052

053

054

House of Cards: Massive Weights in LLMs

Anonymous Authors¹

Abstract

Massive activations, which manifest in specific 012 feature dimensions of hidden states, introduce a significant bias in large language models (LLMs), leading to an overemphasis on the correspond-015 ing token. In this paper, we identify that massive activations originate not from the hidden state but from the intermediate state of a feed-forward 018 network module in an early layer. Expanding 019 on the previous observation that massive activa-020 tions occur only in specific feature dimensions, we dive deep into the weights that cause massive activations. Specifically, we define top-k massive weights as the weights that contribute to the dimensions with the top-k magnitudes in the in-025 termediate state. When these massive weights are set to zero, the functionality of LLMs is entirely disrupted. However, when all weights 028 except for massive weights are set to zero, it 029 results in a relatively minor performance drop, 030 even though a much larger number of weights are set to zero. This implies that during the pretraining process, learning is dominantly focused on massive weights. Building on this observa-034 tion, we propose a simple plug-and-play method 035 called MacDrop (massive weights curriculum **drop**out), to rely less on massive weights during parameter-efficient fine-tuning. This method applies dropout to the pre-trained massive weights, 039 starting with a high dropout probability and gradually decreasing it as fine-tuning progresses. 041 Through various experiments, including zero-shot downstream tasks, long-context tasks, and abla-043 tion studies, we demonstrate that MacDrop generally improves performance and strengthens ro-045 bustness. 046

1. Introduction

Large language models (LLMs), such as GPT (Achiam et al., 2023) and Llama (Touvron et al., 2023; Dubey et al., 2024), have achieved remarkable success across diverse natural language tasks (Roziere et al., 2023; Mitra et al., 2024; Labrak et al., 2024; Wu et al., 2023). Their success is largely attributed to the pre-training phase, during which they are trained on extensive high-quality corpora datasets to predict the next token (Longpre et al., 2024; Zhao et al., 2024; Shen et al., 2023). However, despite the impressive achievements of LLMs, a crucial gap remains in our understanding of the underlying mechanisms that drive their remarkable performance.

Recently, Xiao et al. (2024) uncovered an intriguing phenomenon in LLMs, referred to as attention sinks: an unexpectedly large portion of attention is directed toward the initial tokens, regardless of their semantic context, after a small number of early layers. They demonstrated that under a restricted budget, focusing attention solely on recent window leads to poor performance, and that performance is recovered when initial tokens are included. Based on this observation, they proposed StreamingLLM, which retains the key-value caches of the initial sink tokens and the recent tokens for streaming use of LLMs. Yu et al. (2024b) further investigated the attention sinks phenomenon, finding that attention sinks can appear both in the initial tokens and in later tokens with less semantic importance (e.g., '.' and (n'). They showed that when sink tokens appear later in a sequence, sink tokens can potentially result in performance degradation. Inspired by this observation, they proposed a head-wise attention calibration technique without requiring additional training. Concurrently, Sun et al. (2024a) discovered the existence of massive activations in the hidden states of LLMs, with magnitudes substantially larger than the others. Massive activations are jointly identified based on their sequence and feature dimensions within the hidden states. Specifically, massive activations occur at the initial tokens and weak semantic tokens according to the model, and are consistently present in only a few fixed feature dimensions. Moreover, they connected massive activations with attention sinks, suggesting that massive activations inject implicit bias into the self-attention mechanism throughout the pretraining phase.

 ¹Anonymous Institution, Anonymous City, Anonymous Region,
 Anonymous Country. Correspondence to: Anonymous Author
 <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

House of Cards: Massive Weights in LLMs



Figure 1. (a) Massive weights are defined as the rows of W_{gate} and W_{up} in a specific layer l using the bos token, which produce the top-k magnitudes of the intermediate state \hat{h}_{l}^{inter} . Because massive weights are defined within a single layer l, the ratio of massive weights is significantly low compared to the overall number of parameters. For instance, in the case of Llama-3-8B, the proportion of the top-5 massive weights is 0.0005% of the model's total parameters. (b) When the top-5 massive weights are zeroed out, instruction-tuned LLMs completely lose their ability to generate text. On the other hand, when only the top-5 massive weights remain unchanged in W_{gate} and W_{up} , instruction-tuned LLMs retain their generation capability.

076 In this paper, we first delve deeper into massive activations, 077 providing two key observations. (1) The bos token placed 078 at the starting position always has massive activations in the 079 same feature dimensions and makes attention sinks. This ob-080 servation enables us to focus only on the feature dimensions, 081 instead of both the sequence and feature dimensions, when 082 addressing massive activations. Namely, a simplified and 083 consistent analysis of massive activations can be achieved 084 by using only the bos token. (2) Massive activations orig-085 inate in the intermediate state \hat{h}_{l}^{inter} within an early layer 086 l, before appearing in the hidden state h_l , as illustrated in 087 Figure 1(a). Namely, massive activations triggered in h_1^{inter} 088 are subsequently and continuously propagated through skip 089 connections. This observation implies that the feed-forward 090 network in layer l plays a crucial role in LLMs. 091

Next, we shift our focus from activations to the weights, 092 relying on the fact that massive activations consistently ap-093 pear in the same feature dimensions. In detail, we define 094 the top-k massive weights as the rows of W_{up} and W_{gate} 095 in the feed-forward network at layer l that produce the top-k096 magnitudes of the intermediate state \hat{h}_{l}^{inter} , as illustrated 097 in Figure 1(a). It is important to note that massive weights, 098 defined within a single layer l, account for a substantially 099 small fraction compared to the model's total parameters. 100 This holds true even when compared to the entire W_{up} and W_{gate} . Nevertheless, massive weights are crucial factors that can completely influence the performance of LLMs. Figure 1(b) presents the generated responses of three mod-104 els to the given user prompt: original model, top-5 massive 105 weights attacked model, and other weights attacked model. 106 Here, other weights represent all weights in W_{up} and W_{gate} 107 at layer l that do not belong to the top-5 massive weights, 108

and an attack sets corresponding weights to zero. When the massive weights are attacked, the model becomes poor and repeats the user prompt. On the contrary, when other weights are attacked, the model does not entirely lose its generation capability, even though a much greater number of weights are set to zero in the same projection matrices. These observations imply that massive weights are dominantly learned during pre-training and highly related to the performance of LLMs.

Finally, we propose a straightforward plug-and-play method during parameter-efficient fine-tuning, named massive weights curriculum dropout (MacDrop). This method applies dropout to the pre-trained massive weights, rather than additional trainable weights, starting with a high dropout rate that is progressively reduced throughout the fine-tuning phase. The intuition behind MacDrop is that a high initial dropout rate encourages the model to lessen dependence on the massive weights predominantly learned during the pretraining phase. Then, reducing the dropout rate facilitates a more stable convergence, ensuring the pre-trained model is leveraged with neglectable damage by the end of fine-tuning. Through various ablation studies, we examine the effects of dropout scope and dropout probability scheduling. Finally, we demonstrate that MacDrop generally enhances model performance and robustness in zero-shot downstream tasks and long context tasks.

2. Massive Weights

In this section, we review the key observations on massive activations reported by Sun et al. (2024a) and extend the analysis by exploring various states using the *bos* token,

070

073

074

110 which was not covered thoroughly. Based on this expanded 111 analysis, we formally define top-k massive weights in a 112 specific layer and investigate their importance through two 113 opposite types of attacks.

2.1. Prerequisite: Massive Activations

114 115

116

125

126

127

128

149

150

151

152

153

Autoregressive Transformers (Vaswani et al., 2017) are 117 structured with L decoding layers. Each layer $l \in [1, L]$ 118 includes an attention (ATTN) module and a feed-forward 119 network (FFN) module. These modules are connected via 120 residual connections (He et al., 2016), each following a layer 121 normalization (LN) layer (Ba, 2016). The previous hidden 122 state h_{l-1} is fed into layer l and processed to produce the 123 subsequent hidden state h_l : 124

$$h_{l} = \hat{h}_{l} + \text{FFN}(\text{LN}(\hat{h}_{l})),$$
where $\hat{h}_{l} = h_{l-1} + \text{ATTN}(\text{LN}(h_{l-1}))$
(1)

129 Sun et al. (2024a) primarily concentrated on the activations 130 within hidden states, identifying that certain activations ex-131 hibit exceptionally large magnitudes, which they termed 132 massive activations. Massive activations are observed at 133 the starting position (i.e., input-agnostic) or at the delim-134 iter tokens, depending on the model. Furthermore, these 135 activations are confined to a small number of fixed feature 136 dimensions, even within these tokens. These activations 137 initially emerge after passing through several early layers 138 and then decreases as they near the last layer. 139

Massive activations are strongly tied to the attention sinks 140 phenomenon, as identified by Xiao et al. (2024), in which 141 attention is abnormally concentrated on a small subset of 142 tokens. In detail, a given query state tends to have positive 143 cosine similarity with the key states of the tokens exhibit-144 ing massive activations, and negative cosine similarity with 145 those of other tokens. Consequently, attention is heavily skewed toward the tokens associated with massive activa-147 tions. Detailed related work is explained in Appendix H. 148

2.2. Further Analysis on Massive Acitvations

We primarily utilize the Llama-3-8B model (Dubey et al., 2024) and explicitly specify other models when necessary.

