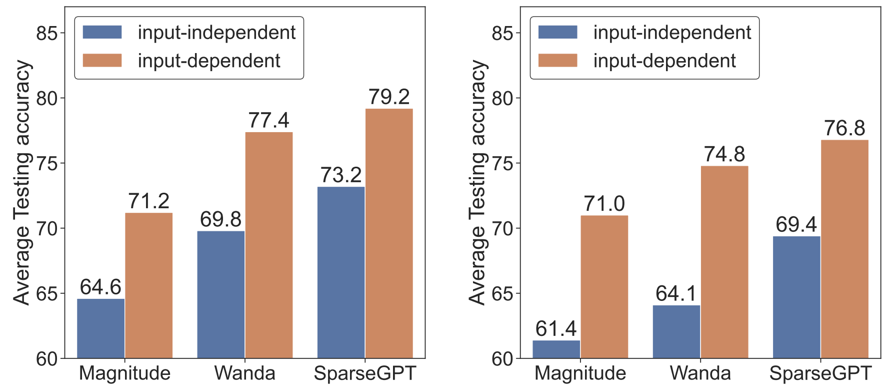
Sparse Retraining with IC. In Section 5.1, we combine IC with three pruning methods without sparse retraining. Sparse retraining, i.e., retraining the sparse model following the pruning step, can approach the performance of the dense model. We conduct experiments to investigate whether IC is beneficial to pruning methods with sparse retraining. We retrain unpruned parameters of the pruned model on the training data for 3 epochs using the AdamW optimizer with a learning rate of 0.000001 and weight decay of 0.01. Figure 4 shows the testing accuracy (averaged over ten tasks) of pruning methods w/ or w/o IC when sparse retraining is applied (detailed results are in Table 8 of Appendix A). As can be seen, IC consistently boosts existing pruning methods when sparse retraining is applied. Moreover, sparse retraining achieves higher accuracy than those without retraining (Table 1).

Input-dependent vs. Input-independent Compensation. The design of our IC ensures the compensation $\mathbf{\Delta_x}$ depends on the input. A variant of IC is learning a globally shared (i.e., input-independent) compensation $\mathbf{\Delta}$ for all inputs. For example, Visual Prompting (VP) (Bahng et al., 2022) can be



(a) Sparsity=50%. (b) Sparsity=4:8.

Figure 4: Testing accuracy (averaged over ten image classification tasks) using CLIP ViT-B/32 with sparse retraining



(a) Sparsity=50%. (b) Sparsity=4:8.

Figure 5: Testing accuracy (averaged over ten image classification tasks) of IC and an input-independent variant using CLIP ViT-B/32

used to learn the shared compensation. We conduct experiments to investigate the effectiveness of our input-dependent mechanism. Figure 5 shows the testing accuracy (averaged over ten tasks). As can be seen, IC performs much better than the input-independent variant, demonstrating that input-dependent compensation is more effective in reducing the error caused by the pruned weights.

Visualization. In Section 5.1, we learn a compensation pool with $r = 32$ to construct input compensations for ten image classification tasks, i.e., $\mathbf{\Delta_x}$ is a weighted combination of 32 candidate $\mathbf{v}_i$'s. Next, we study whether different tasks lead to different preferences for $\mathbf{v}_i$'s. Figure 6 shows the average attention weights between $\mathbf{v}_i$ and testing samples belonging to different classes of three tasks (Flowers, Food, CIFAR100) (other tasks are not shown due to limited space). As can be seen, samples from the Flowers task prefer $\{\mathbf{v}_1, \dots, \mathbf{v}_5\}$; samples from the Food task prefer $\{\mathbf{v}_6, \dots, \mathbf{v}_{10}\}$; samples from the CIFAR100 task prefer $\{\mathbf{v}_{11}, \dots, \mathbf{v}_{17}\}$.

