

TWEO: Transformers Without Extreme Outliers Enables FP8 Training And Quantization For Dummies

Supplementary Material

001 A. Empirical Analysis of Data-Independent 002 Outliers

003 To provide a solid empirical foundation for the *Contra-*
004 *diction Stethoscope* presented in the main text (Sec. 1
005 of the main paper), i.e., that extreme outliers are *data-*
006 *independent, mechanically-produced artifacts* of training,
007 this appendix presents detailed ablation studies. Concretely,
008 the stethoscope reports layer-wise Avg Max, Avg Min, and
009 Mean-Abs statistics, averaged over the batch.

010 A.1. Ablation Study on Qwen2.5-0.5B (Language 011 Model)

012 For the language model, we designed four experimental
013 combinations (see Figure 1) to decouple the influence of
014 pre-trained weights, the embedding layer, and the input data
015 on outlier generation. The results clearly demonstrate:

- 016 • *Product of Training.* Comparing (a) and (c), extreme out-
017 liers are only absent (Avg Max < 10) when both the
018 model weights and the embedding layer are randomly ini-
019 tialized. The moment pre-trained weights are used, out-
020 liers emerge immediately.
- 021 • *Product of Weights.* Comparing (a) and (b), panel (b)
022 shows that even if the embedding layer is re-initialized
023 randomly, as long as the Transformer Block weights are
024 pre-trained, extreme outliers (~1600) are still mechani-
025 cally reproduced.
- 026 • *Data-Independence.* Comparing (c) and (d), the pre-
027 trained model produces outliers of the same magnitude
028 regardless of whether the input is real data or random to-
029 kens.

030 *Conclusion.* Extreme outliers are a *mechanical product* of
031 the pre-trained Transformer Block weights and are indepen-
032 dent of both the embedding layer and the input data.

033 A.2. Statistical Evidence on Data Independence

034 To further substantiate the data-independent nature of ex-
035 treme outliers, we extend the analysis to Qwen2.5-0.5B by
036 comparing real text inputs against random token inputs. We
037 collect statistics from the per-sample maximum activation
038 over $N = 10000$ samples. The results show that extreme
039 outliers emerge consistently under both settings, indicating
040 that the phenomenon persists even when semantic structure
041 is removed from the input.

042 As shown in Table 1, Qwen2.5-0.5B produces extreme
043 outliers far beyond the FP8 range under both real text and
044 random token inputs. The comparable maxima and aver-
045 age maxima indicate that semantic structure is not required

Table 1. Statistical comparison of extreme outliers on Qwen2.5-0.5B under real text and random token inputs. Statistics are computed from the per-sample maximum activation over $N = 10000$ samples.

Input Type	Max	Min	Avg Max
Real text	1880.0	1464.0	1690.4
Random tokens	1928.0	1264.0	1622.0

for the emergence of these outliers. This statistical evidence
further supports our claim that extreme outliers are mechani-
cally produced by the trained weight structure, rather than
being triggered by specific data patterns.

A.3. Ablation Study on ViT-B (Vision Model)

To demonstrate that this finding holds true in the vision
modality, we conducted parallel experiments on the ViT-B
model. Figure 2 shows the input samples used, including
real ImageNet images and randomly generated noise. Fig-
ure 3 displays the corresponding activation results.

The conclusions are identical to those from the Qwen
model:

- *Product of Training.* Comparing (c) and (d), the randomly
initialized ViT-B model (c) produces no extreme outliers
(Avg Max < 7).
- *Data-Independence.* The comparison between (d) and (e)
proves that the pre-trained ViT-B model consistently re-
produces extreme outliers (Avg Max ~1500), regardless
of whether the input is real images (Fig. 2(a), (b)) or ran-
dom noise (Fig. 2(c), (d)).

Conclusion. Our core finding—that outliers are data-
independent and a product of training—is cross-modally
general.

B. Ablation Study on Key TWEO Hyperpa- rameters

This appendix details the impact of TWEO’s two key
hyperparameters—the threshold τ and the penalty power
 p —on model performance, outlier suppression efficacy, and
downstream quantization performance.

B.1. Impact of Scaling Factor (τ) on Outlier Sup- pression

We first investigate the effect of different scaling factor set-
tings (which act as a soft threshold) on the suppression of
outliers during training. As shown in Table 2, we fix the

