A TRAINING DETAILS

In this section, we provide details on the training of our models.

In all experiments we use a batch size of 512 and in all experiments using Adam or OrthoAdam as the optimiser, we use a peak learning rate of 10^{-3}. This excludes the experiments in Section 5.3 which use SGD as the optimiser, which use a peak learning rate of 0.2. In all experiments we use a cosine learning rate schedule with linear warmup for $\{1000, 2000, 6000, 10000\}$ steps for models with $\{$60M, 130M, 350M, 1.4B$\}$ parameters respectively. Note that we use a reduced number of steps for the 1.4B model due to computational constraints. For the main experimental results in Tables 2 and 3, we train the models with $\{$60M, 130M, 350M, 1.4B$\}$ parameters for $\{$160k, 320k, 960k, 600k$\}$ steps respectively. For the ablation study in Section 5.3, we train GPT2 models with 130M parameters for 40k steps with 2000 warmup steps. We use a maximum sequence length of 256 tokens, which we find is sufficient to observe the anomalies of first token attention dominance and large outlier activations found in popular pretrained models such as GPT2 (Radford et al., 2019) and Llama (Touvron et al., 2023a;b; Dubey et al., 2024). The result of our training setup is that models trained for the main experimental results with $\{$60M, 130M, 350M, 1.4B$\}$ parameters are trained on $\{$21B, 42B, 126B, 79B$\}$ tokens respectively. The ablation experiments are trained on 5B tokens. We train models on 8 NVIDIA 32GB V100 GPUs using the Pytorch deep-learning framework (Paszke et al., 2019) and the HuggingFace Transformers library (Wolf et al., 2020).

B LONGER SEQUENCE TRAINING

Model Size	Setup	Full	Coarse	Δ (Coarse)	Moderate	Δ (Moderate)	Fine	Δ (Fine)
60M-256	Vanilla	31.88	43.53	11.65	34.87	2.99	32.15	0.27
60M-512	Vanilla	32.66	48.55	15.89	37.24	4.58	33.07	0.41
60M-1024	Vanilla	33.52	57.68	24.16	38.22	4.70	33.80	0.28
60M-256	S1+OA	31.93	32.46	0.53	32.32	0.39	32.00	0.07
60M-512	S1+OA	31.83	32.30	0.06	32.18	0.35	31.89	0.47
60M-1024	S1+OA	32.25	32.85	0.60	32.73	0.48	32.32	0.07
130M-256	Vanilla	22.89	46.49	23.60	28.31	5.42	23.07	0.18
130M-512	Vanilla	22.80	42.34	19.54	28.14	5.34	22.98	0.18
130M-1024	Vanilla	22.93	38.78	15.85	29.04	6.11	23.16	0.23
130M-256	S1+OA	22.78	23.21	0.43	23.10	0.32	22.83	0.05
130M-512	S1+OA	22.73	23.16	0.43	23.04	0.31	22.79	0.06
130M-1024	S1+OA	23.87	23.28	0.41	23.19	0.32	22.94	0.07

Table 6: Performance results for various model sizes and setups under longer sequence length.

In Table 6 we provide results when trained with sequence length of 512 and 1024, and compare with sequence length 256. As can be seen, our model is very robust when we increase the sequence length, showing no noticeable performance drop in perplexity be it under the general setting, or when quantized. In all cases, especially under quantization schemes, our method outperforms the vanilla one when trained with longer sequences.

C LARGER LLMS

We show that the first token attention and the outliers happen also in large modern LLMs such as Llama-3.1-8B. Furthermore, these issues happen regardless if the training is done in unsupervised manner (next-token prediction) or supervised manner (intruction tuning). We downloaded Llama-3.1-8B (https://huggingface.co/meta-llama/Llama-3.1-8B) and Llama3.1-8B-Instruct (https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct).