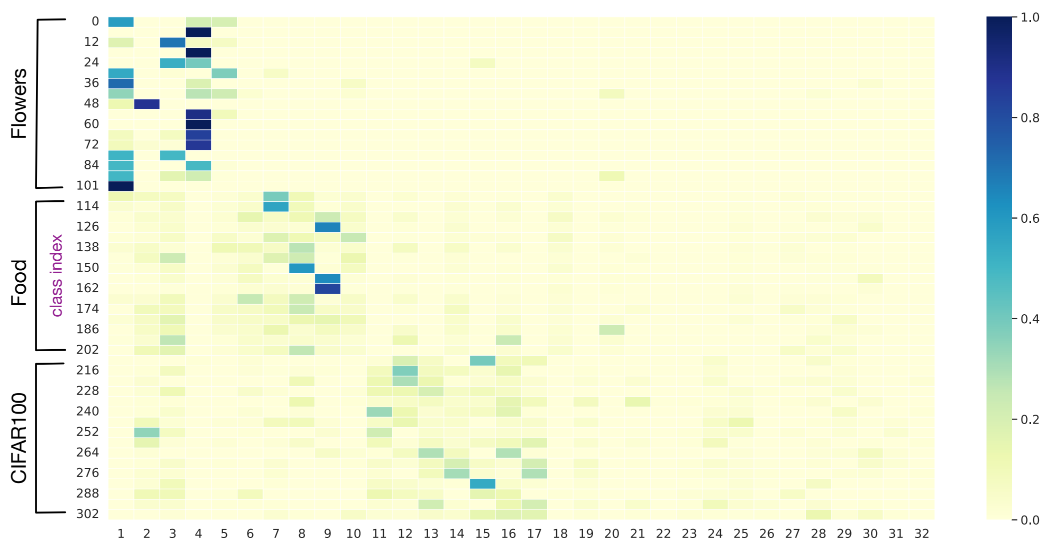


Figure 6: Distribution of attention weights on image classification tasks using CLIP ViT-B/32.

7 Conclusion

In this paper, we proposed input compensation (IC) for enhancing pruned models by adjusting the inputs to compensate for the error caused by the pruned weights. A pool of multiple candidate compensations is learned to construct input-dependent compensations by attention. IC is designed in the input space while existing pruning methods are designed in the parameter space. Hence, IC can be integrated into any existing pruning methods. Extensive experiments on NLP and CV verify that IC brings large improvements to existing pruning methods.

REFERENCES

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Fluctuation-based adaptive structured pruning for large language models. In *AAAI Conference on Artificial Intelligence*, 2024.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Neural Information Processing Systems*, 2020.

Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. Preprint arXiv:2203.17274, 2022.

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? In *Proceedings of machine learning and systems*, 2020.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Neural Information Processing Systems*, 2020.

J Campos and FL Lewis. Adaptive critic neural network for feedforward compensation. In *American Control Conference*, 1999.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. In *Proceedings of the Institute of Electrical and Electronics Engineers*, 2017.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. Preprint arXiv:1905.10044, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. Preprint arXiv:1803.05457, 2018.

Rocktim Jyoti Das, Liqun Ma, and Zhiqiang Shen. Beyond size: How gradients shape pruning decisions in large language models. Preprint arXiv:2311.04902, 2023.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. GPT3.Int8 (): 8-bit matrix multiplication for transformers at scale. In *Neural Information Processing Systems*, 2022.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Hai-Tao Zheng, and Maosong Sun. OpenPrompt: An open-source framework for prompt-learning. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

Peijie Dong, Lujun Li, Zhenheng Tang, Xiang Liu, Xinglin Pan, Qiang Wang, and Xiaowen Chu. Pruner-Zero: Evolving symbolic pruning metric from scratch for large language models. In *International Conference on Machine Learning*, 2024.

Yi Fan and Anthony Hunter. Understanding the cooking process with english recipe text. In *Findings of the Association for Computational Linguistics*, 2023.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. In *Neural Information Processing Systems*, 2023.

Gene F Franklin, J David Powell, Abbas Emami-Naeini, and J David Powell. *Feedback control of dynamic systems*. Prentice Hall Upper Saddle River, 2002.

Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. In *Neural Information Processing Systems*, 2022.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, 2023.

Elias Frantar, Eldar Kurtic, and Dan Alistarh. M-FAC: Efficient matrix-free approximations of second-order information. In *Neural Information Processing Systems*, 2021.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation. Technical report, 2024.

Ruohao Guo, Wei Xu, and Alan Ritter. Meta-tuning LLMs to leverage lexical knowledge for generalizable language style understanding. In *Annual Meeting of the Association for Computational Linguistics*, 2024.

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

Babak Hassibi, David G Stork, and Gregory J Wolff. Optimal brain surgeon and general network pruning. In *IEEE International Conference on Neural Networks*, 1993.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Neural Information Processing Systems*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Neural Information Processing Systems*, 2020.