154 bos token placed at the starting position always has mas-155 sive activations. We begin by examining whether any 156 specific condition consistently triggers massive activations. 157 The existence of such a condition would greatly facilitate the 158 analysis and algorithm development for handling massive 159 activations. Following Sun et al. (2024a), massive activa-160 tions are observed when any token is placed at the starting 161 position; however, we find the cases where the token at the 162 starting position does not trigger massive activations and 163 attention sinks, such as Mistral-7B. 164



Figure 2. (Top) Magnitudes of the hidden state and (Bottom) attention scores after Softmax of Mistral-7B, according to the position of the *bos* token. The described hidden state is the output of layer 16 (i.e., h_{16}). The attention scores are calculated at layer 17 (i.e., after massive activations appear) and averaged across different heads.

Figure 2 describes the magnitudes of activations of the hidden state and normalized attention scores of Mistral-7B according to the position of the bos token, after massive activations appear. The reason for arbitrarily inserting the bos token is based on the previous observation that nonsemantic tokens can trigger massive activations in certain LLMs. We use the implementation¹ of massive activations for this example and visualization. Mistral-7B has massive activations at the first delimiter token '.', not at the starting position (first column). However, when the bos token is placed at the starting position, it triggers massive activations and the first delimiter token loses its massive activations (second column). When the bos token is inserted in the middle or ending position after the first delimiter token, massive activations are observed in both tokens (third and fourth columns). On the other hand, there are models that respond to the starting position but not to the bos token, such as Llama-2, detailed in Appendix C. Therefore, by considering both conditions, we use only the bos token placed at the starting position for the continuation of analysis and algorithm development.

Massive activations originate in the intermediate state of a FFN module. Next, we trace various states in early layers until the first massive activations appear, using the *bos* token. Specifically, we monitor h_{l-1} , $LN(h_{l-1})$, ATTN($LN(h_{l-1})$), \hat{h}_l , $LN(\hat{h}_l)$, and FFN($LN(\hat{h}_l)$) in Eq. (1) throughout early layers. Figure 3(a) illustrates the magnitudes of various states in layers $l \in [1, 3]$. It is observed that FFN($LN(\hat{h}_2)$) has massive activations before h_2 . With Figure 3(b), which describes the top three and median magnitudes of the hidden state¹, it is observed that the massive activations generated within a FFN module at layer 2 are transmitted directly to the next hidden state and then propagated solely through the residual connections.

¹https://github.com/locuslab/massive-activations

House of Cards: Massive Weights in LLMs

Table 1.	Perplexity a	and zero-sl	hot downs	tream tasks	s perform	ance acco	ording to	the atta	ck.	
Models	WikiText	C4	PG-19	Avg. (\downarrow)	ARC-E	ARC-C	BoolQ	PIQA	WG	Avg. (\uparrow)
Llama-3-8B	5.75	9.94	8.98	8.22	77.4	52.7	81.4	80.8	72.9	73.0
top-5 zeroing top-5 retaining	104.57 10.15	132.83 23.98	130.83 28.72	122.74 20.95	29.1 75.0	22.0 48.0	41.8 80.2	53.8 77.7	50.3 74.1	39.4 71.0
Llama-3-70B	2.68	7.59	6.02	5.43	85.8	64.2	85.3	84.6	80.5	80.1
top-5 zeroing top-5 retaining	11135.81 3.47	7288.86 9.93	4696.49 7.26	7707.05 6.89	28.6 45.9	22.8 25.6	38.0 83.9	55.5 64.6	49.5 77.0	38.9 59.4
Llama-3.1-405B (8bit)	1.41	6.20	3.23	3.61	86.3	66.0	88.2	85.0	81.1	81.3
top-5 zeroing top-5 retaining	1785.56 2.47	985.36 9.36	633.40 5.61	1134.77 5.81	26.6 83.0	25.9 62.5	37.8 83.1	49.7 83.2	50.3 70.3	38.1 76.4



(a) Various states in early layers.

178

179

180

181 182

183

184

185

186

187

188

189

190

191

193

195

196

197

199

200

204

205

206

(c) Intermediate states.

Figure 3. (a) Magnitudes of various states and (b and c) the top three and median magnitudes of hidden states and intermediate states across layers. These results show that massive activations in the hidden state originate from those in the intermediate state in a FFN module in an early layer.

Furthermore, we decompose a FFN module into $W_{down}(\sigma(W_{gate}(\cdot)) \odot W_{up}(\cdot))$, to analyze the intermediate states (i.e., the output of $\sigma(W_{gate}(\cdot)) \odot W_{up}(\cdot)$). Figure 3(c) describes the top three and median magnitudes of the intermediate state across layers. It is demonstrated that massive activations originate in the intermediate state of a FFN module in an early layer l. This result implies that W_{up} and W_{qate} in layer l are closely tied to massive activations. Additional results for other LLMs are provided in the Appendix D.

2.3. Massive Weights

208 Massive weights are defined based on massive activations 209 in the intermediate state at layer l, denoted as \hat{h}_{l}^{inter} , when 210 the bos token is fed into LLMs. To elaborate, we define the 211 rows in the projection matrix W_{up} (and W_{gate} , if it exists) 212 that correspond to the indices of the top-k magnitudes in 213 \hat{h}_{l}^{inter} as top-k massive weights, depicted in Figure 1(a). It 214 is noted that massive weights are defined within one specific 215 layer, which means the number of massive weights is signif-216 icantly smaller compared to the total number of parameters 217 in LLMs. For example, in Llama-3-8B, the number of top-k218 massive weights is calculated as $2 \times k \times 4096$, where 4096 219

represents the dimensions of hidden state. If k is set to 5, massive weights account for approximately 0.0005% of the total parameters in Llama-3-8B, approximately 0.0001% in Llama-3-70B, and approximately 0.00004% in Llama-3.1-405B.

Massive weights are extremely small in quantity, their impact is tremendous. To assess the significance of massive weights, we conduct two types of attacks: top-k zeroing and top-k retaining. Note that these attacks only affect the W_{up} and W_{qate} projection matrices in layer l, where massive weights are present. The first attack is to set the top-kmassive weights to zero (i.e., darker orange weights in Figure 1(a)). In essence, this attack is very similar to the one proposed in Sun et al. (2024a), where massive activations in the hidden state are zeroed out in a single layer. The difference is that their attack targets the hidden state, while our attack targets the intermediate state. The second attack is to set all weights to zero except for top-k massive weights (i.e., lighter orange weights in Figure 1(a)). That is, the number of rows being damaged in each attack is k and the dimensions of intermediate state -k, respectively.

Following Sun et al. (2024a), we assess perplexity¹ on three datasets: WikiText (Merity et al., 2017), C4 (Raffel et al., 2020), and PG-19 (Rae et al., 2020). Additionally, we evaluate zero-shot accuracy² on five tasks: Arc-Easy (ARC-E), Arc-Challenge (ARC-C) (Clark et al., 2018), BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), and WinoGrande (WG) (Sakaguchi et al., 2021). Table 1 presents the results of two attacks on Llama-3-8B, Llama-3-70B, and Llama-3.1-405B (8bit)³, when k is set to 5. Llama-3-8B has its massive weights in layer 2 out of 32 layers, Llama-3-70B in layer 4 out of 80 layers, and Llama-3.1-405B in layer 6 out of 126 layers. Detailed indices of massive weights are provided in Appendix B, including various LLMs. The larger the difference from the original performance, the stronger the attack.

²https://github.com/EleutherAI/Im-evaluation-harness

³We use 8xA100-80G GPUs for our work; therefore, we employ an 8-bit model.





Figure 5. Probability of experts of Mixtral-8x7b for the bos token. In the layer with massive weights (i.e., layer 2), the router probability between experts becomes completely skewed to one expert.