In Table 8, we show that in these large models, the first token attention increases (at over 95% compared to the results shown in the main paper. Furthermore, we also checked the cumulative sum of attention to the first token and found it out to be at 73.49%. In other words, 73.49% of the entire attention in Llama-3.1-8B is in the first token. This can be interpreted that the larger the network, the more specialized the heads are, and most of the heads will simply do nothing. Attending on the first token is the mechanism the Transformer has developed to learn to do nothing. We also check the kurtosis of Llama-3.1-8B, showing that the method has a very high kurtosis for both the first token and on average.

Finally, we show that these results remain very similar if the model is finetuned in instruction data. We observe that the first token attention and first token kurtosis is virtually the same in Instruct model as in the original one, while the average kurtosis actually increases in the Instruct model. Thus, we conclude that our findings stand for modern large LLMs, regardless if they are finetuned in instruction data or not.

Method	%1st_attention	Sum_first_token	1st kurtosis	Average kurtosis
Llama-3.1-8B	95.45	73.49	1227	55
Llama-3.1-8B-Instruct	95.53	70.13	1228	69

Table 7: Comparison of attention and kurtosis metrics for LLama-3.1-8B and Llama-3.1-8B-Instruct.

D INSTRUCTION TUNING

To complement the previous experiment, we run a new experiment, doing instruction finetuning (supervised learning) in Alpaca dataset. We compare the results of a model trained with canonical

Method	P_{full}	P_{coarse}	Δ_{coarse}	P_{mod}	Δ_{mod}	P_{fine}	Δ_{fine}	$1^{st}_{att}(\%)$	$1^{st}_{att}(sum)$	$1^{st}_{kurtosis}$	$A_{kurtosis}$
GPT-2-130M	18.33	31.03	12.07	21.09	2.76	18.49	0.16	65.57	41.49	564.75	69.31
GPT-2-130M + S1 + OA	**18.29**	**18.31**	**0.02**	**18.31**	**0.02**	**18.29**	**0.0**	**2.4**	**0.8**	**2.97**	**2.96**

Table 8: Results on instruction tuning using GPT-2-130M as baseline and comparing with our approach. P_{full} represents perplexity without any quantization. P_{coarse}, P_{mod} and P_{fine} represent perplexity under coarse, moderate and fine quantization. Similarly, Δ_{coarse}, Δ_{mod} and Δ_{fine} represent the increase in perplexity under these three quantization schemes compared to the not quantized method. $1^{st}_{att}(\%)$ represent the percentage of tokens that attend to the first one, $1^{st}_{att}(sum)$ represent the cumulative sum of the attention in the first token, $1^{st}_{kurtosis}$ represent the first token kurtosis, while $A_{kurtosis}$ represents the average kurtosis in the network. As we can observe, our method reaches best results under every setting, while reducing the attention on the first token and kurtosis.

softmax and Adam, compared to our method with softmax-1 and OrthoAdam. We present the results in GPT-2-130m models.

We show that while both models reach roughly the same perplexity, only our model keeps the same perplexity under all three quantization schemes. On the other hand, the vanilla model drops for 12.07 points in coarse quantization, 2.76 in moderate quantization, and 0.16 in fine quantization.

To evaluate that this is a direct effect of the first token attention and outliers, we also present the results in the percentage of maximum attention in the first token, the cumulative attention score of the first token, the 1st token kurtosis, and the average kurtosis. We show that in the vanilla model, the maximum attention is in 65.57% of cases in the first token, and the cumulative attention on the first token is 41.49%. In contrast, our method has maximum attention on the first token is 2.4% and the first token contributes to only 0.8% of the attention. Furthermore, while the vanilla model has kurtosis of 564.75 for the first token, and 69.31 for the average token, our method has 2.97 kurtosis for the first token, and 2.96 kurtosis for average token, very similar to the kurtosis of normal distribution (3).

In this way, we empirically show that our findings of the main paper stand also for instruction tuning training. Models trained with instruction tuning drop in accuracy under quantization schemes because of their outliers, while our method remains stable and as we showed, has the same kurtosis as normal distribution and not high attention in the first token.

