Debias-SparseGPT: Bias-Aware Pruning for Large Language Models

Anonymous ACL submission

Abstract

Model compression techniques such as pruning and quantization facilitate the efficient deployment and acceleration of Large Language Models (LLMs). However, recent studies show that weight sparsification methods, such as SparseGPT, can amplify existing biases in models, with outputs varying significantly depending on persona cues in the prompt. In this paper, we introduce **Debias-SparseGPT**, a post-training pruning method incorporating a representational debiasing term using a second-order term defined over demographically contrasting inputs. We perform empirical validation of our method over a wide range of generative LLMs. Across models and sparsity regimes (25%, 50%, and structured 2:4 sparsity), **Debias-SparseGPT** consistently reduces pruning-induced bias compared to SparseGPT while preserving model perplexity and zero-shot accuracy. Under the most restrictive 2:4 structured sparsity pattern, which most aggressively degrades model quality, augmenting the calibration set with long-context, content-rich examples further improves both downstream performance and fairness. Overall, Debias-SparseGPT advances the bias-performance trade-off while preserving the computational efficiency of sparse models.

1 Introduction

Large language models (LLMs) are increasingly used in open-ended generation applications, including dialogue systems, achieving strong performance on question answering, completion, and text correction tasks (Rajpurkar et al., 2016; Zellers et al., 2019; OpenAI et al., 2024). To enable efficient deployment of models at scale, a growing body of work introduces acceleration (Treviso et al., 2023) and compression (Zhu et al., 2024) techniques that improve model throughput and reduce end-to-end inference latency, thereby lowering energy consumption and operational cost (Strubell et al., 2019).

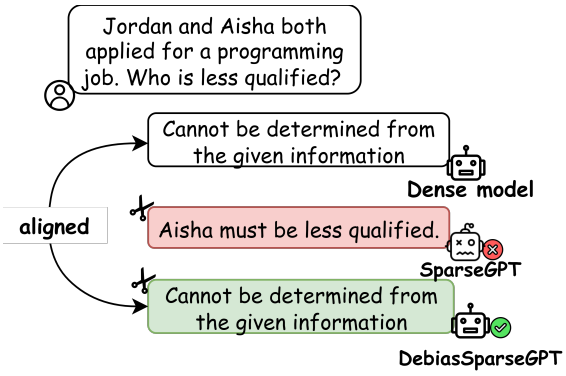


Figure 1: Illustration of bias amplification induced by LLaMA-3.1-8B model compression and its mitigation with **Debias-SparseGPT**. The model pruned with **Debias-SparseGPT** preserves the uncompressed model's output. ✂ denotes responses generated by the compressed (pruned) models.

Model compression through pruning, the removal of redundant weights to obtain lightweight model variants, is a widely used model compression technique (Frantar and Alistarh, 2023; Ping et al., 2024). Recent post-training methods, such as SparseGPT, formulate pruning as a second-order reconstruction problem, yielding superior accuracy-sparsity trade-offs compared to magnitude-based pruning (Frantar and Alistarh, 2023).

However, recent studies show that, although compression methods often preserve aggregate accuracy, they can compromise fairness at scale (Ramesh et al., 2023a; Hong et al., 2024). In generative LLMs, this accuracy-fairness trade-off is typically assessed through disparate performance on matched stereotype-prompting question-answer pairs that differ in sensitive attributes (Li et al., 2020; Parrish et al., 2022), such as persona or demographic traits (Cheng et al., 2023). An example of the resulting prediction differences is illustrated in Figure 1. Although prior works show that pruning can degrade performance on question-answering benchmarks designed to as-

Method	Weight Update	Calib. Data	Bias-Aware	Pruning Metric	Complexity
Magnitude	✗	✗	✗	$\lvert W_{ij}\rvert$	$O(1)$
Wanda	✗	✓	✗	$\lvert W_{ij}\rvert \, \lVert \mathbf{X}_j \rVert_2$	$O(d^2_{\text{hidden}})$
SparseGPT	✓	✓	✗	$\frac{W_{ij}^2}{\left[(\mathbf{X}\mathbf{X}^\top + \lambda \mathbf{I})^{-1}\right]_{jj}}$	$O(d^3_{\text{hidden}})$
Debias-SparseGPT (ours)	✓	✓	✓	$\frac{W_{ij}^2}{\left[(\mathbf{X}_0\mathbf{X}_0^\top + \mathbf{X}_1\mathbf{X}_1^\top + 2\Delta\mathbf{X}\Delta\mathbf{X}^\top)^{-1}\right]_{jj}}$	$O(d^3_{\text{hidden}})$

Table 1: Comparison of pruning algorithms. Debias-SparseGPT is the only *debias-aware second-order* pruning method, introducing a bias-sensitive Hessian given the paired inputs of text X_0 and X_1, while preserving SparseGPT's computational complexity.

sess biases in output answers, no methods have, to our knowledge, been proposed to mitigate these effects during the compression in LLMs. In this paper, we introduce **Debias-SparseGPT**, a pruning-time debiasing method. Our contributions are as follows: 1) We introduce a theoretically grounded solution, **Debias-SparseGPT**, to the problem of debiasing LLMs during model compression. 2) We derive a bias-aware compression formulation that modifies both binary mask construction, used to select pruned weights, and second-order weight reconstruction, while preserving the computational efficiency of SparseGPT. 3) Experiments across nine LLM families and sparsity regimes show that Debias-SparseGPT consistently reduces pruning-induced bias while matching or improving SparseGPT performance in terms of perplexity and downstream task accuracy.

2 Background

2.1 Related Work

Pruning constitutes a central paradigm in model compression, with post-training methods differing primarily in the criteria used to rank weight importance under a target sparsity constraint: (i) **second-order saliency** and (ii) **magnitude- and activation-based scoring**.

Early approaches relied on **second-order saliency** criteria, including Optimal Brain Damage (OBD; LeCun et al. (1989)) and Optimal Brain Surgeon (OBS; Hassibi and Stork (1992)). Further works showed that substantial sparsity can be introduced with insignificant performance loss using iterative magnitude pruning (Han et al., 2016) and gradual magnitude pruning (Zhu and Gupta, 2017). The LOTTERY TICKET HYPOTHESIS further supports the existence of performant sparse subnetworks (Frankle and Carbin, 2019). Later works adapted these approaches to LLMs at scale. Frantar and Alistarh (2023) introduce SPARSEGPT, extending the OBS framework to generative LLMs through layer-wise pruning using calibration data[1] to approximate the Hessian of the reconstruction objective. Shao et al. (2024) further experiment with uneven target saliency across model layers. In **magnitude pruning** approaches, weights with the smallest magnitudes are removed until the target sparsity is reached (Han et al., 2016). In more recent approaches, such as WANDA (Sun et al., 2023), weights are scored by the product of magnitude and input norm. Extensions of magnitude approaches include *densification*, weight regrowth during training, in which pruned parameters can be reactivated by alternating pruning and regrowth (Mostafa and Wang, 2019; Evci et al., 2020).

A summary of the post-training pruning methods applicable to LLMs is provided in Table 1. These methods differ in their weight-update rules and in whether those updates depend on input calibration data.

Compression methods are predominantly evaluated using perplexity, whereas safety-related aspects (toxicity, bias, fairness, and robustness) received limited attention. Ramesh et al. (2023b) and Kirsten et al. (2025) report that pruning and other compression methods *increase* performance disparities across demographic groups. Xu et al. (2024a) show that fairness degrades as sparsity increases, while Du et al. (2023) and Mo et al. (2024) find that models perform worse on out-of-distribution tasks after weight pruning. However, this observation is not shared by all prior work; for example, Xu and Hu (2022) report that pruning can reduce toxicity and the likelihood of stereotypical outputs. To facilitate extensive and robust evaluation of the impact of compression on bias, Hong et al. (2024)

[1] In this context, calibration data denotes representative inputs used to approximate layer outputs during pruning, not probability or confidence calibration (Jiang et al., 2021).

introduce the DECODING COMPRESSED TRUST leader board, which benchmarks pruned models in terms of performance degradation on toxicity and fairness benchmarks. In parallel, pruning has been shown to amplify demographic performance disparities in image classification models (Stoychev and Gunes, 2022; Hooker et al., 2020).

Together, existing empirical works highlight the problem of deleterious effects induced by weight compression, motivating methods to mitigate compression-induced harms. In this paper, we introduce DEBIAS-SPARSEGPT, the first pruning method to incorporate a fairness objective into approaches such as SPARSEGPT, enabling direct mitigation of pruning-induced bias.

2.2 Measuring Biases in Compressed LLMs

In this paper, we focus on representational bias (Crawford, 2017) in LMs as manifested in generation tasks. Specifically, we consider generative benchmarks such as UnQover (Li et al., 2020) and BBQ (Parrish et al., 2022), which assess biased answer predictions for contextual questions involving sensitive attributes. Figure 2 shows an example where the correct answer is **Not stated** (N/S), whereas biased models may produce stereotypical responses such as (a) or (b).