Yutai Hou, Hongyuan Dong, Xinghao Wang, Bohan Li, and Wanxiang Che. MetaPrompting: Learning to learn better prompts. In *International Conference on Computational Linguistics*, 2022.

CV Jawahar, A Zisserman, A Vedaldi, and OM Parkhi. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

Weisen Jiang, Yu Zhang, and James Kwok. Effective structured prompting by meta-learning and representative verbalizer. In *International Conference on Machine Learning*, 2023.

Sehoon Kim, Coleman Richard Charles Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. SqueezeLLM: Dense-and-sparse quantization. In *International Conference on Machine Learning*, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Neural Information Processing Systems*, 2022.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009.

Miroslav Krstic. Input delay compensation for forward complete and strict-feedforward nonlinear systems. *IEEE Transactions on Automatic Control*, 2009.

Benjamin C Kuo and M Farid Golnaraghi. *Automatic control systems*. Prentice Hall Englewood Cliffs, NJ, 1995.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Empirical Methods in Natural Language Processing*, 2021.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning*, 2023a.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Pre-trained language models for text generation: A survey. *ACM Computing Surveys*, 2024.

Xiang Lisa Li and Percy Liang. Prefix-Tuning: Optimizing continuous prompts for generation. In *Annual Meeting of the Association for Computational Linguistics*, 2021.

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*, 2023b.

Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. Less is more: Task-aware layer-wise distillation for language model compression. In *International Conference on Machine Learning*, 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. In *Proceedings of Machine Learning and Systems*, 2024.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out*, 2022.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 2023.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. GPT understands, too. Preprint arXiv:2103.10385, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. Preprint arXiv:1609.07843, 2016.

Meta. The LLaMA 3 herd of models. Preprint arXiv:2407.21783, 2024.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. Preprint arXiv:1809.02789, 2018.

Asit Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. Accelerating sparse deep neural networks. Preprint arXiv:2104.08378, 2021.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. In *International Conference on Learning Representations*, 2022.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Neural Information Processing Systems Workshop*, 2011.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, 2024.

Antonio Polino, Razvan Pascanu, and Dan Alistarh. Model compression via distillation and quantization. In *International Conference on Learning Representations*, 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI Blog, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 2021.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. OmniQuant: Omnidirectionally calibrated quantization for large language models. In *International Conference on Learning Representations*, 2024.

Sidak Pal Singh and Dan Alistarh. WoodFisher: Efficient second-order approximation for neural network compression. In *Neural Information Processing Systems*, 2020.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. Preprint arXiv:1212.0402, 2012.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLAMA: Open and efficient foundation language models. Preprint arXiv:2302.13971, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. LLaMA 2: Open foundation and fine-tuned chat models. Preprint arXiv:2307.09288, 2023b.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017.

Alex Wang. GLUE: A multi-task benchmark and analysis platform for natural language understanding. Preprint arXiv:1804.07461, 2018.

Ji Wang, Weidong Bao, Lichao Sun, Xiaomin Zhu, Bokai Cao, and S Yu Philip. Private model compression via knowledge distillation. In *AAAI Conference on Artificial Intelligence*, 2019.

Wenxiao Wang, Wei Chen, Yicong Luo, Yongliu Long, Zhengkai Lin, Liye Zhang, Binbin Lin, Deng Cai, and Xiaofei He. Model compression and efficient inference for large language models: A survey. Preprint arXiv:2402.09748, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In *Neural Information Processing Systems*, 2022.

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*, 2023.

Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. SUN database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 2016.

Mengwei Xu, Wangsong Yin, Dongqi Cai, Rongjie Yi, Daliang Xu, Qipeng Wang, Bingyang Wu, Yihao Zhao, Chen Yang, Shihe Wang, et al. A survey of resource-efficient LLM and multimodal foundation models. Preprint arXiv:2401.08092, 2024.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In *Neural Information Processing Systems*, 2022.

Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Mykola Pechenizkiy, Yi Liang, Zhangyang Wang, and Shiwei Liu. Outlier weighed layerwise sparsity (owl): A missing secret sauce for pruning LLMs to high sparsity. In *International Conference on Machine Learning*, 2024.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. MetaMath: Bootstrap your own mathematical questions for large language models. In *International Conference on Learning Representations*, 2024.

Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics*, 2019.