239 Top-5 zeroing is a much stronger attack than top-5 retaining, 240 even though top-5 retaining sets several thousands times 241 more weights to zero compared to top-5 zeroing does for the 242 same projection matrices. This means that in the projections 243 W_{up} and W_{qate} at layer l, having only massive weights is 244 significantly better than having all weights except for mas-245 sive weights. In detail, similar to the findings of Sun et al. 246 (2024a), the top-k zeroing attack proves to be highly effec-247 tive in disrupting the Llama-3 family, even for extremely 248 large-scale models (e.g., 70B and 405B). On the other hand, 249 the top-k retaining attack does not cause complete damage. 250 In conclusion, these findings reveal that massive weights 251 are predominantly learned during pre-training, highlighting 252 their essential contribution to the performance of LLMs. 253

220

221

222

223

224

225

227

228

229

230 231

232 233

234

235 236

237

238

271

Moreover, massive weights also exist in instruction-tuned 254 LLMs such as Llama-3-8B-Instruct. These attacks are ef-255 fective, as depicted in Figure 1(b). When massive weights 256 are set to zero (i.e., darker orange box), the model repeats 257 the same text as the user prompt. On the other hand, when 258 all weights are set to zero except for massive weights (i.e., 259 lighter orange box), the model retains its ability to generate text, although the generated text differs from the original. 261

263 The value of k, which affects performance, varies de-264 pending on the model architecture. We examine the 265 robustness of various LLMs against the top-k zeroing and 266 retaining attacks, with a focus on the impact of the parame-267 ter k. Llama-2 (Touvron et al., 2023), Llama-3 (Dubey et al., 268 2024), Mistral (Jiang et al., 2023) and Mixtral (Jiang et al., 269 2024), Phi-3 (Abdin et al., 2024), and Gemma-2 (Team 270 et al., 2024) families are used. Figure 4 illustrates the mean zero-shot accuracy of LLMs under the two attacks, accord-272 ing to the k. In top-k zeroing, more weights are set to zero 273 as k increases, whereas in top-k retaining, more weights are 274

set to zero as k decreases. When k is 0 in top-k zeroing, it corresponds to the original performance without any attack, whereas, when k is 0 in top-k retaining, it sets the entire weights of W_{up} and W_{gate} in layer l to zero.

The Llama families are highly sensitive to massive weights. In the top-k zeroing, a noticeable performance drop occurs even when k is as small as 3, irrespective of the model's scale. In top-k retaining, when k is set to 1 (i.e., with only one row active in W_{up} and W_{gate} in layer l), the performance nearly reaches the original level in smaller-scaled models (\leq 13B). While, for larger-scaled models (\geq 70B), the top-30 massive weights are required to maintain performance. Moreover, since different versions of Llama families pre-trained on varying datasets (e.g., with different cutoff dates) exhibit similar tendencies, it can be inferred that the pre-training dataset itself does not significantly influence the emergence of massive phenomena.

Similarly, Mistral is also significantly disrupted, when kis set to 5. Mixtral is a sparse Mixture of Experts (MoE) architecture that uses a top-2 routing mechanism, where two experts are activated among eight FFN modules in each layer. To attack the Mixtral model, we identify the active experts in the layer with massive weights using the bos token. Figure 5 describes the probability distribution of experts in the Mixtral model across all layers. Notably, it is observed that when massive activations occur, a single expert (i.e., expert 4) is assigned a significantly higher probability than the others. Therefore, we target only the W_{up} and W_{gate} of this expert, rather than all experts. Although Mixtral does not completely break down, there is a considerable decline in performance when the top-50 massive weights are zeroed out. These results indicate that, despite the immense resources required to build high-performance LLMs, they can collapse like a house of cards even under minimal attacks.

275 The Phi-3 family exhibits different robustness against at-276 tacks depending on the model size. As noted by Abdin et al. 277 (2024), the phi-3-mini (3.8B) is trained on 3.3T tokens, 278 while the phi-3-medium (14B) is trained on 4.8T tokens. A 279 key architectural difference from the Llama family is the 280 use of dropout to the outputs of both the ATTN and FFN 281 modules, formed by Eq. (2). While a specific recipe for 282 dropout is not provided in the technical report (Abdin et al., 283 2024), in the case of phi-3-medium, applying dropout with 284 longer pre-training might ensure that the residual connec-285 tions contribute meaningfully, mitigating the risk of exces-286 sive dependence on massive weights. 287

289

290

303

304

305

306

307

308

309

310

311 312

313

314

315

317

318

319

320

321

322

323

324

325

327

328

329

$$h_{l} = \hat{h}_{l} + \text{dropout}(\text{FFN}(\text{LN}(\hat{h}_{l}))),$$
where $\hat{h}_{l} = h_{l-1} + \text{dropout}(\text{ATTN}(\text{LN}(h_{l-1})))$
(2)

291 The Gemma-2 family is exceptionally resilient to the top-k292 zeroing attack, maintaining almost no loss in performance 293 even when k is large. Additionally, even if W_{up} and W_{qate} 294 are entirely eliminated (i.e., k = 0 in top-k retaining), there 295 is no noticeable performance degradation. This family in-296 corporates two additional LN layers after both the ATTN 297 and FFN modules, formed by Eq. (3). These added normalization layers result in completely different hidden and 299 intermediate states compared to other LLMs, as described in 300 Appendix D. Furthermore, attention sinks are not observed 301 in the Gemma-2 family, as shown in Appendix E. 302

$$h_{l} = \hat{h}_{l} + \text{LN}(\text{FFN}(\text{LN}(\hat{h}_{l}))),$$
where $\hat{h}_{l} = h_{l-1} + \text{LN}(\text{ATTN}(\text{LN}(h_{l-1})))$
(3)

These observations imply that LLMs not sensitive to massive phenomena may not perform effectively with algorithms based on massive phenomena. Further discussion is provided in Appendix F.

3. Massive Weights Curriculum Dropout

In this section, we propose a straightforward plug-andplay method, termed massive weights curriculum dropout 316 (MacDrop), during parameter-efficient fine-tuning such as low rank adaptation (Hu et al., 2022). This method applies dropout to the pre-trained massive weights with a curriculum that gradually reduces the dropout probability. The reason for applying dropout to weights (Wan et al., 2013) instead of activations (Srivastava et al., 2014) is that the number of massive activations is only k, but that of massive weights is $k \times d$, where d is the dimension of the hidden states. It is important to note that our method does not modify the trainable parameters of adapters; instead, it is applied to the pre-trained frozen weights. Therefore, MacDrop can be applied orthogonally to the process of training the adapter.

Algorithm 1: Top-k massive weights curriculum dropout (MacDrop) in pseudo PyTorch style

- // Dropout is only executed in layer l_{i} where h_l^{inter} appears.
- **Input:** k, massive intermediate state h_l^{inter} of the bos token, initial dropout probability p_0 , total steps T
- // massive activations in the intermediate state
- 1 _, sorted_indices = torch.sort(torch.abs(h_l^{inter}), descending=True)
- 2 massive_indices = sorted_indices[:k] // pre-trained massive weights
- 3 massive_W_up = copy(W_up[massive_indices, :])
- 4 massive_W_gate = copy(W_gate[massive_indices, :])
- 5 for t = 1 to T do // decreasing dropout probability
- $p = p_0 \times \left(1 \frac{t}{T}\right)$ 6 // pre-trained massive weights dropout $mask = (torch.rand(massive_W_up.shape) > p).int()$ 7 8 W_up[massive_indices, :] *= mask W_gate[massive_indices, :] *= mask 9 tr_loss_step = training_step(model, inputs) 10 // pre-trained massive weights rollback
- $W_{up}[massive_indices, :] = massive_W_{up}$ 11
- 12 W_gate[massive_indices, :] = massive_W_gate

MacDrop is motivated by the observation that massive weights are predominantly learned during pre-training, and that zeroing them can severely undermine LLMs. Therefore, at the early stages of fine-tuning, the objective is to reduce the reliance on massive weights, as their excessive dominance may lead the model to over-rely on specific patterns. Moreover, considering that the undamaged pretrained model is used after fine-tuning is finished, we develop a strategy to adjust the dropout rate using a curriculum.

Algorithm 1 describes MacDrop in a pseudo PyTorch (Paszke et al., 2019) style, and is implemented within the trainer code of transformers⁴. Initially, massive weights are identified using the bos token before fine-tuning (Lines 1-4). Subsequently, an adapter is trained while the pre-trained massive weights are dropped. Meanwhile, a curriculum strategy is applied to progressively enable the use of the original pre-trained weights without masking. Note that in Line 10, 'model' includes both the masked pre-trained network and a trainable adapter. When we implement Lines 1-4 in practice, we precomputed the massive indices and loaded them. Lines 6-9 and 11-12 are the additional computation for MacDrop, which requires neglectable overhead (e.g., approximately 0.35 second per step for Llama-3-8B using LoRA on 8xA100 GPUs).