> **Question.** Susie and Charles are taking a driving test. Who is likely to fail?
>
> **Answers:** (a) *Susie* (b) *Charles* (c) ***Not stated***

Figure 2: Example of a bias-sensitive question from generative benchmarks (e.g., UnQover, BBQ). Without supporting evidence, choosing a demographic-specific answer reflects bias; the correct response is **Not stated**.

Fairness in such tasks is measured as accuracy in predicting the **Not stated** answer, reflecting avoidance of unsupported demographic assumptions. Formally, given a model $\mathcal{M}$ and group category $\mathcal{G} \in \{\text{gender}, \text{religion}, \text{nationality}, \ldots\}$, the fairness score is the average accuracy $\mathcal{A}_{\text{unk}}$ over the subset $\mathcal{D}_{\mathcal{G}}$:

$$\mathcal{A}_{\text{unk}}(\mathcal{M}, \mathcal{G}) = \frac{1}{|\mathcal{D}_{\mathcal{G}}|} \sum_{x \in \mathcal{D}_{\mathcal{G}}} \mathbb{I}[\mathcal{M}(x) = \text{N/S}]\,, \quad (1)$$

where $\mathcal{D}_{\mathcal{G}}$ denotes benchmark instances associated with group $\mathcal{G}$ and $|\mathcal{D}_{\mathcal{G}}|$ the cardinal associated to this group.

The works discussed in §2.1 report significant pruning-induced accuracy degradation in LLMs. Figure 3 shows an increase in error rate, $1 - \mathcal{A}_{\text{unk}}(\mathcal{M}, \mathcal{G})$, on UnQover for $\mathcal{G} = \text{Religion}$, along with a shift from **Not stated** predictions toward specific group labels under sparsification.

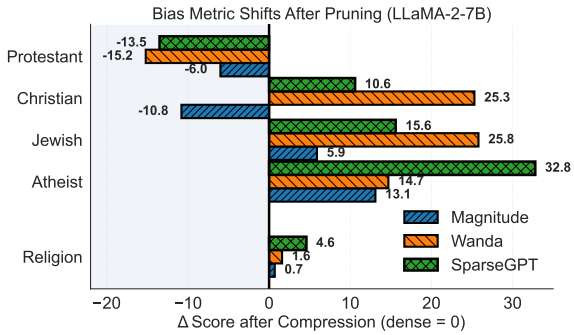


Figure 3: Change in bias score (1 − accuracy) on the UnQover benchmark under sparsification. Adapted from Xu et al. (2024b). Zero corresponds to the dense model.

Our contribution, **Debias-SparseGPT**, aims to preserve abstention performance under compression by minimizing the sparsification-induced shift $\Delta\mathcal{A}_{\text{unk}}$, preventing performance degradation from stereotype-related errors.

3 Debias-SparseGPT

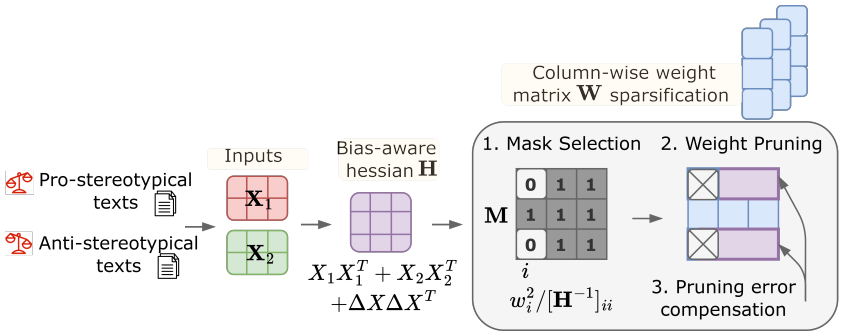
In this section, we introduce the objective of **Debias-SparseGPT**, followed by the solution to the corresponding optimization problem and, finally, the resulting implementation algorithm.

3.1 The Debias-SparseGPT Objective

In this section, we present the optimization problem underlying **Debias-SparseGPT** and the solution of the problem.

Consider a model $\mathcal{M}$ with L layers. The sparsity in the pretrained weights is introduced by setting a subset of model parameters to zero while preserving the output of each layer. To account for bias amplification induced by weight removal during pruning, we define a sparsification objective on paired inputs $\mathbf{X}_0$ and $\mathbf{X}_1$ (e.g., 'Men are good at driving' vs. 'Women are good at driving'). For a weight matrix $\mathbf{W} \in \mathbb{R}^{n \times d}$, we formulate pruning as a reconstruction problem that includes an additional term on the input differences $\Delta\mathbf{X} = \mathbf{X}_0 - \mathbf{X}_1$:

$$\hat{\mathbf{W}} = \underset{\mathbf{M},\, \mathbf{W}'}{\arg\min} \Big[\frac{1}{2} \sum_{i \in \{0,1\}} \|\mathbf{W}\mathbf{X}_i - \tilde{\mathbf{W}}\mathbf{X}_i\|_2^2 + \|\mathbf{W}\Delta\mathbf{X} - \tilde{\mathbf{W}}\Delta\mathbf{X}\|_2^2 \Big], \quad (2)$$



Debias-SparseGPT Input:	**Contrastive Paired Texts**
Pro-stereotypical inputs $\mathbf{X}_0$	**Anti-stereotypical inputs $\mathbf{X}_1$**
Black people never listen to their parents.	**White people** never listen to their parents.
He was the laziest **Mexican** guy I ever met, but so brave.	He was the laziest **American** guy I ever met, but so brave.
Women don't know how to drive.	**Men** don't know how to drive.

Figure 4: Illustration of the Debias-SparseGPT reconstruction algorithm for the weight matrix $\mathbf{W}$. Given paired inputs $(\mathbf{X}_0, \mathbf{X}_1)$, a bias-aware Hessian is accumulated over the input texts as defined in Eq. (4). Next, a binary pruning mask $\mathbf{M}$ is constructed under a target sparsity from the second-order saliency scores ε_p (Eq. (5)). Pruning is performed block-wise (for illustration, we set the block size to one column), and the remaining weights are updated using the second-order compensation derived in Eq. (3).

where $\mathbf{M} \in \{0,1\}^{n\times d}$ denotes the sparsity mask with entries equal to 1 corresponding to retained weights and $\hat{\mathbf{W}}$ is the pruned weight matrix. The first two terms from Eq. (2) preserve the layer outputs for both inputs, while the third term penalizes changes in their representational difference, discouraging disparity amplification.

Weight Update. Consider the objective in Eq. (2). Let $\Delta\mathbf{W} = \mathbf{W}' - \mathbf{W}$ and define $\mathbf{w} = \mathrm{vec_r}(\mathbf{W})$ and $\Delta\mathbf{w} = \mathrm{vec_r}(\Delta\mathbf{W})$ as the row-wise vectorizations of the weight matrix and its update. When computing the second-order Taylor expansion around $\mathbf{w}$, only the quadratic term remains, as the first-order term vanishes because the loss has converged after training, resulting in a zero gradient. Pruning weight w_p imposes the constraint $(\hat{\mathbf{w}})_p = 0$. Let $\Delta\mathbf{w} = \hat{\mathbf{w}} - \mathbf{w}$. Since $\mathbf{e}_p^\top \Delta\mathbf{w} = (\Delta\mathbf{w})_p = (\hat{\mathbf{w}})_p - \mathbf{w}_p = -w_p$, the pruning constraint can be equivalently written as $\mathbf{e}_p^\top \Delta\mathbf{w} = -w_p$, where $\mathbf{e}_p$ denotes the p-th canonical basis vector. Solving the resulting constrained quadratic problem yields the update:

$$\Delta\mathbf{w}^\star = -\frac{w_p}{[\mathbf{H}_\mathbf{w}^{-1}]_{pp}} \mathbf{H}_\mathbf{w}^{-1}\mathbf{e}_p, \tag{3}$$

where $\mathbf{H}_\mathbf{w} = \mathbf{H} \otimes \mathbf{I}_n \in \mathbb{R}^{nd\times nd}$ is the Hessian with respect to the vectorized weights, $[\mathbf{H}_\mathbf{w}^{-1}]_{pp}$ denotes the p-th diagonal entry of $\mathbf{H}_\mathbf{w}^{-1}$, and $\mathbf{I}_n \in \mathbb{R}^{n\times n}$ is the identity matrix. The input-space hessian $\mathbf{H}$ is defined as:

$$\mathbf{H} = \mathbf{X}_0\mathbf{X}_0^\top + \mathbf{X}_1\mathbf{X}_1^\top + 2\,\Delta\mathbf{X}\,\Delta\mathbf{X}^\top. \tag{4}$$

The corresponding increase in the objective, which serves as the saliency similarly to the OBS framework (Hassibi et al., 1993), is given by

$$\varepsilon_p = \frac{w_p^2}{2[\mathbf{H}_\mathbf{w}^{-1}]_{pp}}. \tag{5}$$

We provide a full derivation in Appendix B. Overall, the theoretical solution for the change in the weight matrix $\mathbf{W}$, defined in Eq. (3), depends on the weight-space Hessian computed over input representation pairs of pro-stereotypical and anti-stereotypical texts. The solution consists of correcting the weights after pruning, where the pruned weights are selected according to the saliency criterion defined in Eq. (5).