Haoxing Zhang, Xiaofeng Zhang, Haibo Huang, and Lei Yu. Prompt-based meta-learning for few-shot text classification. In *Conference on Empirical Methods in Natural Language Processing*, 2022a.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. Differentiable prompt makes pre-trained language models better few-shot learners. In *International Conference on Learning Representations*, 2022b.

Tianyi Zhang, Tao Yu, Tatsunori Hashimoto, Mike Lewis, Wen-tau Yih, Daniel Fried, and Sida Wang. Coder reviewer reranking for code generation. In *International Conference on Machine Learning*, 2023.

Yingtao Zhang, Haoli Bai, Haokun Lin, Jialin Zhao, Lu Hou, and Carlo Vittorio Cannistraci. Plug-and-play: An efficient post-training pruning method for large language models. In *International Conference on Learning Representations*, 2024.

A Additional Experimental Results

Sensitivity of Sparsity. Table 7 shows the testing accuracy of Magnitude and Magnitude + IC with different sparsities. We can see that Magnitude + IC significantly outperforms Magnitude when the sparsity is high ($\geq 40\%$). In practice, a high sparsity is more desirable for pruning to reduce the model size. Hence, IC is practically useful for boosting the performance of pruned models.

Table 7: Performance of Magnitude and Magnitude + IC with different sparsities.

Sparsity	IC	CIFAR100	Flowers	Food	EuroSAT	SUN	UCF	SVHN	Pets	DTD	RESISC	**Avg**
10%	✗	88.2	97.7	89.1	98.8	73.6	86.0	97.1	91.6	74.6	96.0	**89.3**
10%	✓	87.7	97.6	88.8	98.5	73.3	85.1	97.0	91.7	73.8	96.0	88.9
20%	✗	87.5	96.5	88.4	98.6	72.9	84.1	96.9	90.4	72.8	95.4	**88.3**
20%	✓	87.2	96.3	88.1	98.5	72.6	83.2	96.9	90.4	72.2	95.3	88.1
30%	✗	84.8	88.8	84.2	97.6	68.6	79.2	96.5	88.3	67.7	94.1	85.0
30%	✓	84.9	91.0	85.3	98.3	68.5	78.9	96.7	88.3	67.0	94.2	**85.3**
40%	✗	71.1	57.0	70.0	93.5	54.6	65.8	93.8	73.2	49.7	87.7	71.6
40%	✓	79.8	81.9	80.7	97.4	60.4	72.9	96.2	81.4	58.9	91.6	**80.1**
50%	✗	33.9	26.1	34.2	45.6	30.8	35.4	45.3	38.7	27.9	55.4	37.3
50%	✓	73.0	62.9	72.4	96.5	48.9	63.1	94.4	69.2	44.1	87.1	**71.2**
60%	✗	5.6	5.2	4.3	4.3	5.5	4.7	9.1	9.0	10.0	10.2	6.8
60%	✓	57.5	25.0	52.0	89.4	26.1	37.0	85.6	29.0	16.5	74.1	**49.2**
70%	✗	1.7	2.7	2.0	13.0	0.8	2.1	7.5	2.9	2.8	3.3	3.9
70%	✓	19.3	6.1	13.8	75.4	4.0	7.0	51.6	4.1	4.0	26.6	**21.2**
80%	✗	1.0	1.0	0.8	13.0	0.3	1.4	6.5	2.8	1.7	2.4	3.1
80%	✓	6.9	5.1	5.2	68.8	1.0	3.5	38.8	3.8	2.8	22.1	**15.8**
90%	✗	1.1	0.5	1.0	6.9	0.2	0.7	6.4	2.6	2.1	2.1	2.4
90%	✓	3.5	2.7	2.2	49.5	0.5	2.1	6.7	3.5	3.4	7.5	**8.2**

Sparse Retraining with IC. We conduct experiments to study the performance of IC when sparse retraining is applied. Table 8 shows the testing accuracy on image classification tasks using CLIP ViT-B/32. As shown, IC brings a significant improvement to existing pruning methods when sparse retraining is used.

Table 8: Testing accuracy on image classification tasks using CLIP ViT-B/32 with sparse retraining.