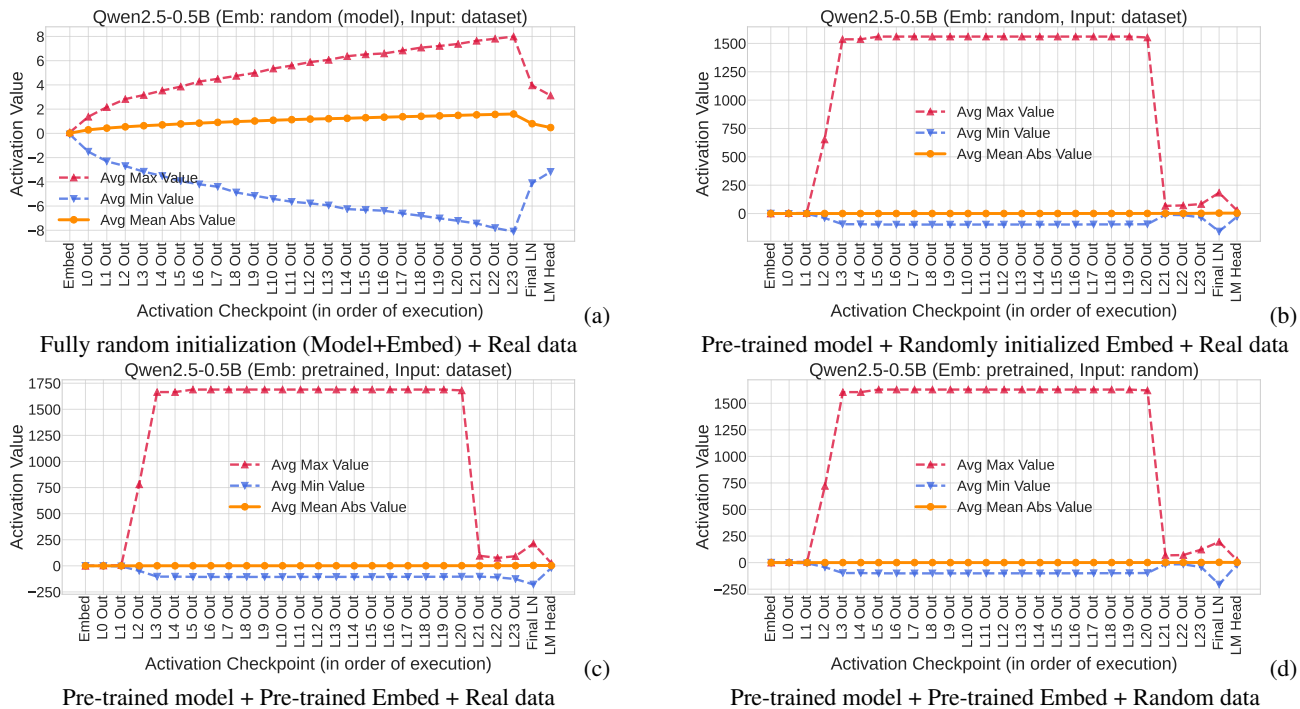


Figure 1. Overview of Qwen2.5-0.5B ablation study. (a) Fully random initialization (Model+Embed) + Real data. (b) Pre-trained model + Randomly initialized Embed + Real data. (c) Pre-trained model + Pre-trained Embed + Real data. (d) Pre-trained model + Pre-trained Embed + Random data.

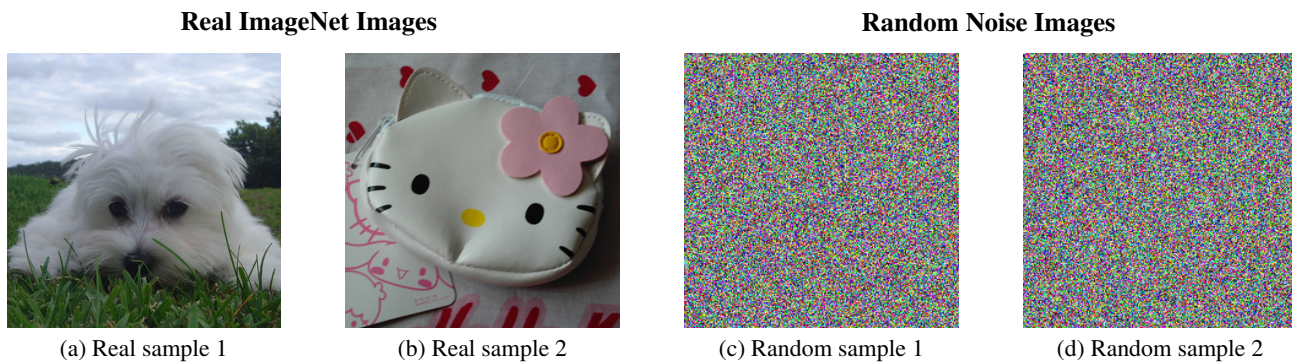


Figure 2. ViT-B input samples. The samples are clearly divided into: real ImageNet images ((a), (b)) and randomly generated noise images ((c), (d)). These examples are used to demonstrate the data-independent nature of outlier generation.

080 penalty power ($p = 4$, the value used in our main paper)
081 and vary the scaling factor τ .