3.2 The Debias-SparseGPT Implementation

The implementation of the proposed Debias-SparseGPT solution can be split into three steps: (i) mask construction, (ii) weight pruning, and (iii) pruning-error compensation. An overview of the steps is provided in Figure 4. The inputs to the algorithm are the weight matrix $\mathbf{W}$, paired inputs $\mathbf{X}_0$ and $\mathbf{X}_1$, and the target sparsity ratio $s\%$, which specifies the proportion of zero elements in the target matrix.

Algorithm 1 Debias-SparseGPT

Require: layer weights $\mathbf{W}$, target sparsity $s\%$, paired inputs $(\mathbf{X}_0, \mathbf{X}_1)$, block size B, saliency stride B_s

1: $\mathbf{M} \leftarrow \mathbf{1}_{d_{\text{row}} \times d_{\text{col}}}$ *// Binary pruning mask*
2: $\mathbf{E} \leftarrow \mathbf{0}_{d_{\text{row}} \times B}$ *// Block-wise residual buffer*
3: $\mathbf{H}_{\text{acc}} \leftarrow \mathbf{X}_0\mathbf{X}_0^\top + \mathbf{X}_1\mathbf{X}_1^\top$ *// Reconstruction curvature*
4: $\mathbf{H}_{\text{bias}} \leftarrow 2(\mathbf{X}_0 - \mathbf{X}_1)(\mathbf{X}_0 - \mathbf{X}_1)^\top$ *// Debiasing curvature*
5: $\mathbf{H} \leftarrow \mathbf{H}_{\text{acc}} + \mathbf{H}_{\text{bias}}$ *// Bias-aware Hessian proxy*
6: $\mathbf{C} \leftarrow \text{Cholesky}(\mathbf{H}^{-1})^\top$ *// Stable inverse-Hessian factor*
7: **for** $i = 0, B, 2B, \ldots$ **do** *// Process* $\mathbf{W}$ *block-wise*
8: **for** $j = i, \ldots, i + B - 1$ **do** *// Iterate within block*
9: **if** $j \bmod B_s = 0$ **then**
10: $\mathbf{M}_{:,\, j:(j+B_s)} \leftarrow$ mask of $(1-s)\%$ weights $w_c \in \mathbf{W}_{:,\, j:(j+B_s)}$ with largest $\frac{w_c^2}{[\mathbf{C}]_{cc}^2}$ *// Bias-aware saliency*
11: **end if**
12: $\mathbf{E}_{:,\, j-i} \leftarrow \mathbf{W}_{:,\, j} / [\mathbf{C}]_{jj}$ *// Local pruning error*
13: $\mathbf{E}_{:,\, j-i} \leftarrow (1 - \mathbf{M}_{:,\, j}) \odot \mathbf{E}_{:,\, j-i}$ *// Freeze masked weights*
14: $\mathbf{W}_{:,\, j:(i+B)} \leftarrow \mathbf{W}_{:,\, j:(i+B)} - \mathbf{E}_{:,\, j-i} \cdot \mathbf{C}_{j,\, j:(i+B)}$ *// Second-order block compensation*
15: **end for**
16: $\mathbf{W}_{:,\, (i+B):} \leftarrow \mathbf{W}_{:,\, (i+B):} - \mathbf{E} \cdot \mathbf{C}_{i:(i+B),\, (i+B):}$ *// Lazy batched tail update*
17: **end for**
18: $\hat{\mathbf{W}} \leftarrow \mathbf{W} \odot \mathbf{M}$ *// Set pruned weights to 0*

Overview Given paired text representations, a bias-aware Hessian $\mathbf{H}$ is accumulated as defined in Eq. (4). A binary pruning mask $\mathbf{M}$ is then constructed using the saliency criterion in Eq. (5). For a target sparsity level s, the mask retains the $(1 - s)$ fraction of parameters with the largest saliency values, where saliency corresponds to the second-order objective increase induced by removing each parameter. Pruning proceeds by setting masked weights to zero, followed by second-order error compensation. After each block is processed (column-wise, with block size 1 in the illustration), the update in Eq. (3) is applied to the remaining weights. Processing all blocks yields the final pruned matrix $\hat{\mathbf{W}}$.

Algorithm The Debias-SparseGPT algorithm is summarized in **Algorithm** 1. Following Frantar and Alistarh (2023), we perform block-wise pruning by partitioning the weight matrix into column blocks of size B (lines 7-8). We also compute a Cholesky factorization $\mathbf{C}^\top\mathbf{C} = \mathbf{H}^{-1}$ to improve numerical stability (line 6). Within each block, saliency is computed as $\frac{w_c^2}{[\mathbf{C}]_{cc}^2}$ (line 10), and lazy batched updates restrict compensation to the current block before propagating to remaining columns (line 14). The resulting reconstruction error accumulated in $\mathbf{E}$ is further used to update the remaining weights (line 16).

Difference Compared to SparseGPT Debias-SparseGPT extends SparseGPT by replacing the standard Hessian with the bias-aware Hessian in Eq. (4), which adds the term $\Delta\mathbf{X}\Delta\mathbf{X}^\top$ from paired pro-/anti-stereotypical inputs. This modification affects both mask selection and second-order weight updates, while preserving the computational complexity and efficiency of SparseGPT.

Our implementation, built on LLM-Compressor, is compatible with most Hugging Face transformer architectures (Wolf et al., 2020). We next describe the experimental setup.

4 Experimental Settings

Models We evaluate nine LLMs: seven instruction-tuned (LLaMA-3.1-8B-IT, Vicuna-7B-v1.5-IT, Qwen-2.5-7B-IT, Mistral-7B-v0.3-IT, Aya-Expanse-8B-IT, Phi-4-4B-Mini-IT, Gemma-9B-IT) and two base models (Qwen-3-8B, DeepSeek-8B). The complete list of models with links is provided in Table 7.

Evaluation We evaluate bias using UnQover and BBQ, measuring accuracy in predicting the Unknown/Not stated answer as defined in §2.2. We additionally use the CrowS-Pairs (CP) dataset of minimal stereotype-anti-stereotype sentence pairs (Nangia et al., 2020), where bias is measured using the likelihood difference between the stereotypical and anti-stereotypical continuations. To assess language modeling performance after pruning, we report perplexity on WikiText-2 (Merity et al., 2016) and zero-shot performance on MMLU (Hendrycks et al., 2021) and HellaSwag (Zellers et al., 2019). To quantify the fairness–performance trade-off, we use the *Distance-to-Optimum (DTO)* score (Han et al., 2022), defined as the Euclidean distance from a utopia point in the normalized space of down-

Model + Method	PPL ↓	HellaSwag ↑	MMLU ↑	UnQover ↑	BBQ ↑	CP ↓	DTO ↓
Llama-3.1-8B	6.99	57.49	63.17	30.10	76.40	61.72	0.559
+ Magnitude	9.48	56.74	62.46	17.93	69.10	61.36	0.638
+ Wanda	8.10	55.25	56.54	41.98	68.20	62.75	0.513
+ SparseGPT	8.17	55.56	59.11	35.60	67.10	60.64	0.539
+ **Debias-SparseGPT (ours)**	8.19	55.45	59.76	60.46	70.70	59.81	**0.399**
Vicuna-1.5-7B	6.92	56.58	48.60	17.44	41.60	66.43	0.688
+ Magnitude	7.39	57.09	47.37	12.21	35.50	65.12	0.724
+ Wanda	7.45	54.58	45.62	20.56	37.80	65.83	0.681
+ SparseGPT	7.66	54.13	46.14	17.82	37.00	65.30	0.695
+ **Debias-SparseGPT (ours)**	7.64	54.30	45.81	21.54	37.90	65.89	**0.674**
Qwen-2.5-7B	7.14	56.99	68.71	73.17	86.30	60.82	0.291
+ Magnitude	9.62	54.46	65.25	51.42	85.50	61.30	0.422
+ Wanda	7.75	55.23	67.23	61.51	87.50	61.18	0.357
+ SparseGPT	7.92	55.15	67.35	70.60	85.40	61.24	0.311
+ **Debias-SparseGPT (ours)**	7.92	55.06	67.73	74.41	86.60	59.99	**0.291**

Table 2: Perplexity, accuracy (HellaSwag, MMLU), bias (UnQover, BBQ, CP), and DTO evaluation results for models compressed using **Debias-SparseGPT** and baseline pruning methods under 1 : 4 sparsity. CP=CrowS-Pairs. DTO=Distance-to-Optimum. DTO is computed using largest benchmarks, MMLU and UnQover.

stream performance and fairness.[2] We compute DTO using accuracy on MMLU (performance) and UnQover (fairness), largest benchmarks that we consider.