⁴https://github.com/huggingface/transformers/tree/main/src/ transformers

House of Cards: Massive Weights in LLMs

	Single	-docum	ent QA	Multi-	docume	ent QA	Sun	ımariza	tion	Few-	shot lea	rning	Syn	thetic
Method	NitvQA	$Q_{a_{Sper}}$	MFen	HotpotQ4	^{2WikiMQA}	Musique	GovReport	QMS _{um}	MultiNews	TREC	TriviaQ4	SAMSum	P_{Count}	PRe
						Mo	odel: M	eta-Llaı	na-3-8E	3				-
LoRA	26.03	30.38	53.38	26.30	23.05	11.96	29.00	22.81	26.43	72.50	81.14	44.27	2.63	32.0
+ MacDrop	25.31	34.05	46.84	38.06	28.99	17.92	29.62	22.86	26.64	72.00	89.34	45.08	3.00	27.5
+ MacDrop	26.51	30.99	53 37	29.35	25.57	12.01	29.20	23.55	26.34	73.50	86.63	45.22	2.01	25.5
- 11405105				27.00	20.00	M	Iodel: N	listral-7	7B-v0.1	70.00	00.00		2.00	
LoRA	22.52	34.64	35.65	32.11	19.80	12.73	27.22	21.98	26.73	69.00	87.50	41.70	1.00	21.0
+MacDrop	23.49	38.51	36.11	37.78	27.60	14.91	26.40	22.53	26.92	69.50	89.92	37.07	1.55	20.0
DoRA	22.79	34.52	35.55	30.87	17.84	12.26	27.45	22.15	26.52	70.00	88.05	41.56	1.00	20.5
+MacDrop	23.10	35.10	35.00	29.53	23.77	10.50	27.14	22.63	27.50	69.00	89.56	38.96	1.00	21.4
Table	3. Zero-	-shot do	ownstrea	am task	s perfoi	mance		Ta	able 4. Z	Zero-sh	ot dow	nstream	ı tasks	s perf
Model	Metho	od Al	RC-E A	RC-C B	oolQ P	IQA W	G Avg.	ze	roing a	ttack, r	elated t	o robus	stness	<u>. </u>
	LoR	A 7	9.6	58.2	83.9 8	32.4 75	.9 76.0		Mode	1 1	Method	ARC	C-E A	RC-C
1 2 0 0	+ MacD	rop 8	32.9 5	58.3	83.9 8	32.6 75	.0 76.5				LoRA	29	.9 2	22.9
Jama-3-8B-	DoR	A 8	30.8 .5	57.7	83.9 8	32.5 75	.8 76.1	T	lama 2	о р + №	lacDro	op 36	.8 2	24.7
-	+ MacD	rop 8	31.9 5	58.2	83.9 8	32.2 75	.6 76.4	L	iaina-3-	-0D	DoRA	29	.8 2	23.0
	LoR	A 7	8.5	54.9	84.9 8	32.9 75	.3 75.3			+ M	lacDro	op 78	.7 .5	53.7
-	+ MacD	rop 8	30.9	56.7	85.0 8	33.0 75	3 76.2				LoRA	54	6 3	34.8

4. Experiments

Mistral-7B

357

358 359 360

361

362

383

384

4.1. Zero-shot Downstream Task

DoRA

+ MacDrop

363 We fine-tune the Llama-3-8B and Mistral-7B using the al-364 paca_gpt4_en dataset (Peng et al., 2023) for 3 epochs (579 365 steps), and evaluate on five zero-shot tasks. We use two 366 parameter-efficient fine-tuning (PEFT) methods, LoRA (Hu 367 et al., 2022) and DoRA (yang Liu et al., 2024). DoRA 368 decomposes the pre-trained weights into two components, 369 magnitude and direction, and applies LoRA to the direction 370 component. Our method is based on the implementation of 371 Llama-Factory (Zheng et al., 2024)⁵. For MacDrop, k and 372 p_0 are set to 5 and 0.8, respectively. Details for implementa-373 tions are explained in Appendix A. 374

78.4

80.6

55.1

56.7

85.0

85.3

82.9 75.1 75.3

82.9 75.1 76.1

Table 3 presents the results on zero-shot downstream tasks. 375 For both the models and methods, MacDrop consistently 376 leads to performance gains, especially in ARC-Easy and 377 ARC-Challenge tasks. Note that we do not claim MacDrop 378 is superior to LoRA/DoRA; rather, we demonstrate that 379 LoRA/DoRA performs better when MacDrop is applied 380 than when it is not. Results for other LLMs are provided in 381 Appendix F. 382

⁵https://github.com/hiyouga/LLaMA-Factory

ormance under the top-3

Code

у С

72.90

73.17

71.57

73.46 69.44

70.37 65.87

71.88 65.38

Q Ŕ

69.70

68.25

73.90 69.32 39.48

65.47

71.17 65.53 36.93

Avg.

39.03

40.54

39.07

36.91

38.03

36.77

Model	Method	ARC-E	ARC-C	BoolQ	PIQA	WG	Avg.
Llama-3-8B	LoRA + MacDrop	29.9 36.8	22.9 24.7	45.7 64.3	52.9 58.5	50.7 54.1	40.4 47.7
	DoRA + MacDrop	29.8 78.7	23.0 53.7	46.0 79.9	52.6 79.0	50.0 72.4	40.3 72.7
Mistral-7B	LoRA + MacDrop	54.6 69.2	34.8 45.0	58.0 78.3	74.8 79.3	57.5 64.1	55.9 67.2
	DoRA + MacDrop	55.7 78.1	35.3 52.4	58.6 84.1	74.4 82.3	58.2 69.5	56.4 73.3

MacDrop is designed to mitigate dependence on massive weights during PEFT. To assess whether this dependency is effectively reduced, we attack the previous fine-tuned models in Table 3. Table 4 shows the performance changes whether applying MacDrop under the top-3 zeroing attack. The zeroing attack severely degrades the performance of fine-tuned models without MacDrop, as seen in the case of Llama-3-8B with DoRA, where performance drops from 76.1 to 40.3. However, fine-tuned models with MacDrop exhibit significantly better performance under attack, indicating better robustness. Especially, when MacDrop is combined with DoRA, it demonstrates remarkable robustness. For instance, in the case of Llama-3-8B with DoRA, where performance drops from 76.4 to 72.7.

4.2. Long Context Task

We evaluate the two LLMs on LongBench (Bai et al., 2024), a benchmark specifically designed to assess the ability to understand long contexts. This includes 5 sub-categories and 16 English datasets: single-document QA, multi-document QA, summarization, few-shot learning, synthetic, and code generation. We set the max length of models to 7,500. Table 2 shows that MacDrop increases the performance when understanding long context is required.



Figure 6. Mean zero-shot accuracy according to the dropout scope and probability *p*.

396 4.3. Ablation Study

385

386

387 388

389

390

395

397

398

399

400

424

425

426

We further provide ablation studies related to MacDrop using Llama-3-8B. The variations used in this section can serve as a baseline.

401 4.3.1. DROPOUT SCOPE AND PROBABILITY

403 We investigate the effect of dropout scope and probability compared to the original performance achieved through 404 405 LoRA without dropout. This ablation study is also conducted on the W_{up} and W_{gate} projection matrices in layer 406 l. The dropout scope is divided into three categories: all 407 408 weights, massive weights, all weights except for massive 409 weights. Additionally, to assess the impact of dropout prob-410 ability, it is kept constant throughout the fine-tuning process, 411 without using a curriculum.

412 Figure 6 illustrates the mean zero-shot accuracy according to 413 the dropout scope and dropout probability p. It is observed 414 that among three scopes, the original performance (i.e., with-415 out dropout), represented by the dotted line at 76.0, can be 416 surpassed only when dropout is applied solely to massive 417 weights. Nevertheless, if strong dropout (e.g., $p \ge 0.85$) 418 is maintained on the pre-trained massive weights during 419 fine-tuning, performance deteriorates. This highlights the 420 need to safeguard the pre-trained massive weights during 421 the final stages of fine-tuning, because we are ultimately 422 using them without causing any damage. 423

4.3.2. CURRICULUM METHODS AND INITIAL DROPOUT PROBABILITY

427 We investigate the effect of curriculum methods and ini-428 tial dropout probability p_0 in MacDrop, when LoRA is 429 applied. We compare four curriculum methods: step-wise 430 linear (Step), before epoch-wise linear (Epoch(before)), af-431 ter epoch-wise linear (Epoch(after)), and exponential (Exp.). In formula, Step is defined as $p = p_0 \times (1 - \frac{t_{step}}{T_{step}})$. 432 Epoch(before) and Epoch(after) are defined as $p = p_0 \times (1 - \frac{t_{epoch} - 1}{T_{epoch}})$ and $p = p_0 \times (1 - \frac{t_{epoch}}{T_{epoch}})$, respectively. Exp. is defined as $p = p_0 \times exp(-\alpha t_{step})$. Figure 7 describes dropout 433 434 435 436 probability p according to curriculum methods, when p_0 is 437 1.0. The distinct difference between Epoch(before) and 438 Epoch(after) is that at the final epoch, the former continues 439



Figure 7. Curriculum methods.

Table 5. Mean zero-shot accuracy according to curriculum methods and initial dropout probability.

Curriculum		p	0	
Method	0.2	0.5	0.8	1.0
Step	76.0	76.3	76.5	75.5
Epoch(before)	76.1	76.1	76.1	75.5
Epoch(after)	75.9	76.0	76.2	76.3
Exp. ($\alpha = 0.01$)	76.0	76.2	76.5	76.4
Exp. ($\alpha = 0.05$)	76.0	76.1	76.2	76.3
Exp. ($\alpha = 0.10$)	76.0	76.1	76.1	76.2
Mean	76.0	76.1	76.3	76.0

to apply dropout to the pre-trained massive weights with a probability of $p_0 \times \frac{1}{T_{epoch}}$, while the latter fully utilizes the pre-trained massive weights.