082 Table 2 clearly shows that a lower threshold τ results
083 in stronger outlier suppression (i.e., smaller peak magni-
084 tudes during and after training). We found that for the GPT2
085 model, a lower threshold (e.g., $\tau = 1$) with its strong regu-
086 larization had little impact on model performance (PPL).
087 However, for the Swin Transformer series, an excessively
088 low threshold ($\tau = 1$) slightly impaired model accuracy.
089 Therefore, the experiments reported in our main text use

$\tau = 3$ as a balanced choice.

090

B.2. Impact of Penalty Power (p) on Outlier Sup- pression

091

092

Next, we investigate the impact of the penalty power p on
model performance. As shown in Table 3, we fix the thresh-
old ($\tau = 3$, the value used in our main paper) and vary the
penalty power p .

093

094

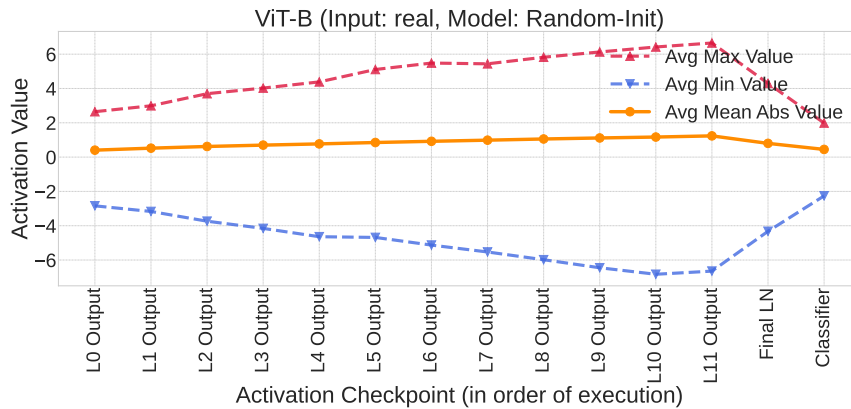
095

096

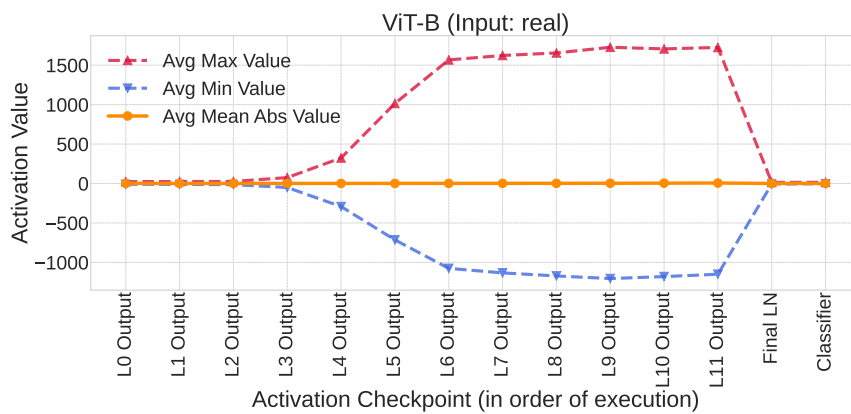
As seen in Table 3, a higher penalty power p leads to bet-
ter outlier suppression. We found that for GPT2, stronger

097

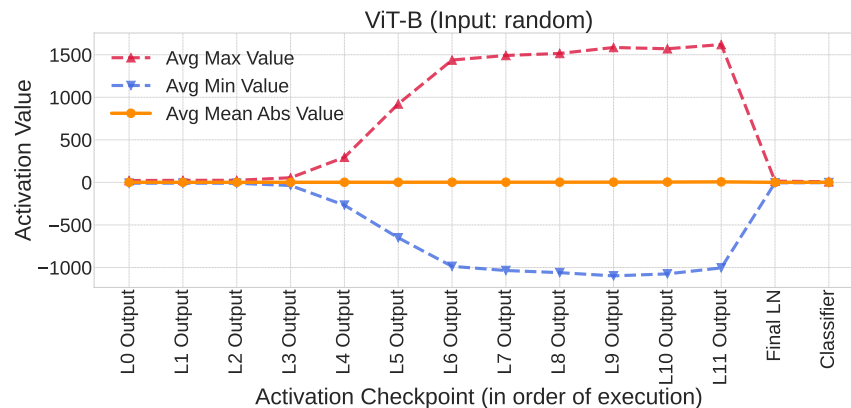
098



(c) Randomly initialized model + Real images



(d) Pre-trained model + Real images (Input a/b)



(e) Pre-trained model + Random noise

Figure 3. Data-independence results for ViT-B. (c) Randomly initialized model + Real images. (d) Pre-trained model + Real images. (e) Pre-trained model + Random noise.