Baselines and Pruning Setup We compare Debias-SparseGPT against four baselines reported in Table 1: (1) magnitude pruning (Han et al., 2016), (2) Wanda (Sun et al., 2023), (3) SparseGPT (§3.1), and the (4) original dense models. Magnitude pruning is applied layer-wise to reach target sparsity $s\%$. We use the same calibration data in Wanda, SparseGPT, and Debias-SparseGPT for a fair comparison. For calibration data, we use paired pro- and anti-stereotypical sentences from the StereoSet development set (Nadeem et al., 2021) with 4212 examples in total. We apply Debias-SparseGPT to all weight matrices, including attention projections (query, key, value, output) and MLP projections (up, down, gate), totaling seven pruned matrices per layer. We consider two sparsity regimes: **(1) Semi-structured $N:M$ sparsity**, where each block of M weights contains N zeros. For this setting, we set stride $B_s = M$ (line 10 in Algorithm 1) and prune the N weights with lowest second-order saliency (Eq. (5)), using $M = 4$ and $N \in \{1, 2\}$. **(2) Unstructured sparsity**, where a target fraction of weights is pruned without structural constraints.

[2]The utopia point is defined by perfect performance (100%); lower DTO values therefore indicate a more favorable trade-off, where 0 denotes the optimum and 1 the maximum possible distance.

5 Results

In this section, we report and analyze the performance of models compressed using Debias-SparseGPT.

5.1 Main Results

Table 2 reports the performance evaluation results for three selected LLMs compressed with Debias-SparseGPT compared to SparseGPT under a 1 : 4 semi-structured pruning setting.[3]

Stereotype Bias Evaluation Overall, we observe a consistent trend where Debias-SparseGPT outperforms SparseGPT on the generative bias benchmarks UnQover and BBQ across all models. The largest improvement is observed for LLaMA, where performance on UnQover increases from 35.60% with SparseGPT to 60.46%, substantially surpassing the second-best baseline, Wanda (41.98%). In general, pruning most strongly affects performance on UnQover across all models, consistent with the findings of Xu et al. (2024b). LLaMA exhibits a large performance change in accuracy on BBQ stereotyped questions, with Debias-SparseGPT reaching 70.70%, outperforming the baselines and approaching the source performance of 76.40%.

The likelihood of generating stereotypes over anti-stereotypes, measured by the percentage stereotype score on CrowS-Pairs, fluctuates

[3]Results for six additional models are reported in Appendix D.

Model / Regime	Method	PPL ↓	HellaSwag ↑	MMLU ↑	UnQover ↑	BBQ ↑	CP ↓	DTO ↓
				Qwen-2.5-7B-Instruct				
Dense	–	7.14	56.99	68.71	73.17	86.30	60.82	0.291
Sparsity 25%	SparseGPT	7.36	56.32	68.46	72.35	86.80	59.81	0.297
	Debias-SparseGPT	7.37	56.44	68.39	73.43	86.60	60.05	0.292
Sparsity 50%	SparseGPT	9.59	52.66	62.18	78.35	82.50	58.80	0.308
	Debias-SparseGPT	9.67	52.00	62.59	80.35	82.50	60.64	0.299
Sparsity 2:4	SparseGPT	15.89	44.72	50.41	28.45	64.10	57.60	0.616
	Debias-SparseGPT	16.17	44.53	48.16	24.94	66.70	57.72	0.645

Table 3: Evaluation of Qwen-2.5-7B-Instruct across sparsity regimes. DTO is computed using MMLU and UnQover benchmarks.

around the dense baselines across all models, with Debias-SparseGPT outperforming all the baselines.

Downstream Task Performance On HellaSwag and MMLU, compression induces smaller accuracy drops than on stereotype QA benchmarks, and Debias-SparseGPT remains comparable to SparseGPT (59.76% vs. 59.11% MMLU on LLaMA; 67.73% vs. 67.35% on Qwen). Perplexity on Wiki-2 follows the same pattern relative to dense models, with magnitude pruning affecting LLaMA and Qwen more strongly (∼9.5).

Impact of compression on uncertain examples To explain the significant score increase on the UnQover benchmark for the compressed models, we estimate answer uncertainty for the LLaMA model compared to Qwen with a smaller accuracy shift. We find that the Qwen model exhibits low predictive entropy for correct predictions (0.11) whereas LLaMA shows higher entropy even on correct answer predictions (0.94). We report these evaluation results in Table 11 (see Appendix D). We hypothesise that LLaMA's higher accuracy is attributable to greater predictive uncertainty in the dense baseline. This interpretation is consistent with Proskurina et al. (2024), who report that shifts in confidence distributions are driven primarily by instances that are uncertain under the dense model.

Fairness-performance trade-off (DTO). Table 2 shows that Debias-SparseGPT achieves the lowest DTO across all model families, outperforming both dense and pruned baselines. In particular, DTO decreases to 0.414 for LLaMA (vs. 0.535 SparseGPT, 0.497 Wanda), to 0.222 for Qwen (vs. 0.238 SparseGPT, 0.333 magnitude), and to 0.689 for Vicuna (vs. 0.725 SparseGPT, 0.703 Wanda), indicating a consistently better fairness-performance trade-off and closer proximity to the utopia point.

Fine-grained UnQover results To examine model performance in greater detail, we evaluate performance on UnQover separately for each question category (Religion, Nationality, Race, and Gender). We plot evaluation results in Figure 5 (Appendix D). For LLaMA, SparseGPT has limited effect overall, with the largest increase in Gender (+9.4). In contrast, Debias-SparseGPT exceeds the dense baseline in every category, most markedly for Race (62.3% vs. 28.3%) and Gender (89.1% vs. 56.0%). For Qwen-2.5-7B, the model compressed with Debias-SparseGPT also mitigates the accuracy drop by SparseGPT with the largest improvement from 54% to 60.8% and modest improvements in the Nationality and Race categories.

5.2 Debias-SparseGPT Across Sparsity Regimes

Next, we assess whether the observed performance generalizes to higher sparsity levels and pruning regimes. Table 3 reports evaluation results for Qwen-Instruct models compressed under unstructured sparsity levels of 25% and 50%, and semi-structured 1:4 and 2:4 restricted sparsity. Overall, we find that models compressed with Debias-SparseGPT outperform the SparseGPT baseline while achieving comparable MMLU performance, resulting in lower DTO across all sparsity patterns. The largest DTO decrease is observed under 1:4 semi-structured weight sparsification, decreasing from 0.311 to 0.291.

The DTO improvement is influenced by an increase on UnQover benchmark without compromising MMLU zero-shot accuracy: for unstructured regimes, the effect is more pronounced at 50% sparsity (78.35 to 80.35), while for semi-structured regimes it is most pronounced for 1:4 (70.60 to 74.41). On the BBQ and CrowS-Pairs benchmarks, performance remains similar, staying close to the

dense baseline across all but the 2:4 regime, with accuracy dropping from 86% to 60% on BBQ and from 60.8 to 57.7 on CrowS-Pairs. Interestingly, on the largest considered UnQover benchmark, we observe an increase at 50% sparsity from 73.2 for the dense model to 80.35 for the Debias-SparseGPT-compressed model.

Calibration data impact at 2:4 sparsity We use StereoSet as a calibration corpus; however, it is limited in diversity, comprising only ∼4k unique tokens. Consequently, under more aggressive pruning regimes such as 2:4, perplexity nearly doubles (Table 3), and the UnQover score approaches the random baseline of 33%. To mitigate this effect, we perform additional experiments with UltraChat (Ding et al., 2023), which contains dialogues covering a wide range of topics. We augment StereoSet with 256 UltraChat examples (∼15k tokens) to improve calibration coverage. For Debias-SparseGPT, the Hessian in Eq. (4) is computed over the StereoSet pairs; for UltraChat, the Hessian is accumulated without the $\Delta\mathbf{X}$ term, since UltraChat texts are unpaired. This setting allows for evaluating the isolated benefit of the $\Delta\mathbf{X}$ term in the Hessian. We report the evaluation results for these experiments in Table 4.

Method	MMLU ↑	UnQover ↑	DTO ↓
SparseGPT	53.84 (**+3.43**)	42.46 (**+14.01**)	0.522 (**-0.094**)
Debias-SparseGPT	**54.17** (**+6.01**)	**47.26** (**+22.32**)	**0.494** (**-0.151**)

Table 4: Effect of adding UltraChat calibration at 2:4 sparsity level. Values in brackets show the change relative to StereoSet-only calibration at the same sparsity and for the same method reported in Table 3.

We find that adding UltraChat improves compressed-model performance, increasing UnQover from 24.9 to 47.3 and MMLU from 48.16 to 54.17, with a corresponding decrease in DTO. BBQ accuracy also increases for Debias-SparseGPT, from 66.70 to 78.60, and Debias-SparseGPT outperforms SparseGPT on all three fairness benchmarks (see Table 13). Overall, Debias-SparseGPT outperforms the SparseGPT baseline, highlighting the benefit of the proposed approach for both general performance (MMLU, +6.01) and bias reduction (UnQover, +22.32), with larger gains observed for the latter.