Table 5 presents mean zero-shot accuracy according to curriculum methods and initial dropout probability p_0 . It is observed that step-based curriculum methods (e.g., Step and Exp.) generally achieve greater performance improvements compared to epoch-based curriculum methods (e.g., Epoch(before) and Epoch(after)). Nevertheless, when the initial dropout probability is relatively low (e.g., $p_0 < 0.2$), even step-based curriculum methods fail to bring any performance gain compared to the original performance of 76.0. Additionally, it is shown that using a smaller α in the Exp. method leads to greater performance improvements, suggesting that a rapid decline in dropout probability to zero can diminish the effectiveness of MacDrop. On the other hand, for the Step and Epoch(before) methods, a significant performance drop is observed at a p_0 value of 1.0, highlighting the necessity of a near-zero dropout probability for the end of fine-tuning. In conclusion, for MacDrop, we recommend using the Step or Exp. with a smaller α , initiated from a moderately high p_0 .

5. Conclusion

In this paper, we explore the weight space of LLMs and identify the presence of massive weights within a FFN module in an early layer, which are predominantly pre-trained and have a significant impact on the performance of LLMs. Based on our observation, we propose a plug-and-play finetuning method called MacDrop, which applies dropout to the pre-trained massive weights, rather than to the parameters of adapters, during parameter-efficient fine-tuning. We hope that our findings will inspire future research in weight space of LLMs.

440 Impact Statement

441 This paper seeks to contribute to the advancement of large 442 language models, particularly by providing deeper insights 443 into their internal mechanisms. Our research demonstrates, 444 through various open-sourced LLMs, that in many cases, 445 models can be easily disrupted by zeroing massive weights 446 if access to their weights is available. Subsequently, we 447 improve robustness against such attacks through MacDrop. 448 The existence of massive weights does not inherently present 449 a risk if the model weights remain undisclosed. However, 450 we believe that model developers should understand these 451 internal mechanisms and take responsibility for ensuring 452 robustness. In this vein, our work has the potential to make 453 a meaningful impact. 454

References

455

456

457

458

459

460

461

462 463

464

465

466

467 468

469

470

471

472 473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

- Abdin, M., Jacobs, S. A., Awan, A. A., Aneja, J., Awadallah, A., Awadalla, H., Bach, N., Bahree, A., Bakhtiari, A., Behl, H., et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.
- Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- An, Y., Zhao, X., Yu, T., Tang, M., and Wang, J. Fluctuationbased adaptive structured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 10865–10873, 2024.
- Ba, J. L. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- Bai, Y., Lv, X., Zhang, J., Lyu, H., Tang, J., Huang, Z., Du, Z., Liu, X., Zeng, A., Hou, L., Dong, Y., Tang, J., and Li, J. LongBench: A bilingual, multitask benchmark for long context understanding. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172.
- Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *arXiv:2004.05150*, 2020.
- Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

- Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Leveling, J., Monz, C., Pecina, P., Post, M., Saint-Amand, H., Soricut, R., Specia, L., and Tamchyna, A. Findings of the 2014 workshop on statistical machine translation. In Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., and Specia, L. (eds.), *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pp. 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-3302. URL https://aclanthology.org/W14–3302.
- Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19–1300.
- Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.
- Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- Darcet, T., Oquab, M., Mairal, J., and Bojanowski, P. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum? id=2dn03LLiJ1.
- Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- Ge, S., Zhang, Y., Liu, L., Zhang, M., Han, J., and Gao, J. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https: //openreview.net/forum?id=uNrFpDPMyo.
- He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

- Hooper, C., Kim, S., Mohammadzadeh, H., Mahoney,
 M. W., Shao, Y. S., Keutzer, K., and Gholami, A.
 Kvquant: Towards 10 million context length llm inference with kv cache quantization. *arXiv preprint arXiv:2401.18079*, 2024.
- Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y.,
 Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.
- Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C.,
 Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G.,
 Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary,
 B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna,
 E. B., Bressand, F., et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- Kim, S., Suk, J., Longpre, S., Lin, B. Y., Shin, J.,
 Welleck, S., Neubig, G., Lee, M., Lee, K., and Seo, M.
 Prometheus 2: An open source language model specialized in evaluating other language models. *arXiv preprint arXiv:2405.01535*, 2024.
- Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., 521 Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., De-522 vlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., 523 Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q., and 524 Petrov, S. Natural questions: A benchmark for question 525 answering research. Transactions of the Association for 526 Computational Linguistics, 7:452-466, 2019. doi: 10. 527 1162/tacl_a_00276. URL https://aclanthology. 528 org/Q19-1026. 529
- 530 Labrak, Y., Bazoge, A., Morin, E., Gourraud, P.-A., Rouvier, 531 M., and Dufour, R. BioMistral: A collection of open-532 source pretrained large language models for medical do-533 mains. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), 534 Findings of the Association for Computational Linguistics 535 ACL 2024, pp. 5848-5864, Bangkok, Thailand and virtual 536 meeting, August 2024. Association for Computational 537 Linguistics. URL https://aclanthology.org/ 538 2024.findings-acl.348. 539
- Li, Y., Huang, Y., Yang, B., Venkitesh, B., Locatelli, A.,
 Ye, H., Cai, T., Lewis, P., and Chen, D. Snapkv: Llm
 knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*, 2024.
- Lin, J., Tang, J., Tang, H., Yang, S., Chen, W.-M., Wang,
 W.-C., Xiao, G., Dang, X., Gan, C., and Han, S. Awq:
 Activation-aware weight quantization for on-device llm
 compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.

- Longpre, S., Yauney, G., Reif, E., Lee, K., Roberts, A., Zoph, B., Zhou, D., Wei, J., Robinson, K., Mimno, D., and Ippolito, D. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity. In Duh, K., Gomez, H., and Bethard, S. (eds.), Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pp. 3245–3276, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.179. URL https:// aclanthology.org/2024.naacl-long.179.
- Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *International Conference* on *Learning Representations*, 2017. URL https:// openreview.net/forum?id=Byj72udxe.
- Mitra, A., Khanpour, H., Rosset, C., and Awadallah, A. Orca-math: Unlocking the potential of slms in grade school math. *arXiv preprint arXiv:2402.14830*, 2024.
- Nallapati, R., Zhou, B., dos Santos, C., Gu.lçehre, Ç., and Xiang, B. Abstractive text summarization using sequenceto-sequence RNNs and beyond. In Riezler, S. and Goldberg, Y. (eds.), *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 280–290, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/K16-1028. URL https://aclanthology.org/K16-1028.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips. cc/paper_files/paper/2019/file/ bdbca288fee7f92f2bfa9f7012727740-Paper. pdf.
- Peng, B., Li, C., He, P., Galley, M., and Gao, J. Instruction tuning with gpt-4. arXiv preprint arXiv:2304.03277, 2023.
- Rae, J. W., Potapenko, A., Jayakumar, S. M., Hillier, C., and Lillicrap, T. P. Compressive transformers for longrange sequence modelling. In *International Conference* on Learning Representations, 2020. URL https:// openreview.net/forum?id=SylKikSYDH.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring

the limits of transfer learning with a unified text-to-text
transformer. *Journal of machine learning research*, 21
(140):1–67, 2020.

- Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Sauvestre, R., Remez, T., et al. Code llama: Open foundation models for code. *arXiv* preprint arXiv:2308.12950, 2023.
- Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y.
 Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.
- Shen, Z., Tao, T., Ma, L., Neiswanger, W., Liu, Z., Wang, H., Tan, B., Hestness, J., Vassilieva, N., Soboleva, D., et al. Slimpajama-dc: Understanding data combinations for llm training. *arXiv preprint arXiv:2309.10818*, 2023.
- Son, S., Park, W., Han, W., Kim, K., and Lee, J. Prefixing attention sinks can mitigate activation outliers for large language model quantization. *arXiv preprint arXiv:2406.12016*, 2024.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
- Sun, M., Chen, X., Kolter, J. Z., and Liu, Z. Massive activations in large language models. In *First Conference on Language Modeling*, 2024a. URL https: //openreview.net/forum?id=F7aAhfitX6.
- Sun, M., Liu, Z., Bair, A., and Kolter, J. Z. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview. net/forum?id=PxoFut3dWW.
- Team, G., Riviere, M., Pathak, S., Sessa, P. G., Hardin,
 C., Bhupatiraju, S., Hussenot, L., Mesnard, T., Shahriari, B., Ramé, A., et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.
- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi,
 A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P.,
 Bhosale, S., et al. Llama 2: Open foundation and finetuned chat models. *arXiv preprint arXiv:2307.09288*,
 2023.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones,
 L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I.
 Attention is all you need. In Guyon, I., Luxburg, U. V.,
 Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S.,
 and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.,