099 regularization ($p = 5$) did not significantly affect model
 100 performance. However, for the Swin series, excessively
 101 strong regularization ($p = 5$) slightly impaired performance.
 102 Therefore, the experiments reported in our main
 103 text uniformly use $p = 4$.

B.3. Combined Impact of Threshold and Power on Quantization Performance

Finally, we conducted a more detailed ablation study on the GPT2 model to investigate the combined impact of differ-

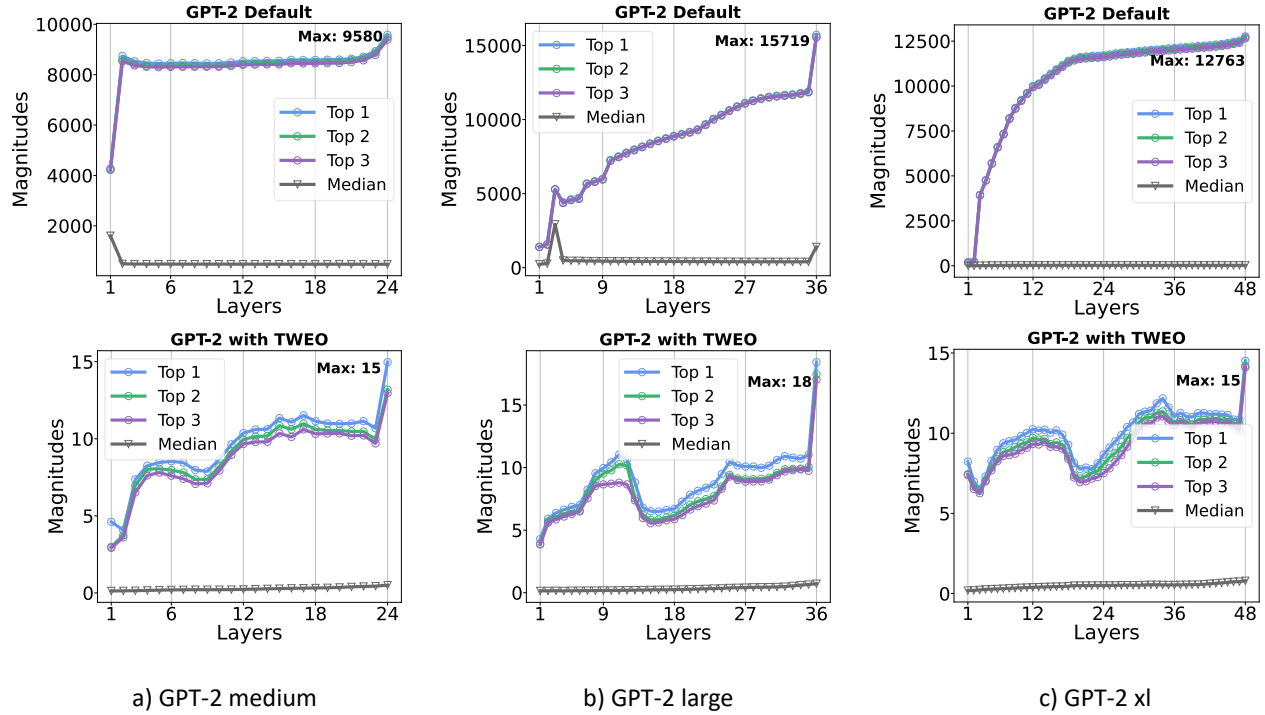


Figure 4. Comparison of activation magnitudes during GPT-2 training. Top 1 refers to the largest activation value in each layer. Top 2, Top 3 refer to the second and third largest. Median is the median of all activations.

Table 2. Ablation study on the effect of different TWEO scaling factors (τ) on outlier suppression.

Model	Threshold (τ)	Accuracy (%)	Peak	Final
<i>ImageNet Accuracy</i>				
Swin-T [†]	–	81.2	1556	534
Swin-T	1	80.6	6	6
Swin-T	2	81.3	11	8
Swin-T [#]	3	81.4	22	15
Swin-T	5	81.4	34	24
<i>ImageNet Accuracy</i>				
Swin-S [†]	–	82.7	6402	1758
Swin-S	1	82.5	10	5
Swin-S	2	82.6	15	7
Swin-S [#]	3	82.8	22	10
Swin-S	5	83.2	35	21
Swin-S	10	83.1	62	31
<i>Wikitext-2 Perplexity (PPL)</i>				
GPT2 [†]	–	20.04	823	563
GPT2	1	18.68	6	5
GPT2 [#]	3	18.68	16	15
GPT2	5	18.89	28	28

[†] Baseline model trained without TWEO.

[#] Hyperparameter setting used in the main paper.

Table 3. Impact of different TWEO penalty powers (p) on model performance and outlier suppression.