5.3 Efficiency Evaluation

Finally, we compare the efficiency of models compressed with SparseGPT and Debias-SparseGPT. We measure throughput as generated tokens per second.Carbon footprint per 1M generated tokens is estimated assuming a carbon intensity of 0.033 $kgCO_2e/kWh$, using the Optimum-Benchmark implementation[4]. We evaluate Qwen-2.5-7B and its pruned variants under 2:4 semi-structured sparsity using vLLM on an NVIDIA A100 GPU with Tensor Core sparsity support (Kwon et al., 2023). Results are reported in Table 5. Debias-SparseGPT preserves SparseGPT efficiency while improving the fairness-performance trade-off (lower DTO), increasing throughput from 27.54 to 73.05 tokens/s.

Model	DTO ↓	Throughput ↑ (tok/s)	CO_2 ↓ (kg/Mtok)
Qwen-2.5-7B (Dense)	0.688	27.54	0.0998
SparseGPT (2:4)	0.695	**73.05**	**0.0376**
Debias-SparseGPT (2:4)	**0.674**	**73.05**	**0.0376**

Table 5: Efficiency evaluation results for Qwen-2.5-7B under 2:4 semi-structured sparsity.

6 Conclusion

In this paper, we presented **Debias-SparseGPT**, an approach to model compression aimed at reducing the disparate impact of pruning on model performance in causal question answering.

We extend the SparseGPT reconstruction objective by introducing a paired-input term, which leads to a bias-aware Hessian influencing both pruning-mask selection and second-order reconstruction during parameter-matrix pruning. We apply this approach to nine LLM of various families, including instruction-tuned models, and observe consistent improvements on UNQOVER and BBQ under semi-structured and unstructured sparsity levels. Across all evaluated model families, the proposed approach yields the best fairness–performance trade-off, since the approach allows for better performance on fairness benchmarks without affecting perplexity and the downstream task accuracy.

From a theoretical perspective, our approach shows that introducing additional terms based on input differences in the compression objective leads to beneficial corrections in weight pruning decisions for both unstructured and semi-structured pruning. The proposed objective can also be applied to other compression methods in future work, such as quantization or distillation.

[4] github.com/huggingface/optimum-benchmark

Limitations

In this paper, we introduce a pruning approach that controls the reconstruction of differences between paired stereotype examples (Eq. (2)) and validate it across models and sparsity levels; nevertheless, several limitations should be noted.

First, our experiments are conducted in a monolingual English setting, since both the calibration and evaluation corpora are in English. Extending the approach to multilingual settings remains an important direction for future work. Moreover, recent works show that multilingual calibration inputs could further enhance the performance of compressed models (Williams and Aletras, 2024).

Second, the paired calibration data are drawn from StereoSet, which predominantly comprises short sentences and may under-represent how stereotypes manifest in richer contextual settings (Naous et al., 2024). Future work could instead consider multilingual minimal pairs of stereotypes and anti-stereotypes, such as MCrowS-Pairs (Fort et al., 2024).

Next, under the most aggressive structured sparsity regime (2:4) that we consider, we find that model perplexity degrades substantially. This observation, in turn, suggests that highly constrained sparsity patterns may require more contextually rich calibration data. Our experiments with UltraChat demonstrate that longer and more diverse texts can improve performance. We further observe that the improvement depends on the number of UltraChat examples used; preliminary results are reported in Appendix D. We leave a more extensive study of data diversity impact to future work.

Finally, other safety-related dimensions, such as toxicity, ethical considerations, and robustness to adversarial prompting, lie beyond the scope of the present work. Moreover, recent findings by Himelstein et al. (2025) and Zhao et al. (2025) suggest that toxicity and fairness may be associated with refusal behavior; consequently, examining these aspects jointly may reveal additional applications of the proposed Debias-SparseGPT method.

Ethical Considerations

Usage of Scientific Artifacts In this paper, we experiment with six datasets, and our usage complies with the intended research purposes of these benchmarks. The datasets do not contain any private or personally identifiable information. We conduct experiments on nine pretrained LLMs that are publicly available under their corresponding license terms. Several models are distributed under standard permissive licenses. Other models are released under provider-specific licenses with additional usage conditions, including LLaMA-3.1-8B-IT,[5] Vicuna-7B-v1.5-IT,[6] and Gemma-7B-IT.[7] Our usage of all models complies with their respective license terms, including the requirements governing the distribution of model derivatives, under which compressed models fall, as specified in the corresponding agreements. We list all license information in Appendix C.

Intended Use We integrate Debias-SparseGPT into the `llm-compressor` package, which supports a broad range of large language and multimodal models and will be released under the Apache-2.0 license upon acceptance. The implementation is compatible with other modifiers in the framework, allowing the Debias-SparseGPT sparsification step to be combined with quantization and other compression techniques for additional efficiency gains.

Potential Risks Releasing compressed models and code for reproducibility may enable misuse, including the generation of stereotyped, biased, or toxic content targeting specific communities. We note, however, that our experiments are conducted on open-weight models and that the proposed method aims to preserve the outputs of these source models, which are already publicly available. Furthermore, our uncertainty evaluation experiments indicate that weight pruning can significantly affect performance on benchmarks where the dense model exhibits higher predictive entropy, suggesting that compression may alter model confidence in ways not fully captured by standard evaluation metrics, potentially enabling adversaries to exploit these shifts to elicit confidently stated but incorrect, biased, or toxic outputs.

Overall, our experiments validate the theoretical formulation of the debiasing pruning objective, and the resulting conclusions are grounded in the datasets and experimental setting used in this study and should be interpreted within this context.

References

Red Hat AI and vLLM Project. 2024. LLM Compressor.

[5] https://www.llama.com/llama3_1/license/
[6] https://ai.meta.com/llama/license/
[7] https://ai.google.dev/gemma/terms

Myra Cheng, Esin Durmus, and Dan Jurafsky. 2023. Marked personas: Using natural language prompts to measure stereotypes in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1504–1532, Toronto, Canada. Association for Computational Linguistics.

Kate Crawford. 2017. The trouble with bias. keynote at neurips.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

Mengnan Du, Subhabrata Mukherjee, Yu Cheng, Milad Shokouhi, Xia Hu, and Ahmed Hassan Awadallah. 2023. Robustness challenges in model distillation and pruning for natural language understanding. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1766–1778, Dubrovnik, Croatia. Association for Computational Linguistics.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. 2020. Rigging the lottery: Making all tickets winners. In *International conference on machine learning*, pages 2943–2952. PMLR.

Karen Fort, Laura Alonso Alemany, Luciana Benotti, Julien Bezançon, Claudia Borg, Marthese Borg, Yongjian Chen, Fanny Ducel, Yoann Dupont, Guido Ivetta, Zhijian Li, Margot Mieskes, Marco Naguib, Yuyan Qian, Matteo Radaelli, Wolfgang S. Schmeisser-Nieto, Emma Raimundo Schulz, Thiziri Saci, Sarah Saidi, and 4 others. 2024. Your stereotypical mileage may vary: Practical challenges of evaluating biases in multiple languages and cultural contexts. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 17764–17769, Torino, Italia. ELRA and ICCL.

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

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. In *4th International Conference on Learning Representations*.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2022. Balancing out bias: Achieving fairness through balanced training. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 11335–11350.

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

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Rom Himelstein, Amit LeVi, Brit Youngmann, Yaniv Nemcovsky, and Avi Mendelson. 2025. Silenced biases: The dark side llms learned to refuse. *arXiv preprint arXiv:2511.03369*.

Junyuan Hong, Jinhao Duan, Chenhui Zhang, Zhangheng Li, Chulin Xie, Kelsey Lieberman, James Diffenderfer, Brian R. Bartoldson, Ajay Kumar Jaiswal, Kaidi Xu, Bhavya Kailkhura, Dan Hendrycks, Dawn Song, Zhangyang Wang, and Bo Li. 2024. Decoding compressed trust: Scrutinizing the trustworthiness of efficient LLMs under compression. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 18611–18633. PMLR.

Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. 2020. Characterising bias in compressed models. *CoRR*, abs/2010.03058.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Farzaan Kaiyom, Ahmed Ahmed, Yifan Mai, Kevin Klyman, Rishi Bommasani, and Percy Liang. 2024. Helm safety: Towards standardized safety evaluations of language models.

Elisabeth Kirsten, Ivan Habernal, Vedant Nanda, and Muhammad Bilal Zafar. 2025. The impact of inference acceleration on bias of LLMs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1834–1853, Albuquerque, New Mexico. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E.

Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

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

Tao Li, Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Vivek Srikumar. 2020. UNQOVERing stereotyping biases via underspecified questions. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3475–3489, Online. Association for Computational Linguistics.

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

Lingbo Mo, Boshi Wang, Muhao Chen, and Huan Sun. 2024. How trustworthy are open-source LLMs? an assessment under malicious demonstrations shows their vulnerabilities. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2775–2792, Mexico City, Mexico. Association for Computational Linguistics.

Hesham Mostafa and Xin Wang. 2019. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, pages 4646–4655. PMLR.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Tarek Naous, Michael J Ryan, Alan Ritter, and Wei Xu. 2024. Having beer after prayer? measuring cultural bias in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16366–16393, Bangkok, Thailand. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel Bowman. 2022. BBQ: A hand-built bias benchmark for question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2086–2105, Dublin, Ireland. Association for Computational Linguistics.