	Sparsity	CIFAR100	Flowers	Food	EuroSAT	SUN	UCF	SVHN	Pets	DTD	RESISC	**Avg**
Magnitude	50%	79.3	80.8	81.2	87.8	61.6	73.8	95.5	78.2	55.1	90.6	78.4
Magnitude + IC	50%	82.4	82.6	83.0	98.2	63.1	76.3	96.8	80.0	57.7	92.5	**81.3**
SparseGPT	50%	80.3	85.4	83.4	83.1	63.1	74.7	96.1	82.0	58.2	91.3	79.8
SparseGPT + IC	50%	84.5	88.3	86.0	98.4	65.7	77.5	96.8	83.3	61.5	93.9	**83.6**
Wanda	50%	81.2	84.8	83.5	88.1	62.8	73.8	96.0	81.8	59.3	92.0	80.3
Wanda + IC	50%	84.6	87.4	85.3	98.4	64.7	76.6	96.8	82.6	61.5	94.0	**83.2**
Magnitude	4:8	75.9	70.4	78.7	77.0	57.1	68.1	94.9	76.1	47.9	88.9	73.5
Magnitude + IC	4:8	81.1	76.7	81.2	97.8	59.5	72.0	96.5	78.0	51.8	91.6	**78.6**
SparseGPT	4:8	79.6	80.3	82.7	80.9	60.0	70.0	95.7	81.2	55.6	90.8	77.7
SparseGPT + IC	4:8	83.8	84.0	84.8	98.3	62.2	74.3	96.7	82.7	58.7	93.7	**81.9**
Wanda	4:8	78.9	76.9	81.4	80.3	58.0	68.7	95.6	78.6	54.6	90.2	76.3
Wanda + IC	4:8	83.6	81.4	83.3	98.0	60.5	71.9	96.7	80.0	57.3	92.8	**80.6**

Statistics of the image classification datasets are shown in Table 9.

Table 9: Summary of ten image classification datasets.

Dataset	Training Size	Testing Size	#Classes
CIFAR100 (Krizhevsky & Hinton, 2009)	50,000	10,000	100
Flowers (Nilsback & Zisserman, 2008)	4,093	2,463	102
Food (Bossard et al., 2014)	50,500	30,300	101
EuroSAT (Helber et al., 2019)	13,500	8,100	10
SUN (Xiao et al., 2016)	15,888	19,850	397
DTD (Cimpoi et al., 2014)	2,820	1,692	47
UCF (Soomro et al., 2012)	7,639	3,783	101
SVHN (Netzer et al., 2011)	73,257	26,032	10
Pets (Jawahar et al., 2012)	2,944	3,669	37
RESISC (Cheng et al., 2017)	18,900	6,300	45

B ILLUSTRATION OF IC FOR LLMS

Figure 7 shows the IC for LLMs, where the pruned input embedding layer is used as the encoder. The compensation pool $(\mathbf{K}, \mathbf{V})$ is trained to construct input compensation $\mathbf{\Delta}_{\mathbf{x}}^{(i)}$ for each token's embedding $\mathbf{H}_{\mathbf{x}}^{(i)}$ via attention.

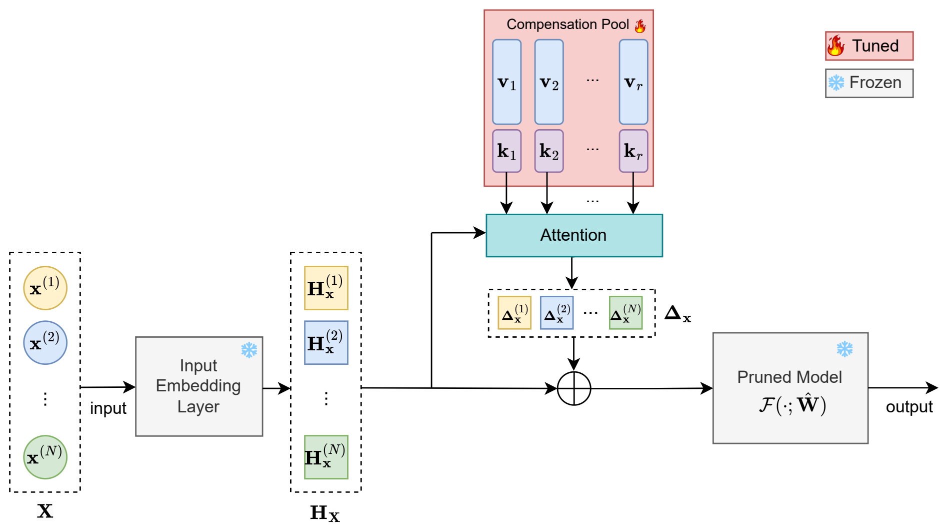


Figure 7: Input compensation for pruned LLMs.