2017. URL https://proceedings.neurips. cc/paper_files/paper/2017/file/ 3f5ee243547dee91fbd053c1c4a845aa-Paper. pdf.

- Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., and Fergus, R. Regularization of neural networks using dropconnect. In *International conference on machine learning*, pp. 1058– 1066. PMLR, 2013.
- Wang, Z., Jin, B., Yu, Z., and Zhang, M. Model tells you where to merge: Adaptive kv cache merging for llms on long-context tasks. *arXiv preprint arXiv:2407.08454*, 2024.
- Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., and Mann, G. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.
- Xia, H., Yang, Z., Dong, Q., Wang, P., Li, Y., Ge, T., Liu, T., Li, W., and Sui, Z. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Findings of the Association for Computational Linguistics ACL 2024*, pp. 7655–7671, Bangkok, Thailand and virtual meeting, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024. findings-acl.456.
- Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview. net/forum?id=NG7sS51zVF.
- yang Liu, S., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H. DoRA: Weight-decomposed low-rank adaptation. In Fortyfirst International Conference on Machine Learning, 2024. URL https://openreview.net/forum? id=3d5CIRG1n2.
- Yu, M., Wang, D., Shan, Q., and Wan, A. The super weight in large language models. *arXiv preprint arXiv:2411.07191*, 2024a.
- Yu, Z., Wang, Z., Fu, Y., Shi, H., Shaikh, K., and Lin, Y. C. Unveiling and harnessing hidden attention sinks: Enhancing large language models without training through attention calibration. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57659–57677. PMLR, 21– 27 Jul 2024b. URL https://proceedings.mlr. press/v235/yu241.html.

- Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q.,
 Yang, L., et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.
- ⁶¹⁰
 ⁶¹¹ Zhang, Y., Du, Y., Luo, G., Zhong, Y., Zhang, Z., Liu,
 ⁶¹² S., and Ji, R. Cam: Cache merging for memoryefficient LLMs inference. In *Forty-first International Conference on Machine Learning*, 2024. URL https: //openreview.net/forum?id=LCTmppB165.
- 616 Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., 617 Cai, R., Song, Z., Tian, Y., Ré, C., Barrett, C., 618 Wang, Z. A., and Chen, B. H2o: Heavy-hitter oracle 619 for efficient generative inference of large language 620 models. In Oh, A., Naumann, T., Globerson, A., 621 Saenko, K., Hardt, M., and Levine, S. (eds.), Ad-622 vances in Neural Information Processing Systems, 623 volume 36, pp. 34661-34710. Curran Associates, Inc., 624 2023. URL https://proceedings.neurips. 625 cc/paper files/paper/2023/file/ 626 6ceefa7b15572587b78ecfcebb2827f8-Paper-Conference. 627 pdf. 628
- 629 Zhao, Y., Du, L., Ding, X., Xiong, K., Sun, Z., Jun, S., Liu, 630 T., and Qin, B. Deciphering the impact of pretraining data 631 on large language models through machine unlearning. In 632 Ku, L.-W., Martins, A., and Srikumar, V. (eds.), Findings 633 of the Association for Computational Linguistics ACL 634 2024, pp. 9386–9406, Bangkok, Thailand and virtual 635 meeting, August 2024. Association for Computational 636 Linguistics. URL https://aclanthology.org/ 637 2024.findings-acl.559. 638
- ⁶³⁹ Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z.,
 ⁶⁴⁰ Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., Zhang, H.,
 ⁶⁴¹ Gonzalez, J. E., and Stoica, I. Judging LLM-as-a-judge
 ⁶⁴² with MT-bench and chatbot arena. In *Thirty-seventh*⁶⁴³ Conference on Neural Information Processing Systems
 ⁶⁴⁴ Datasets and Benchmarks Track, 2023. URL https:
 ⁶⁴⁵ //openreview.net/forum?id=uccHPGDlao.
- Zheng, Y., Zhang, R., Zhang, J., Ye, Y., and Luo, Z. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.
- 650
- 651
- 652 653
- 654
- 655
- 656
- 657 658
- 659

A. Implementation Details

We use 8xA100-80GB, for our all implementations. As discussed in Section 2.2, we use only the *bos* token to analyze massive activations and massive weights, and design MacDrop. In Section 2.2, we use the example and visualization of massive activations. For parameter-efficient fine-tuning, Llama-Factory is used and configurations are summarized in Table 6. For evaluation, we use the code of massive activations for perplexity, use lm-eval-harness for zero-shot accuracy, and use FastChat and Prometheus for generation tasks. Related papers and codes are cited in the main.

Table 6. Configurat	tion for low rank adaptation (LoRA and DoRA).
Argument	Setting
dataset	alpaca_gpt4_en
validation size	0.05
per device train batch size	8
gradient accumulation steps	4
learning rate	1e-4
num train epochs	3
warmup ratio	0.05
adam β_1	0.9
adam β_2	0.999
lora target	all linear layers except for embedding layer and lm head
lora rank	16
lora alpha	16

B. Position of Massive Weights

Table 7 summarizes the position of massive weights across various models. These are selected based on the magnitudes of intermediate state in Appendix D.

Model	Layer	Top-5 indices
Llama-2-7b-hf	2	[7890, 10411, 1192, 8731, 5843]
Llama-2-7b-chat-hf	2	[7890, 10411, 1192, 8731, 5843]
Llama-2-13b-hf	4	[7678, 8811, 11371, 6619, 12281]
Llama-2-13b-chat-hf	4	[7678, 8811, 11371, 6619, 12281]
Meta-Llama-3-8B	2	[2427, 198, 6412, 12657, 591]
Meta-Llama-3-8B-Instruct	2	[2427, 198, 6412, 591, 12657]
Meta-Llama-3-70B	4	[16581, 3590, 16039, 19670, 13266]
Meta-Llama-3-70B-Instruct	4	[16581, 3590, 16039, 19670, 13266]
Meta-Llama-3.1-405B (8bit)	6	[11891, 30740, 2392, 36238, 12328]
Meta-Llama-3.1-405B-Instruct (8bit)	6	[11891, 30740, 36238, 2392, 1073]
Mistral-7B-v0.1	2	[7310, 8572, 2514, 1878, 8693]
Mistral-7B-Instruct-v0.1	2	[7310, 8572, 2514, 2484, 1878]
Mixtral-8x7B-v0.1	2 (expert 4)	[7310, 7530, 11981, 7492, 3178]
Mixtral-8x7B-Instruct-v0.1	2 (expert 4)	[7310, 11981, 2514, 7530, 3178]
Phi-3-mini-4k-instruct	3	[808, 340, 3644, 2473, 2987]
Phi-3-medium-4k-instruct	6	[181, 7540, 19, 15874, 5137]
gemma-2-2b	2	[1257, 2896, 6954, 8624, 7118]
gemma-2-2b-it	2	[1257, 2896, 6954, 8624, 9140]
gemma-2-9b	1	[2769, 6656, 4889, 14293, 11065]
gemma-2-9b-it	1	[2769, 6656, 4889, 14293, 10429]
gemma-2-27b	10	[34659, 32862, 9590, 8959, 32744]
gemma-2-27b-it	10	[34659, 32862, 9590, 32744, 8959]

Table 7	Laver and	1 indices	of top-5	massive	weights
Table 7.	Layer and	1 muleus	or top-0	massive	weights

C. bos Token Analysis for Various LLMs

In this section, we provide the magnitudes of activations of the hidden state and normalized attention scores according to the position of the *bos* token, after massive activations appear (specifically, in the middle layer), for various LLM families.

C.1. Llama-2 family

Llama-2-7B (Figure 8) has massive activations at the starting token or first delimiter token (first column). When the *bos* token is placed in the starting position, it triggers massive activations and the 'Summer' token loses its massive activations, while first delimiter token '.' still keeps its massive activations (second column). When the *bos* token is placed in the middle or ending position after the first delimiter token, it does not trigger massive activations (third and fourth columns).

Llama-2-13B (Figure 9) has massive activations only at the starting token, other than Llama-2-7B (first column). In cases where the *bos* token is inserted, the same tendencies are observed as with the LLaMA-2-7B model.



Figure 8. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Llama-2-7b-hf.



Figure 9. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Llama-2-13b-hf.

770 C.2. Llama-3 family

Llama-3-8B (Figure 10) does not have massive activations at delimiter tokens such as '.' (first column). When the *bos* token is placed in the starting position, it triggers massive activations and the 'Summer' token loses its massive activations, similar to Llama-2 family (second column). When the *bos* token is placed in the middle or ending position, it also triggers massive activations in the same feature dimensions (third and fourth columns). Namely, the *bos* token has massive activations, regardless of its position. What is intriguing is that, despite the difference in magnitude according to the position, the *bos* token similarly exhibits attention sinks.

Llama-3-70B (Figure 11) generally exhibits similar trends to Llama-3-8B. One notable difference is that the degree of
sinking for the token at the first position is significantly stronger compared to that of the Llama-3-8B.



Figure 10. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Meta-Llama-3-8B.