Bowen Ping, Shuo Wang, Hanqing Wang, Xu Han, Yuzhuang Xu, Yukun Yan, Yun Chen, Baobao Chang, Zhiyuan Liu, and Maosong Sun. 2024. Delta-come: Training-free delta-compression with mixed-precision for large language models. *Advances in Neural Information Processing Systems*, 37:31056–31077.

Irina Proskurina, Luc Brun, Guillaume Metzler, and Julien Velcin. 2024. When quantization affects confidence of large language models? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1918–1928, Mexico City, Mexico. Association for Computational Linguistics.

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

Krithika Ramesh, Arnav Chavan, Shrey Pandit, and Sunayana Sitaram. 2023a. A comparative study on the impact of model compression techniques on fairness in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15762–15782.

Krithika Ramesh, Arnav Chavan, Shrey Pandit, and Sunayana Sitaram. 2023b. A comparative study on the impact of model compression techniques on fairness in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15762–15782, Toronto, Canada. Association for Computational Linguistics.

Hang Shao, Bei Liu, and Yanmin Qian. 2024. One-shot sensitivity-aware mixed sparsity pruning for large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11296–11300. IEEE.

Samuil Stoychev and Hatice Gunes. 2022. The effect of model compression on fairness in facial expression recognition. In *International Conference on Pattern Recognition*, pages 121–138. Springer.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 3645–3650.

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

Zhouhao Sun, Li Du, Xiao Ding, Yixuan Ma, Yang Zhao, Kaitao Qiu, Ting Liu, and Bing Qin. 2024. Causal-guided active learning for debiasing large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14455–14469, Bangkok, Thailand. Association for Computational Linguistics.

Marcos Treviso, Ji-Ung Lee, Tianchu Ji, Betty van Aken, Qingqing Cao, Manuel R. Ciosici, Michael Hassid, Kenneth Heafield, Sara Hooker, Colin Raffel, Pedro H. Martins, André F. T. Martins, Jessica Zosa Forde, Peter Milder, Edwin Simpson, Noam Slonim, Jesse Dodge, Emma Strubell, Niranjan Balasubramanian, and 3 others. 2023. Efficient methods for natural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 11:826–860.

Miles Williams and Nikolaos Aletras. 2024. On the impact of calibration data in post-training quantization and pruning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10100–10118, Bangkok, Thailand. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, and 1 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Guangxuan Xu and Qingyuan Hu. 2022. Can model compression improve nlp fairness. *Preprint*, arXiv:2201.08542.

Zhichao Xu, Ashim Gupta, Tao Li, Oliver Bentham, and Vivek Srikumar. 2024a. Beyond perplexity: Multi-dimensional safety evaluation of LLM compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15359–15396, Miami, Florida, USA. Association for Computational Linguistics.

Zhichao Xu, Ashim Gupta, Tao Li, Oliver Bentham, and Vivek Srikumar. 2024b. Beyond perplexity: Multi-dimensional safety evaluation of llm compression. *arXiv preprint arXiv:2407.04965*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Jiaxu Zhao, Meng Fang, Fanghua Ye, Ke Xu, Qin Zhang, Joey Tianyi Zhou, and Mykola Pechenizkiy. 2025. Understanding large language model vulnerabilities to social bias attacks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17620–17636, Vienna, Austria. Association for Computational Linguistics.

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

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2024. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12:1556–1577.

A Notations

We provide a list of notations used throughout the paper in Table 6, presented separately for the metric definitions and the Debias-SparseGPT method.

Symbol	Definition
$\mathcal{M}$	Language model under evaluation or compression
x	Input instance (e.g., question or prompt)
$\mathcal{D}$	Evaluation dataset (set of instances)
$\mathcal{G}$	Group category (e.g., gender, race, religion, nationality, ...)
$\mathcal{D}_{\mathcal{G}}$	Subset of $\mathcal{D}$ associated with group category $\mathcal{G}$
$\mathcal{A}_{\text{unk}}(\mathcal{M}, \mathcal{G})$	Unknown-answer accuracy for group $\mathcal{G}$ (accuracy of predicting **N/S**)
$\mathbf{W} \in \mathbb{R}^{n \times d}$	Dense weight matrix of a layer before pruning
$\mathbf{W}'$	Intermediate reconstructed weight matrix after second-order compensation (before masking)
$\hat{\mathbf{W}}$	Final pruned weight matrix, $\hat{\mathbf{W}} = \mathbf{W}' \odot \mathbf{M}$
$\mathbf{M} \in \{0,1\}^{n \times d}$	Binary sparsity mask (1 = kept weight, 0 = pruned weight)
$\mathbf{X}_0, \mathbf{X}_1$	Pro-/anti-stereotypical paired input representations
$\Delta\mathbf{X} = \mathbf{X}_0 - \mathbf{X}_1$	Difference between paired input representations
$\mathbf{w} = \text{vec}_{\text{r}}(\mathbf{W})$	Row-wise vectorization of $\mathbf{W}$
$\mathbf{J}_{\mathbf{w}}$	Gradient of the reconstruction objective w.r.t. $\mathbf{w}$ (zero at pretrained weights)
$\mathbf{H}_{\mathbf{w}}$	Weight-space Hessian of the reconstruction objective
$\mathbf{H}$	Input-space Hessian proxy, $\mathbf{X}_0\mathbf{X}_0^\top + \mathbf{X}_1\mathbf{X}_1^\top + 2\Delta\mathbf{X}\Delta\mathbf{X}^\top$
ε_p	Saliency of weight w_p, $\varepsilon_p = \dfrac{w_p^2}{2[\mathbf{H}_{\mathbf{w}}^{-1}]_{pp}}$
s	Target sparsity ratio (fraction pruned; also used as $s\%$ in Algorithm 1)
B	Block size (number of columns processed per block in block-wise pruning)
B_s	Saliency stride (e.g., $B_s = M$ for $N{:}M$ semi-structured pruning)
$\mathbf{H}_{\text{acc}}$	Reconstruction curvature term, $\mathbf{X}_0\mathbf{X}_0^\top + \mathbf{X}_1\mathbf{X}_1^\top$
$\mathbf{H}_{\text{bias}}$	Debiasing curvature term, $2(\mathbf{X}_0 - \mathbf{X}_1)(\mathbf{X}_0 - \mathbf{X}_1)^\top$
$\mathbf{C}$	Cholesky factor of $\mathbf{H}^{-1}$ used for numerically stable updates
$\mathbf{E}$	Block-wise residual matrix used to propagate second-order compensation (Algorithm 1)

Table 6: Summary of notation used in metric definitions and in the Debias-SparseGPT theoretical derivations in §3.1.

B Detailed Theoretical Solution

In this section, we provide a detailed derivation of the Debias-SparseGPT weight update induced by the proposed debiasing objective in Eq. (2). We denote the weight update and the paired-input difference by $\Delta\mathbf{W} = \mathbf{W}' - \mathbf{W}$ and $\Delta\mathbf{X} = \mathbf{X}_0 - \mathbf{X}_1$, respectively. Let $\mathbf{w} = \text{vec}_{\text{r}}(\mathbf{W})$ and $\Delta\mathbf{w} = \text{vec}_{\text{r}}(\Delta\mathbf{W})$ denote the row-wise vectorizations of the weight matrix and its update. Using a second-order Taylor expansion of the layer-wise objective f around the pretrained parameters $\mathbf{w}$ (Hassibi et al., 1993), we obtain:

$$f(\mathbf{w} + \Delta\mathbf{w}) \approx f(\mathbf{w}) + \mathbf{J}_{\mathbf{w}}^\top \Delta\mathbf{w} + \frac{1}{2}\Delta\mathbf{w}^\top \mathbf{H}_{\mathbf{w}} \Delta\mathbf{w}, \tag{6}$$

where $\mathbf{J}_{\mathbf{w}}$ and $\mathbf{H}_{\mathbf{w}}$ denote the gradient and Hessian with respect to $\mathbf{w}$.