Model	Power (p)	Accuracy (%)	Peak	Final
<i>ImageNet Accuracy</i>				
Swin-T [†]	–	81.2	1556	534
Swin-T	3	81.2	48	24
Swin-T [#]	4	81.4	22	15
Swin-T	5	81.1	12	10
<i>ImageNet Accuracy</i>				
Swin-S [†]	–	82.7	6402	1758
Swin-S	2	82.3	292	127
Swin-S	3	82.4	80	48
Swin-S [#]	4	82.8	22	10
Swin-S	5	82.7	15	8
<i>Wikitext-2 Perplexity (PPL)</i>				
GPT2 [†]	–	20.04	823	563
GPT2	2	18.77	158	113
GPT2 [#]	4	18.68	16	15
GPT2	5	19.07	10	10

[†] Baseline model trained without TWEO.

[#] Hyperparameter setting used in the main paper.

ent threshold and power combinations on downstream 8-bit quantization performance.

The ablation study in Table 4 shows that stronger outlier suppression from TWEO (e.g., $\tau = 1, p = 4$) leads to better downstream quantization performance (especially for per-tensor quantization). It is worth noting that the hyperparameters adopted in our main text ($\tau = 3, p = 4$) are not the optimal choice for downstream quantization (compared to $\tau = 1, p = 4$). This indicates that the choice of TWEO hyperparameters involves a trade-off between model pre-training performance and downstream quantization performance. Future work focused on quantization can leverage our TWEO method with more aggressive (i.e., stronger suppression) hyperparameters, such as $\tau = 1, p = 4$, to achieve superior quantization results.

123 C. OpenWebText Pre-training Hyperparameters 124

125 We pre-trained a series of GPT-2 [3] models from scratch
126 on the OpenWebText dataset. The training used a total
127 batch size simulating 1M tokens and was run for 100k steps,
128 equivalent to 100B tokens of training. The training hyper-
129 parameters largely follow those of GPT-3 [1]: for the 3B
130 and 7B models, we adopted configurations similar to GPT-
131 3; the other smaller models (124M to 1.5B) maintain con-
132 figurations similar to the standard GPT-2. For GPT-2 124M
133 to Large, a 0.5M batch size for 100k steps is equivalent to
134 50B tokens; for XL, 3B, and 7B, a 1M batch size for 100k
135 steps is equivalent to 100B tokens. Detailed configurations
136 are shown in Table 5.

137 D. Comparison with SmoothQuant: Quantizing the Residual Stream 138

139 As discussed in the main text (Sec. 4.3 of the main paper),
140 a core advantage of the TWEO paradigm is its resolution of
141 the fundamental bottleneck in the old paradigm: the inability
142 to quantize the residual stream ($x = x + f(x)$). In old
143 paradigm models, extreme outliers are passed and accumu-
144 lated layer by layer in the residual stream x . Consequently,
145 previous methods (like SmoothQuant) are forced to add the
146 quantized $f(x)$ to a high-precision x , incurring significant
147 inference overhead. Due to space constraints, the main text
148 (Table 4 of the main paper) only presented results for GPT-
149 2 XL. This section provides the complete experimental data
150 (Table 6), comparing the full performance of Default and
151 TWEO models when the residual stream is quantized.

152 **Full Analysis** The complete results in Table 6 are decisive
153 and validate the claims in the main text:

- 154 • *SmoothQuant Fails in a Fair Comparison.* When
155 SmoothQuant is forced to quantize the residual stream

(Yes), its performance on all Default models *completely collapses* (e.g., Medium PPL 1510.17, Large PPL 3751.66). This confirms that outliers accumulated in the residual stream are the fundamental obstacle to true full-model quantization.

- 161 • *TWEO Makes Residual Quantization Possible.* Con-
162 versely, the TWEO models, even when using the simplest
163 AbsMax and quantizing the residual stream, show only a
164 minor drop in PPL. This performance far exceeds the De-
165 fault + SmoothQuant combination that *does not* quantize
166 the residual.
- 167 • *TWEO Renders Complexity Obsolete.* More interest-
168 ingly, on the outlier-free models created by TWEO, ap-
169 plying the complex SmoothQuant method yields marginal
170 gains compared to simple AbsMax, making the trade-
171 off in complexity unjustifiable. This proves that TWEO
172 has leveled the quantization difficulty of activations and
173 weights, rendering complex difficulty transfer methods
174 like SmoothQuant unnecessary.

175 E. Training Overhead and FP8 Acceleration 176 Analysis

To evaluate the additional computational overhead intro-
duced by TWEO and its resulting performance advantages,
we recorded the training resource consumption for vision
and language models under different settings.

As shown in Table 7, the additional training overhead
from TWEO is minimal for vision models (Swin Trans-
former). On Swin-S (50M) and Swin-B (88M) models, the
total training time increased by only 3.3% and 3.0%, re-
spectively, while VRAM usage remained unchanged.