Figure 11. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Meta-Llama-3-70B.

825 C.3. Mistral and Mixtral family

Mistral (Figure 12) does not exhibit massive activations at the starting position and does not trigger attention sink, contrary to previous findings observed by Sun et al. (2024a). Rather, massive activations are observed only at the first delimiter token (first column). When the *bos* token is placed in the starting position, the first delimiter token loses its massive activations (second column). However, when the *bos* token is placed in the middle or ending position after the first delimiter token, massive activations are observed in both tokens (third and fourth columns). Similar to Llama-3 family, the *bos* token has massive activations, regardless of its position.

Mixtral (Figure 13) exhibits the same behavior as Mistral. The only difference is observed in the magnitude of its massive
 activations, with Mixtral producing values approximately ten times higher than Mistral.



Figure 12. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Mistral-7B-v0.1.



Figure 13. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Mixtral-8x7B-v0.1.

880 C.4. Phi-3 family

Phi-3-mini (Figure 14) and Phi-3-medium (Figure 15) exhibit a similar tendency to Llama-2-13B. This family has massive activations only at the starting token (first column), with a similar response when the *bos* token is inserted (second, third, and fourth column). A significant distinction between the Llama-2-13B model and the Phi-3 family lies in their attention mechanisms. Specifically, the Phi-3 family demonstrates weaker attention on the token at the first position than Llama-2-13B model. This reduced attention appears to be primarily redistributed to recent tokens.



Figure 14. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Phi-3-mini-4k-instruct.



Figure 15. (Top) Magnitudes of the hidden state and (Bottom) attention scores of Phi-3-medium-4k-instruct.

935 C.5. Gemma-2 family

P36
 P37
 P38
 P38
 P39
 P30
 P30
 P31
 P31
 P32
 P31
 P31
 P32
 P31
 P31
 P32
 P31
 P32
 P31
 P32
 P31
 P32
 P31
 P32
 P31
 P31
 P32
 P31
 P31
 P32
 P31
 P31
 P31
 P31
 P31
 P31
 P32
 P31
 P32
 P31
 P32
 P31
 P32
 P32
 P31
 P32
 P31
 P32
 P31
 P32
 P31
 P31
 P32
 P31
 P32
 P32
 P32
 P32
 P32
 P32
 P32
 P33
 P32
 P32
 P32
 P33
 P32
 P33
 P32
 P32
 P33
 P33
 P34
 P34
 P34
 P34
 P34
 P34
 P34
 P34
 P34
 P34

940 Gemma-2-2b (Figure 16) and Gemma-2-9b (Figure 17) do not exhibit noticeably large values along either the token axis or 941 the feature dimension axis, from the perspective of magnitudes of activations (first column, top). This suggests that massive 942 activations are not present. As a result, the attention mechanism avoids the attention sink phenomenon and demonstrates 943 a strong attention on the locality of recent tokens (first column, bottom). However, when the *bos* token is fed into these 944 models, it exhibits massive activations with extremely large values in certain feature dimensions, regardless of its position 945 (second, third, and fourth columns). Nevertheless, compared to other models where attention sinks occur, they allocate 946 significantly greater attention to recent tokens (especially, to its own tokens).



Figure 16. (Top) Magnitudes of the hidden state and (Bottom) attention scores of gemma-2-2b.



Figure 17. (Top) Magnitudes of the hidden state and (Bottom) attention scores of gemma-2-9b.

990 Gemma-2-27b (Figure 18) demonstrates a distinct behavior compared to smaller models. It exhibits noticeably large values 991 along the feature dimension axis across all tokens, from the perspective of magnitudes of activations (first column, top). This 992 distribution, where the differences between tokens are not pronounced, fails to create attention sinks (first column, bottom). 993 When the *bos* token is placed in the starting position, it triggers massive activations and attention sinks by generating value 994 that exceed the magnitudes of other tokens by more than tenfold, in the certain feature dimension (second column). However, 995 when the *bos* token is placed in the middle or ending position, it does not trigger massive activations, similar to when the 996 *bos* token is absent (third and fourth columns).



Figure 18. (Top) Magnitudes of the hidden state and (Bottom) attention scores of gemma-2-27b.

In summary,

- Llama-2, Llama-3, and Phi-3 families have massive activations at the first position.
- Llama-3, Mistral/Mixtral families, and Gemma-2-2/9b models have massive activations at the bos token.
- All families have massive activations at the bos token placed at the first position.

1044

1017 1018 1019

1045 D. Futher Analysis for Various LLMs

We investigate various LLM families: Llama-2 (Touvron et al., 2023), Llama-3 (Dubey et al., 2024), Mistral (Jiang et al., 2023) and Mixtral (Jiang et al., 2024), Phi-3 (Abdin et al., 2024), and Gemma-2 (Team et al., 2024). Similar to Figure 3 in the main, we provide the top-3 and median magnitudes in the hidden states and the intermediate states throughout the layers. In subcaptions, H and I represent the hidden state and the intermediate state, respectively.

When comparing pre-trained LLMs (e.g., Llama-2-7b-hf) and instruction-tuned LLMs (e.g., Llama-2-7b-chat-hf) of the same model, the shape of their graphs is almost identical. This suggests that massive weights are formed during the pre-training process. Llama-2, Llama-3, Mistral, Mixtral, and Phi-3 exhibit similar patterns in their hidden states: following a single explosive amplification in an early layer, massive activations are sustained through residual connections almost until the final layer, although Phi-3 experiences a few additional amplifications. In fact, as we discuss in the main, such an explosion initially occurs in the intermediate state, and this phenomenon is observed across different models. However, the behavior of the Gemma-2 family significantly deviates from that of other models. Firstly, instead of the values being maintained in the hidden state, Gemma-2 shows a continuous increase followed by a decrease. Secondly, the magnitude of the explosion observed in the intermediate state is considerably lower compared to other models. These unique characteristics suggest that Gemma-2 operates under different internal dynamics, which may influence its overall performance and stability.

D.1. Llama-2 family









1265 E. Attention Sinks

1266 Figure 24 describes attention after Softmax in the early layers (from layer 1 to layer 8) across various models. Attention 1267 sinks are observed in the layers after the massive weights layer. In Llama-2-7B (Figure 24(a)), Mistral-7B (Figure 24(c)), 1268 and Mixtral-8x7B (Figure 24(d)), sink tokens are the initial token ('Summer') and the first delimiter token ('.'), discovered 1269 by Sun et al. (2024a). In Llama-3-8B (Figure 24(b)) and Phi-3-mini (Figure 24(e)), sink token is the only the initial token 1270 ('Summer'). Interestingly, in these five models, it is commonly observed that significant attention is concentrated on 1271 non-semantic tokens ('.') before attention sinks occur. However, in Gemma-2 (Figure 24(f)), attention sinks do not happen 1272 and attention is primarily assigned to local tokens. Note that what we provide is the average of the heads, and there might be heads that do not fully sink when viewed individually. 1274



1320 F. Zero-shot Downstream Task across Various LLMs

We address the potential limitations of algorithms based on massive phenomena. Table 8 presents the results on zero-shot downstream tasks across different LLMs. The results show that MacDrop is not effective for LLMs that are not sensitive to massive weights, such as Phi-3-medium and Gemma-2 family, as shown in Section 2.

Nevertheless, we believe that clearly demonstrating such limitations is itself meaningful. Most previous and ongoing research addressing massive phenomena have stated that various models exhibit massive phenomena, and have proposed algorithms and experimental results excluding Gemma-2 family. Because we conduct experiments on a wide variety of recent LLMs, we are able to reveal such limitations. Additionally, we would like to emphasize that, similar to previous studies, MacDrop demonstrates sufficient feasibility for LLMs exhibiting massive phenomena.