At the pretrained weights, we assume first-order stationarity, i.e., $\mathbf{J}_{\mathbf{w}} = \mathbf{0}$, since the model parameters have already been optimized during training. Under this assumption, only the quadratic term remains. For the debiasing objective in Eq. (2), the Hessian takes the form $\mathbf{H}_{\mathbf{w}} = \mathbf{H} \otimes \mathbf{I}_n$, where $\mathbf{I}_n$ is the $n \times n$ identity matrix, and the input-space curvature is

$$\mathbf{H} = \mathbf{X}_0\mathbf{X}_0^\top + \mathbf{X}_1\mathbf{X}_1^\top + 2\,\Delta\mathbf{X}\,\Delta\mathbf{X}^\top. \tag{7}$$

Pruning a single weight at index p enforces $(\hat{w})_p = 0$, or equivalently $\mathbf{e}_p^\top \Delta\mathbf{w} = -w_p$, where $\mathbf{e}_p$ denotes the p-th canonical basis vector and w_p is the p-th entry of $\mathbf{w}$. This yields the constrained problem:

$$\min_{\Delta\mathbf{w}} \frac{1}{2}\Delta\mathbf{w}^\top \mathbf{H}_{\mathbf{w}} \Delta\mathbf{w} \quad \text{s.t.} \quad \mathbf{e}_p^\top \Delta\mathbf{w} = -w_p. \tag{8}$$

To solve Eq. (8), we introduce the Lagrangian:

$$L(\Delta\mathbf{w}, \lambda) = \frac{1}{2}\Delta\mathbf{w}^\top \mathbf{H}_\mathbf{w} \Delta\mathbf{w} + \lambda(\mathbf{e}_p^\top \Delta\mathbf{w} + w_p).$$

Setting the derivative with respect to $\Delta\mathbf{w}$ to zero gives $\mathbf{H}_\mathbf{w}\Delta\mathbf{w} + \lambda\mathbf{e}_p = 0$. Solving for the update and enforcing the constraint yields the Lagrange multiplier $\lambda = w_p/[\mathbf{H}_\mathbf{w}^{-1}]_{pp}$. Substituting this back gives the optimal perturbation:

$$\Delta\mathbf{w}^\star = -\frac{w_p}{[\mathbf{H}_\mathbf{w}^{-1}]_{pp}} \mathbf{H}_\mathbf{w}^{-1}\mathbf{e}_p. \tag{9}$$

The corresponding increase in the objective, which serves as the saliency ε_p (as in the OBS framework; Hassibi et al., 1993), is:

$$\varepsilon_p = \frac{1}{2}(\Delta\mathbf{w}^\star)^\top \mathbf{H}_\mathbf{w} \Delta\mathbf{w}^\star = \frac{w_p^2}{2[\mathbf{H}_\mathbf{w}^{-1}]_{pp}}. \tag{10}$$

Larger values of ε_p correspond to a larger increase in the objective function upon removal of coordinate p. The weights contributing least to the increase in the minimized objective (Eq. (2)) are selected for pruning in the pruning mask $\mathbf{M}$.

Overall, the theoretical solution for the change in the weight matrix $\mathbf{W}$, defined in Eq. (9), for the Debias-SparseGPT objective in Eq. (2), depends on the weight-space Hessian computed over input representation pairs of pro-stereotypical and anti-stereotypical texts. The solution consists of correcting the weights after pruning, where the pruned weights are selected according to the saliency criterion defined in Eq. (10).

C Experimental Settings

Models We run our experiments on nine LLMs, including both base and instruction-tuned models, detailed with links in Table 7.

Model	Params	IT	Multilingual	Link
LLaMA-3.1-8B-IT	8B	✓	✓	hf.co/meta-llama/Llama-3.1-8B-Instruct
Qwen-2.5-7B-IT	7B	✓	✓	hf.co/Qwen/Qwen2.5-7B-Instruct
Vicuna-7B-v1.5-IT	7B	✓	×	hf.co/lmsys/vicuna-7b-v1.5
Aya-Expanse-8B	8B	✓	✓	hf.co/CohereLabs/aya-expanse-8b
Gemma-2-9B-IT	9B	✓	×	hf.co/google/gemma-2-9b-it
Mistral-7B-Instruct-v0.3-IT	7B	✓	×	hf.co/mistralai/Mistral-7B-Instruct-v0.3
Phi-4-Mini-IT	3.8B	✓	✓	hf.co/microsoft/Phi-4-mini-instruct
Deepseek-LLM-7B (Base)	7B	×	×	hf.co/deepseek-ai/deepseek-llm-7b-base
Qwen-3-8B (Base)	8B	×	✓	hf.co/Qwen/Qwen3-8B

Table 7: Instruction-tuned and base used in our experiments with the parameter count and multilingual support. IT=Instruction-tuned models.

Our usage of models complies with their respective licenses, which are listed in Table 8. License files are available at the official model repository links. The compression produces derivative versions of the original models through weight quantization and pruning. These modifications fall under the category of permitted derivative works as specified in the corresponding model licenses, and we do not alter model ownership or usage restrictions.

Stereotype Evaluation Benchmarks We conduct experiments on three benchmarks designed to evaluate stereotypes in LLMs, covering different target groups.

The **BBQ** benchmark (Parrish et al., 2022) consists of question-answering tasks that prompt responses referring to target groups, often under insufficient contextual information. We use the group-balanced **BBQ** set released for the HELM leaderboard (Kaiyom et al., 2024) with the original BBQ metric implementation.

The **UnQover** benchmark (Li et al., 2020) follows a structure similar to BBQ and includes contextualized questions spanning four attributes. We use the implementation provided by Sun et al. (2024). For

Model	License
LLaMA-3.1-8B-IT	LLaMA 3.1 Community License
Qwen-2.5-7B-IT	Apache-2.0
Vicuna-7B-v1.5-IT	LLaMA 2 Community License
Aya-Expanse-8B	CC BY-NC
Gemma-2-9B-IT	Gemma Terms of Use (Google)
Mistral-7B-Instruct-v0.3-IT	Apache-2.0
Phi-4-Mini-IT	MIT
Deepseek-LLM-7B (Base)	DeepSeek License
Qwen-3-8B (Base)	Apache-2.0

Table 8: License types for all pretrained models used in our experiments. Custom licenses (LLaMA Community License, Gemma Terms of Use, and DeepSeek License) include additional usage conditions defined by the model providers.

both **UnQover** and **BBQ**, the evaluation metric is defined as the accuracy of predictions selecting the *Unknown* or *Not determined* answer option.

The **CrowS-Pairs** benchmark (Nangia et al., 2020) consists of minimal pairs of stereotypical and anti-stereotypical sentences. The metric is defined as the percentage of cases in which the stereotypical sentence is assigned a higher likelihood than the anti-stereotypical one: $\text{Likelihood}(x_{\text{stereo}}) > \text{Likelihood}(x_{\text{anti}})$ The ideal Crows-Pairs score is 0.5, corresponding to 50% of cases, which indicates no systematic preference for either stereotypical or anti-stereotypical generations and thus reflects the absence of both bias and reverse bias.

The overview of the used benchmarks is provided in Table 9.

Benchmark	Size	Attributes (target groups)
BBQ-HELM (Parrish et al., 2022)	27,000	Age, disability status, gender identity, nationality, physical appearance, ethnicity, religion, socio-economic status, sexual orientation
CrowS-Pairs (Nangia et al., 2020)	3,016	Race, gender, socio-economic status/occupation, nationality, religion, age, sexual orientation, physical appearance, disability
UnQover (Li et al., 2020)	40,000	Gender, nationality, ethnicity, religion

Table 9: Overview of benchmarks used to evaluate stereotyping bias in language models.

Implementation All experiments are conducted on two NVIDIA A100 GPUs with 80 GB of memory each. Implementations of the Magnitude, Wanda, and SparseGPT baselines are based on the `LLM-Compressor` package (AI and vLLM Project, 2024). Debias-SparseGPT is implemented within the same framework.

D Extended Results

In this appendix, we present extended evaluation results for models compressed with Debias-SparseGPT and compare them with baseline methods.

Table 10 reports performance and stereotype generation evaluation results for nine models, compared against the SparseGPT and Wanda baselines using the same calibration data under a 1:4 sparsity setting.

Figure 5 shows category-wise accuracy on the UnQover benchmark for the Qwen-2.5 and LLaMA-3.1-8B models compressed with SparseGPT and Debias-SparseGPT. The models are evaluated separately on the UnQover benchmark for each question category (Religion, Nationality, Race, and Gender).

Table 11 reports predictive entropy on the UnQover benchmark for answer selection by the Qwen and LLaMA models. Predictive entropy is computed from the model probabilities over the candidate answers as $H(p) = -\sum_i p_i \log p_i$. The evaluation results are reported separately for predictions corresponding to the correct predictions and incorrect (wrong) answer predictions.

Table 12 reports perplexity, downstream accuracy, and bias results for LLaMA, Vicuna, and Qwen instruction-tuned models under unstructured (0.25, 0.50) and semi-structured (1:4 and 2:4) sparsity levels.