186 E.1. Vision FP8 Training Results

To complement the main-text results on outlier suppression
and quantization, we further evaluate whether TWEO en-
ables stable native FP8 training in vision Transformers. We
conduct experiments on Swin models under the same na-
tive FP8 setup used in our language experiments. The re-
sults show that standard FP8 training suffers severe col-
lapse, whereas TWEO stabilizes optimization and recovers
near-BF16 accuracy.

As shown in Table 8, native FP8 training without TWEO
leads to substantial accuracy degradation on both Swin-T
and Swin-B. In contrast, TWEO reduces the peak activa-
tion magnitude to a small and stable range, allowing FP8
training to achieve accuracy nearly identical to the BF16
baseline. This result confirms that TWEO is effective not
only for language models, but also for vision Transformers
under native FP8 training.

Table 4. Combined impact of different TWEO threshold (τ) and power (p) combinations on GPT2 quantization performance (PPL \downarrow).

Method	τ	p	Full Precision	Activation Only 8-bit (A8)	Weight Only 8-bit (W8)	Full 8-bit Quant (W8A8)	
			(BF16) PPL	per-tensor / per-token	per-tensor / per-channel	W8(T)A8(T) / W8(T)A8(K)	W8(C)A8(T) / W8(C)A8(K)
Default [†]	-	-	20.04	82.94 / 21.19	20.71 / 20.32	86.60 / 22.02	
TWEO	1	4	18.68	19.64 / 18.81	19.17 / 18.94	20.19 / 19.31	
TWEO [#]	3	4	18.83	20.21 / 19.00	19.32 / 19.11	20.82 / 19.51	
TWEO	3	2	18.77	137.62 / 19.35	19.73 / 19.32	158.15 / 20.42	
TWEO	3	3	18.74	21.48 / 18.95	19.35 / 19.10	22.37 / 19.57	
TWEO [#]	3	4	18.83	20.21 / 19.00	19.32 / 19.11	20.82 / 19.51	

[†] Baseline model trained without TWEO.

[#] Hyperparameter setting used in the main paper.

Table 5. Detailed OpenWebText pre-training hyperparameters. We set the corresponding number of layers (n_{layers}), model dimension (d_{model}), number of attention heads (n_{head}), head dimension (d_{head}), Batch Size, and Learning Rate for GPT-2 models of different scales (124M to 7B).

Model	Params	n_{layers}	d_{model}	n_{head}	d_{head}	Batch Size	Learning Rate
GPT-2	124M	12	768	12	64	0.5M	6.0e-4
GPT-2 Medium	350M	24	1024	16	64	0.5M	6.0e-4
GPT-2 Large	774M	36	1280	20	64	0.5M	3.0e-4
GPT-2 XL	1.5B	48	1600	25	64	1M	2.0e-4
GPT-2 3B	2.7B	32	2560	32	80	1M	1.6e-4
GPT-2 7B	6.7B	32	4096	32	128	1M	1.2e-4

Table 6. Comparison with SmoothQuant on quantizing the Residual Stream ($x = x + f(x)$). This experiment demonstrates that TWEO enables effective quantization of the residual stream for the first time, whereas old paradigm models collapse completely under this setting.

Model PPL (BF16)	Method	Res.?	W8(C)A8(T) / W8(C)A8(K)
GPT-2-M Default (PPL: 16.77)	SmoothQuant	No	19.76 / 17.28
	SmoothQuant	Yes	1510.17 / 19.23
GPT-2-M TWEO (PPL: 15.18)	AbsMax	Yes	16.32 / 15.40
	SmoothQuant	Yes	16.09 / 15.34
GPT-2-L Default (PPL: 14.92)	SmoothQuant	No	15.70 / 15.06
	SmoothQuant	Yes	3751.66 / 20.57
GPT-2-L TWEO (PPL: 13.79)	AbsMax	Yes	18.39 / 14.07
	SmoothQuant	Yes	18.24 / 14.03
GPT-2-XL Default (PPL: 13.84)	SmoothQuant	No	14.81 / 14.01
	SmoothQuant	Yes	1876.70 / 21.93
GPT-2-XL TWEO (PPL: 12.77)	AbsMax	Yes	13.06 / 12.63
	SmoothQuant	Yes	12.89 / 12.51

W8(C)/A8(T): Weights per-channel, Activations per-tensor

W8(C)/A8(K): Weights per-channel, Activations per-token

203
204

E.2. Comparison with a DeepSeek-Style FP8 Strategy

205
206

We further compare TWEO with a DeepSeek-style FP8 strategy from the perspective of training efficiency. The

Table 7. Training resource consumption comparison for vision models (Swin Transformer) on 4090 GPUs.