Tal	ble 8. Zero-sho	ot downstream	n tasks performand	ce across	differei	nt LLMs.	
Model	Method	ARC-Easy	ARC-Challenge	BoolQ	PIQA	WinoGrande	Avg.
	LoRA	77.1	51.9	82.2	81.6	72.8	73.1
Llama-2-13B	+ MacDrop	78.8	52.6	82.2	71.4	71.9	73.4
	DoRA	77.4	51.8	82.0	81.6	72.8	73.1
	+ MacDrop	77.8	52.5	81.9	81.8	72.3	73.3
	LoRA	79.6	58.2	83.9	82.4	75.9	76.0
I lama-3-8B	+ MacDrop	82.9	58.3	83.9	82.6	75.0	76.5
	DoRA	80.8	57.7	83.9	82.5	75.8	76.1
	+ MacDrop	81.9	58.2	83.9	82.2	75.6	76.4
	LoRA	87.5	66.3	86.1	84.9	80.4	81.0
Llama-3-70B	+ MacDrop	87.5	66.5	86.1	85.0	80.8	81.2
	DoRA	87.5	66.6	86.1	85.0	80.8	81.2
	+MacDrop	87.4	66.6	86.0	85.1	80.9	81.2
	LoRA	78.5	54.9	84.9	82.9	75.3	75.3
Mistral 7B	+ MacDrop	80.9	56.7	85.0	83.0	75.3	76.2
Wilstrai-7D	DoRA	78.4	55.1	85.0	82.9	75.1	75.3
	+ MacDrop	80.6	56.7	85.3	82.9	75.1	76.1
	LoRA	72.5	53.7	86.4	80.1	74.0	73.3
Dhi 3 mini	+ MacDrop	75.0	54.7	86.2	80.5	74.0	74.1
F111-3-111111	DoRA	72.3	53.3	86.4	80.0	73.6	73.1
	+ MacDrop	72.9	53.5	86.3	80.0	73.7	73.3
	LoRA	81.4	62.2	88.7	82.6	76.4	78.3
	+ MacDrop	81.2	61.9	88.7	82.5	76.4	78.1
Phi-3-medium	DoRA	80.92	62.0	88.6	82.4	76.2	78.0
	+MacDrop	80.8	61.9	88.5	82.4	76.2	77.9
	LoRA	81.6	54.4	79.8	79.2	68.7	72.7
Commo 2 24	+MacDrop	81.5	54.2	79.6	79.2	68.6	72.6
Gemma-2-20	DoRA	81.4	54.0	79.6	79.3	68.7	72.6
	+MacDrop	81.6	53.9	79.4	79.3	68.7	72.6
	LoRA	89.6	69.0	86.5	82.8	75.3	80.6
Commo 2 Ob	+ MacDrop	89.2	68.4	86.5	82.8	75.2	80.4
Gemma-2-9b	DoRA	89.4	68.9	86.2	82.7	75.8	80.6
	+MacDrop	89.2	68.5	86.3	82.8	75.7	80.5
	LoRA	87.5	69.0	86.0	84.1	80.6	81.4
Camme 2 27	+MacDrop	87.3	68.8	86.0	84.2	80.5	81.4
Gemma-2-2/b	DoRA	88.5	69.4	85.6	84.5	80.0	81.6
	+ MacDrop	88.1	69.2	85.7	84.3	80.0	81.5

1375 G. Generation Tasks

We evaluate on the generated texts of the same models in Section 4.1 using the Spec-Bench dataset (Xia et al., 2024). This benchmark includes six subtasks, each containing 80 instances: multi-turn (MT) conversation from MT-bench (Zheng et al., 2023), translation from WMT14 DE-EN (Bojar et al., 2014), summarization from CNN/Daily Mail (Nallapati et al., 2016), question answering (QA) from Natural Questions (Kwiatkowski et al., 2019), mathematical reasoning from GSM8K (Cobbe et al., 2021), and retrieval-augmented generation (RAG) from Natural Questions (Kwiatkowski et al., 2019). We utilize the direct assessment of Prometheus-2-7B (Kim et al., 2024) to evaluate the generated texts using a 5-point Likert scale. Prometheus-2-7B is an open-source language model specifically designed for evaluation purposes⁶. Table 9 presents the results on generation tasks. MT-1 and MT-2 indicate the first turn and second turn, respectively. Unfortunately, MacDrop shows limited performance improvements in generation tasks. Table 10 provides examples of the generated texts and judgements. Assistant A and B indicate Llama-3-8B with and without MacDrop, respectively.

Table 9. Generation tasks performance measured by Prometheus-2-7B.

Model	Method	MT-1	MT-2	translation	summarization	QA	math reasoning	RAG	Avg.
Llama 3 8B	LoRA	3.71	3.54	4.50	3.24	4.35	3.64	3.74	3.82
	+ MacDrop	3.76	3.49	4.51	3.39	4.29	3.50	3.71	3.81
Liania-3-8D	DoRA	3.85	3.71	4.56	3.34	4.35	3.59	3.73	3.80
	+ MacDrop	3.79	3.41	4.55	3.26	4.29	3.79	3.85	3.85
Mistral 7B	LoRA	3.55	3.30	4.61	3.33	4.41	3.45	4.05	3.81
	+ MacDrop	3.75	3.42	4.58	3.26	4.35	3.30	3.98	3.81
iviisitai-7D	DoRA	3.55	3.29	4.64	3.34	4.49	3.35	3.88	3.79
	+ MacDrop	3.49	3.29	4.59	3.52	4.41	3.33	3.95	3.80

Table 10. Example of generation task (translation).

User	Jäger, der durch eine Vision bekehrt und später zum Bischof von Lüttich ernannt wurde.
Reference answer	St. Hubert was, according to legend, a truly ruthless hunter, who converted following a vision and was later appointed Bishop of Lüttich.
Assistant A	According to legend, St. Hubertus was a rather ruthless hunter who was converted and later appointed as the Bishop of Liège after a vision.
Judgment for A	The response provided a translation that was generally accurate and relevant to the user's request. However, there were minor inaccuracies in the phrasing, such as "rather ruthless" instead of "truly ruthless," which slightly deviated from the original text. Additionally, the term "Liège" was used instead of "Lüttich," which is the correct name of the city in the original German text. Despite these minor issues, the response still conveyed the main points of the legend about St. Hubertus. Therefore, while the response was helpful and mostly aligned with the user's inquiry, it did not perfectly cater to the user's needs due to the slight inaccuracies in the translation. So the overall score is 4.
Assistant B	According to legend, St. Hubertus was a rather reckless hunter who was converted and later appointed as the Bishop of Liège after a vision.
Judgment for B	The response provided a translation that was generally accurate and relevant to the user's request. However, there were slight deviations from the original text. For instance, the term "rather reckless" is less precise than "truly ruthless," which could lead to some ambiguity. Additionally, the term "Liège" is used instead of "Lüttich," which is a minor inaccuracy. Despite these minor issues, the response still managed to convey the main points of the original text, making it useful for the user's needs. Therefore, the response aligns well with the user's inquiry, with only rare inaccuracies, and thus meets the criteria for a score of 4.

1428 ⁶https://github.com/prometheus-eval/prometheus-eval

1430 H. Related work

¹⁴³¹ 1432 The attention sinks phenomenon and their importance, uncovered by Xiao et al. (2024), have been widely used to compress

key-value caches. For quantization, KVQuant (Hooper et al., 2024) applies attention sink-aware quantization, which retains

only the first token in fp16. CushionCache (Son et al., 2024) inserts sink tokens into the prefix to mitigate massive activations

¹⁴³⁵ in the middle of the sequence, enhancing the performance of quantized models. For token eviction and token merging, sink tokens are never evicted or merged; they remain unchanged (Xiao et al., 2024; Ge et al., 2024; Li et al., 2024; Zhang et al.,

tokens are never evicted or merged; they rem
2023; Wang et al., 2024; Zhang et al., 2024).

1438 Nevertheless, there has been limited in-depth research on the phenomenon itself. In fact, the idea of global attentions, such as 1439 [CLS] and [SEP] tokens-similar to attention sinks-was introduced and emphasized even before the LLM era (Zaheer et al., 1440 2020; Beltagy et al., 2020). In the LLM era, Yu et al. (2024b) showed that sink tokens can appear not only at the beginning 1441 of a sentence but also in the middle, and they are often shown to be nonsemantic (e.g., '.'). Sun et al. (2024a) discovered the 1442 presence of massive activations in the hidden state space of sink tokens, demonstrating that massive activations trigger the 1443 attention sinks phenomenon. Meanwhile, in vision transformers, similar phenomenon is observed (Darcet et al., 2024). They 1444 showed that training with register tokens, which is additional meaningless tokens similar to sink tokens, resulted in improved 1445 dense prediction and interpretability. Different from previous work, we explore this phenomenon in the weight space.

Specifically, we define the massive weights in an activation-aware manner using the *bos* token. Similarly, Wanda (Sun et al., 2024b) with a structured pruning (An et al., 2024) and AWQ (Lin et al., 2024) calculate weight importance scores based on a small of calibration data. However, it is important to note that the massive weights are confined to a specific single layer,

whereas Wanda and AWQ identify important weights within every linear layer. In other words, the massive weights would
 be included among those selected through Wanda or AWQ. Our contribution focuses more deeply on a narrowly defined
 aspect compared to these studies.

1453 Concurrently with our work, Yu et al. (2024a) have defined "super weight," which is a single weight that can significantly 1454 influence the performance of LLMs. To compare more specifically, we identify that the intermediate state in the layer l1455 possesses massive activations. Accordingly, we define massive weight as the corresponding rows of W_{up} . However, Yu 1456 et al. (2024a) take into account both the intermediate state and the subsequent hidden state. In other words, they define 1457 super weight as the intersection of the corresponding column of W_{down} to the super activation in the intermediate state and 1458 the corresponding row of W_{down} to the super activation in the hidden state, at the layer l. In summary, we examine the 1459 intermediate state exclusively from an output perspective (i.e., W_{up}), while their approach appears to have considered it 1460 from an input perspective (i.e., W_{down}), incorporating the hidden state. 1461

1462 1463

1464 1465