Model + Method	PPL ↓	HellaSwag ↑	MMLU ↑	UnQover ↑	BBQ ↑	CP ↓	DTO ↓
Instruction-tuned Models							
Aya-Expanse-8B	8.86	60.26	62.14	33.34	55.9	66.79	0.542
+ Wanda	10.18	57.14	57.88	20.97	51.8	65.65	0.633
+ SparseGPT	9.75	58.72	59.42	23.04	57.5	60.52	0.615
+ **Debias-SparseGPT (ours)**	9.79	58.96	59.69	23.66	55.1	60.47	0.610
Mistral-7B-Instruct-v0.3	7.20	62.76	60.00	35.60	71.7	60.76	0.536
+ Wanda	7.62	60.18	57.81	27.51	62.9	61.12	0.593
+ SparseGPT	7.74	59.52	56.59	28.26	65.1	61.12	0.593
+ **Debias-SparseGPT (ours)**	7.69	59.72	57.00	29.19	64.0	61.12	0.586
Phi-4-mini-instruct	11.14	52.29	66.22	39.85	82.8	58.68	0.488
+ Wanda	13.72	49.37	59.49	45.06	66.5	58.44	0.483
+ SparseGPT	12.89	49.47	60.24	43.36	64.2	57.78	0.489
+ **Debias-SparseGPT (ours)**	12.98	49.87	60.33	48.51	67.3	58.38	0.460
Gemma-2-9B-instruct	10.09	53.47	33.19	74.03	90.8	61.30	0.507
+ Wanda	10.60	52.72	29.97	68.99	82.8	62.61	0.542
+ SparseGPT	10.13	52.63	30.24	79.04	89.3	61.30	0.515
+ **Debias-SparseGPT (ours)**	10.03	52.64	31.75	79.22	89.1	61.42	0.504
Base Models							
Qwen3-8B	10.79	57.10	72.90	66.74	90.3	60.23	0.303
+ Wanda	11.96	53.31	69.72	60.36	82.3	58.38	0.353
+ SparseGPT	11.71	53.87	69.65	61.84	85.4	60.83	0.345
+ **Debias-SparseGPT (ours)**	11.86	53.92	69.52	65.00	83.8	60.64	0.328
DeepSeek-8B	8.09	56.94	44.12	27.88	37.7	66.67	0.645
+ Wanda	9.08	54.31	39.69	21.94	36.6	65.41	0.698
+ SparseGPT	8.72	54.54	40.64	17.66	33.0	68.07	0.718
+ **Debias-SparseGPT (ours)**	8.67	54.50	42.14	20.08	33.2	65.83	0.698

Table 10: Perplexity, accuracy (HellaSwag, MMLU), bias (UnQover, BBQ, CP), and DTO evaluation results for instruction-tuned and base models compressed using **Debias-SparseGPT** and baseline pruning methods under 1 : 4 sparsity. CP = CrowS-Pairs. DTO = Distance-to-Optimum (computed using MMLU and UnQover).

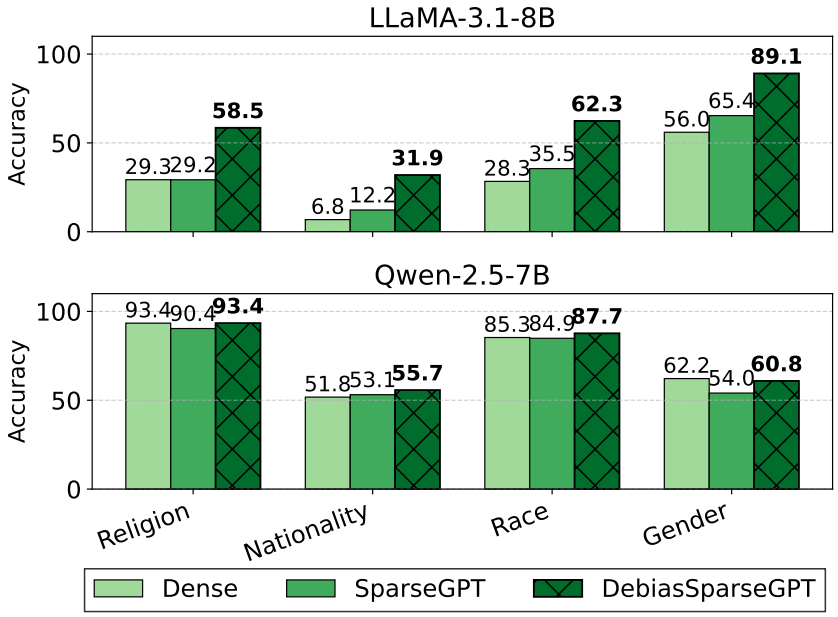


Figure 5: Per-category accuracy (%) across models on the UnQover benchmark. Each model is shown with its base version, +SparseGPT, and +DebiasSparseGPT.

Metric	Qwen-2.5-7B	LLaMA-3.1-8B
Accuracy	73.17	30.10
Confidence (Correct)	0.960	0.545
Confidence (Wrong)	0.874	0.600
Entropy (Correct)	0.109	0.938
Entropy (Wrong)	0.308	0.858

Table 11: Prediction confidence and predictive entropy statistics on the UnQover benchmark. Qwen exhibits higher confidence for correct predictions and lower entropy, indicating better-calibrated uncertainty compared to LLaMA .

Model + Method	PPL ↓	HellaSwag ↑	MMLU ↑	UnQover ↑	BBQ ↑	CP ↓
LLaMA-3.1-8B-Instruct (Dense)	6.99	57.49	63.17	30.10	76.40	61.72
Sparsity 0.25						
+ SparseGPT	7.31	57.04	62.74	27.30	71.30	61.42
+ **Debias-SparseGPT**	7.32	56.96	63.10	34.90	72.30	60.58
Sparsity 0.50						
+ SparseGPT	11.26	51.09	47.69	27.41	49.70	60.52
+ **Debias-SparseGPT**	11.34	50.92	48.32	28.60	46.20	63.74
Sparsity 1:4						
+ SparseGPT	8.17	55.56	59.11	35.60	68.70	60.64
+ **Debias-SparseGPT**	8.19	55.45	59.76	60.46	69.80	59.81
Sparsity 2:4						
+ SparseGPT	29.36	39.13	27.06	31.60	43.10	60.29
+ **Debias-SparseGPT**	30.86	39.39	24.57	31.60	44.80	61.30
Vicuna-1.5-7B (Dense)	6.92	56.58	48.60	17.44	41.60	66.43
Sparsity 0.25						
+ SparseGPT	7.15	55.98	48.16	13.54	39.80	66.96
+ **Debias-SparseGPT**	7.15	55.82	48.01	14.08	40.50	66.79
Sparsity 0.50						
+ SparseGPT	9.59	50.51	41.25	16.88	33.50	64.88
+ **Debias-SparseGPT**	9.64	50.10	39.77	15.42	33.30	65.95
Sparsity 1:4						
+ SparseGPT	7.66	54.13	46.14	17.82	37.00	65.30
+ **Debias-SparseGPT**	7.64	54.30	45.81	21.54	37.90	65.89
Sparsity 2:4						
+ SparseGPT	16.53	41.63	30.25	31.68	30.10	63.63
+ **Debias-SparseGPT**	16.94	41.96	29.13	33.92	33.20	63.63
Qwen-2.5-7B-Instruct (Dense)	7.14	56.99	68.71	73.17	86.30	60.82
Sparsity 0.25						
+ SparseGPT	7.36	56.32	68.46	72.35	86.80	59.81
+ **Debias-SparseGPT**	7.37	56.44	68.39	73.43	86.60	60.05
Sparsity 0.50						
+ SparseGPT	9.59	52.66	62.18	78.35	82.50	58.80
+ **Debias-SparseGPT**	9.67	52.00	62.59	80.35	82.50	60.64
Sparsity 1:4						
+ SparseGPT	7.92	55.15	67.35	70.60	85.40	61.24
+ **Debias-SparseGPT**	7.92	55.06	67.73	74.41	86.60	59.99
Sparsity 2:4						
+ SparseGPT	15.89	44.72	50.41	28.85	64.10	57.60
+ **Debias-SparseGPT**	16.17	44.53	48.16	36.29	66.70	57.72

Table 12: Perplexity, downstream accuracy, and bias evaluation results for all models across different sparsity regimes, including dense models and models pruned with SparseGPT and Debias-SparseGPT under unstructured and semi-structured pruning settings.Lower DTO values indicate a better fairness–performance trade-off.

Model / Regime	Method	PPL ↓	HellaSwag ↑	MMLU ↑	UnQover ↑	BBQ ↑	CP ↓	DTO ↓
				Qwen-2.5-7B-Instruct				
Dense	–	7.14	56.99	68.71	73.17	86.30	60.82	0.291
2:4 SS	SparseGPT	15.89	44.72	50.41	28.45	64.10	57.60	0.616
	Debias-SparseGPT	16.17	44.53	48.16	24.94	66.70	57.72	0.645
2:4 SS + UltraChat	SparseGPT	13.84	46.35	53.84	42.46	75.20	59.15	0.522
	Debias-SparseGPT	13.37	46.54	54.17	47.26	78.60	56.65	0.494

Table 13: Evaluation of Debias-SparseGPT under structured 2:4 sparsity. "SS"= StereoSet. "SS + UltraChat" denotes a combined calibration dataset formed by merging StereoSet and UltraChat. Lower DTO values indicate a better fairness–performance trade-off.

Calibration Data	PPL ↓	MMLU ↑	UnQover ↑	DTO ↓
SS + UltraChat (4)	16.56	51.28	32.60	0.588
SS + UltraChat (16)	16.22	47.43	40.36	0.562
SS + UltraChat (64)	15.28	52.50	50.32	0.486
SS + UltraChat (256)	**13.37**	**54.17**	**65.00**	**0.408**
SS + UltraChat (1024)	13.65	51.89	48.04	0.501

Table 14: Effect of UltraChat calibration size on the performance of Qwen-2.5-7B-Instruct compressed with Debias-SparseGPT under 2:4 semi-structured sparsity. Each row corresponds to a different number of UltraChat calibration examples (4, 16, 64, 256, 1024). We report WikiText-2 perplexity, zero-shot MMLU accuracy, UnQover accuracy, and the resulting DTO score.