Model	Method	VRAM (GB)	Total Time (300epoch)	8-GPU Time
Swin-S (50M)	Default	18.0G	240 H	30H
	TWEO	18.0G	248H (+3.3%)	31H
Swin-B (88M)	Default	23.5G	300 H	37.5H
	TWEO	23.5G	309H (+3.0%)	38.6H

Table 8. Vision FP8 training results on ImageNet. Standard native FP8 training collapses, while TWEO recovers near-BF16 performance.

Model	Top-1 (BF16)	Top-1 (FP8)	Peak	Gap
Swin-T	81.2	50.2 (<i>Collapse</i>)	>1500	-30.9
Swin-T + TWEO	81.4	81.2	32	-0.2
Swin-B	83.5	61.9 (<i>Collapse</i>)	>7000	-21.6
Swin-B + TWEO	83.5	83.4	47	-0.1

DeepSeek-style setup adopts mixed precision and fine-grained per-tile scaling to bypass outlier-sensitive components, whereas TWEO enables a much simpler full-layer per-tensor FP8 configuration by structurally suppressing outliers. For a fair comparison, both methods are implemented with the same Transformer Engine backend and

207
208
209
210
211
212

Table 9. Comparison with a DeepSeek-style FP8 strategy on Swin models. TWEO achieves higher throughput and lower memory usage under the same Transformer Engine backend.

Model	Method	Top-1 (%)	Memory	Speed	Speedup
Swin-T	DeepSeek-style FP8	81.2	20G	1640	+2.5%
	TWEO FP8	81.2	15G	2508	+39%
Swin-S	DeepSeek-style FP8	82.7	34G	1302	+5.0%
	TWEO FP8	82.8	27G	1654	+34%

Table 10. Training resource consumption and FP8 acceleration effects for language models (GPT-2) on H800 GPUs. XL and 3B models were trained on 8 NVIDIA H800s; the 7B model was trained on 16 NVIDIA H800s. All experiments used DeepSpeed ZeRO-3 optimization.

Model	Method	VRAM (GB)	Total Time (H)	iter/H
GPT-2 XL	Default	57	1210	661
	TWEO	57	1219 (+1%)	656 (-1%)
	FP8	55	1000 (-17%)	800 (+21%)
GPT-2 3B	Default	63	1597	500
	TWEO	63	1628 (+2%)	490 (-2%)
	FP8	60	1228 (-23%)	650 (+30%)
GPT-2 7B	Default	56	3696	432
	TWEO	56	3774 (+2%)	423 (-2%)
	FP8	52	2715 (-27%)	588 (+36%)

213 evaluated on 8 NVIDIA H800 GPUs.

214 Table 9 shows that TWEO consistently matches or
215 slightly improves accuracy while substantially reducing
216 memory usage and increasing training throughput. We at-
217 tribute this advantage to TWEO’s ability to remove ex-
218 treme outliers at the source, thereby avoiding the block-
219 wise switching and frequent cast operations required by
220 fine-grained mixed-precision strategies. These results fur-
221 ther support TWEO as a simple yet hardware-friendly solu-
222 tion for practical FP8 training.

223 As shown in Table 10, which records the total train-
224 ing time (in hours) for **100k iterations**, for large language
225 models (GPT-2), the computational overhead of the TWEO
226 method itself is essentially negligible (showing a minor 1-
227 2% increase in training time compared to the default BF16
228 baseline). More importantly, by fundamentally resolving
229 extreme outliers, TWEO makes the previously highly un-
230 stable FP8 training both feasible and efficient. By di-
231 rectly enabling the NVIDIA Transformer Engine without
232 any additional optimization, FP8 training achieved signifi-
233 cant speedups on the GPT-2 XL, 3B, and 7B models: train-
234 ing time was reduced by 17%, 23%, and 27%, respectively,
235 and training throughput (iter/H) correspondingly increased
236 by 21%, 30%, and 36%.

Table 11. TWEO’s impact on Vision Transformer PTQ. All quan-
tization uses *static AbsMax quantization with per-tensor activa-
tions (W(C)A(T))*. The table clearly shows baseline models col-
lapsed at W6A6, while TWEO successfully recovered accuracy to
highly competitive levels.

Model	Method	W32A32 (%)	W8A8 (%)	W6A6 (%)
ViT-B (87M)	Default	81.32	79.86	7.41
	+TWEO (Ours)	81.22	80.29	66.37
Swin-T (28M)	Default	81.24	77.69	0.19
	+TWEO (Ours)	81.40	80.94	51.69
Swin-S (50M)	Default	82.74	80.22	0.13
	+TWEO (Ours)	82.79	82.55	77.27
Swin-B (88M)	Default	83.53	40.16	0.10
	+TWEO (Ours)	83.41	83.08	80.77

F. Vision Transformer PTQ Results

237 To complement the main-text language quantization exper-
238 iments, we report PTQ results on vision Transformers here.
239 We quantize ViT and Swin models with static AbsMax
240 quantization, using per-channel quantization for weights
241 and the most outlier-sensitive *per-tensor* scheme for acti-
242 vations (W(C)A(T)).

243 Results in Table 11 again validate TWEO’s general-
244 ity. In the W8A8 setting, TWEO-trained models signifi-
245 cantly outperformed the baselines. In the more aggressive
246 W6A6 setting, the baseline models completely collapsed
247 (e.g., Swin-T at 0.19%, ViT-B at 7.41% accuracy), whereas
248 TWEO-trained models successfully recovered the accuracy
249 to highly competitive levels (Swin-S at 77.27%, Swin-B at
250 80.77%). This shows that TWEO’s outlier suppression ben-
251 efits downstream quantization not only for language mod-
252 els, but also for vision Transformers.

G. Vision Generation Models: DiT Experiments

254 To further verify TWEO’s cross-modal generality men-
255 tioned in the abstract, we applied it to a vision generation
256 task using the DiT (Diffusion Transformer) [2] model. We
257 evaluated TWEO’s impact on generation quality (FID).
258

259 As shown in Table 13, we applied TWEO to a series of
260 DiT models (from DiT-S/2 to DiT-XL/2). The training setup
261 was uniformly $\tau = 30, p = 4, \lambda = 0.01$. The results show
262 that compared to the baseline (Baseline), the FID scores
263 of the TWEO-applied models (+TWEO) increased slightly
264 (e.g., from 9.62 to 9.88 for DiT-XL/2 at 256x256 resolu-
265 tion). This suggests a slight trade-off may exist between
266 suppressing outliers with TWEO and the final generation
267 quality in complex generative tasks. Nonetheless, this val-
268 idates that the TWEO loss function can be plug-and-play
269

Table 12. Comparison of 8-bit quantization performance (PPL \downarrow) on large-scale GPT-2 models (3B and 7B). **A8**: Activation Quantization, **W8**: Weight Quantization. (T): per-tensor, (C): per-channel, (K): per-token.

Model	Method	Full Precision	Activation Only 8-bit (A8)	Weight Only 8-bit (W8)	Full 8-bit Quant (W8A8)	
		(BF16) PPL	per-tensor / per-token	per-tensor / per-channel	W8(T)A8(T) / W8(T)A8(K)	W8(C)A8(T) / W8(C)A8(K)
GPT-2 3B	Default	12.67	72674.75 / 17.54	12.79 / 12.70	55252.88 / 17.38	75394.51 / 17.56
	TWEO	12.24	12.71 / 12.28	12.29 / 12.27	12.76 / 12.33	12.73 / 12.32
GPT-2 7B	Default	12.49	6117.04 / 414.28	12.77 / 12.67	5463.33 / 386.15	5821.13 / 417.67
	TWEO	12.02	12.93 / 12.18	12.03 / 12.01	12.93 / 12.18	12.92 / 12.17

Table 13. Generative task performance comparison on DiT models. Training setup: $\tau = 30, p = 4, \lambda = 0.01$. Max Value is the maximum outlier encountered during 250-step generation (no CFG).

Model	# param (M)	Resolution	FID \downarrow	Max Value \downarrow
DiT-S/2	33	256	68.40	8632
+TWEO	33	256	73.21	569
DiT-B/2	130	256	43.47	26782
+TWEO	130	256	44.58	1223
DiT-L/2	458	256	23.33	56364
+TWEO	458	256	23.92	1856
DiT-XL/2	675	256	9.62	40991
+TWEO	675	256	9.88	2203
DiT-XL/2	675	512	11.93	16641
+TWEO	675	512	12.35	1859

270 applied to generative Transformers and complete training
 271 successfully, opening up possibilities for exploring stable
 272 low-bit training in generative models in the future.

273 H. Extended Quantization Results on Larger 274 Models

275 Due to space limitations in the main text, we only re-
 276 ported quantization results for smaller models or focused
 277 on specific ablation studies. In this section, we present the
 278 comprehensive post-training quantization (PTQ) results for
 279 larger scale models, specifically GPT-2 3B and GPT-2 7B.

280 Table 12 compares the performance of the Default base-
 281 line and our TWEO model across various 8-bit quantization
 282 settings. Consistent with our findings on smaller models,
 283 the Default 3B and 7B models suffer from catastrophic col-
 284 lapse (e.g., PPL $> 10^4$) under activation quantization due to
 285 the presence of extreme outliers. In contrast, TWEO models
 286 successfully suppress these outliers during pre-training, en-
 287 abling the models to maintain near-BF16 performance even
 288 with naive 8-bit quantization methods, without requiring
 289 complex post-training compensation techniques.

References 290

- [1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020. 291
 292
 293
 294
 295
 296
 [2] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 297
 298
 299
 300
 [3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9, 2019. 301
 302
 303
 304