E OPTIMISER BASIS

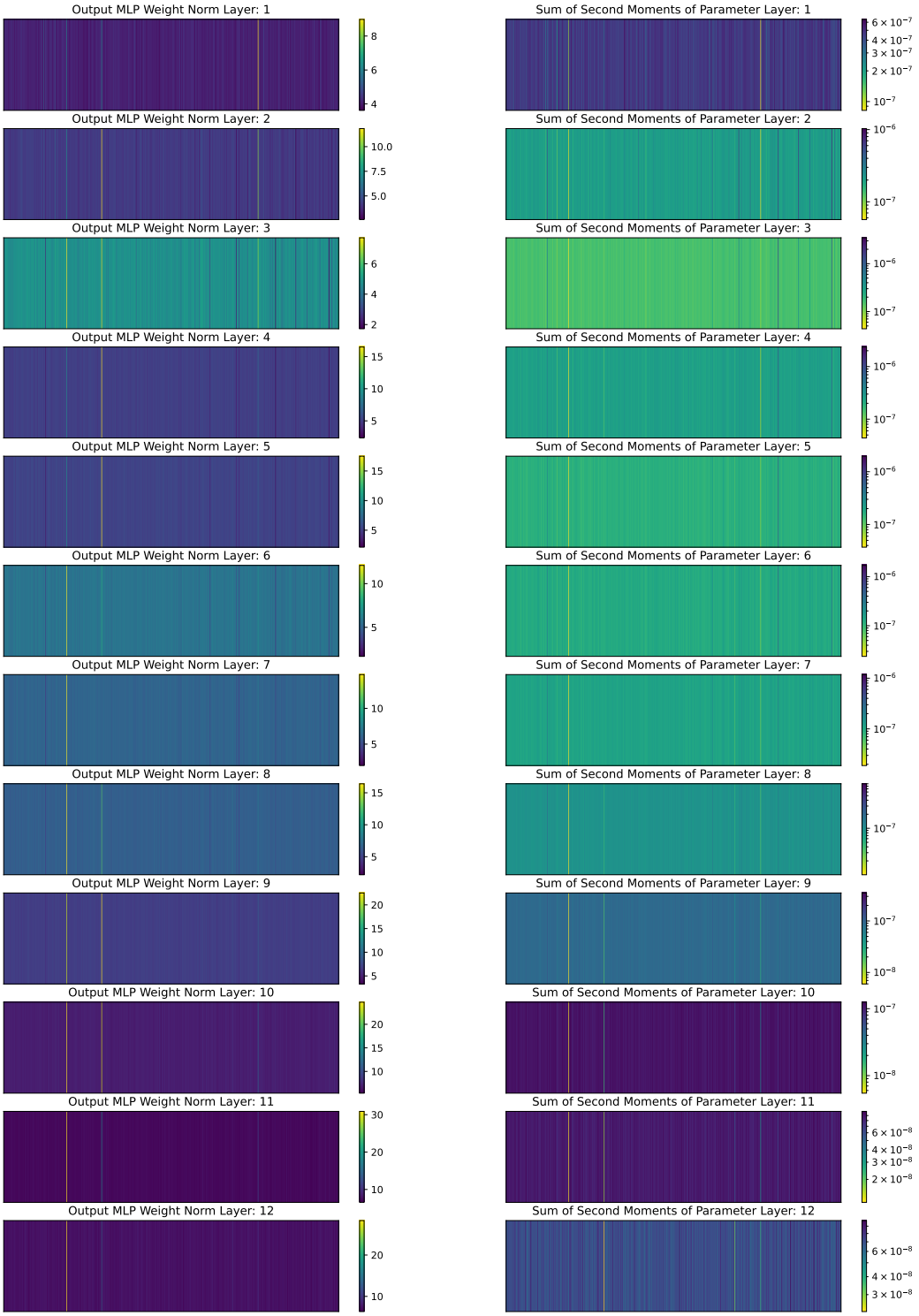

Figure 4: Left: MLP output feature weight euclidean norm for each layer in our GPT2-130M model training with canonical softmax and Adam. Right: Sum of gradient second moments for the corresponding MLP output feature weight when training with canonical softmax and Adam. Training with softmax and Adam leads to small outlier gradient second moment moving averages with lead to disproportionately large gradient steps for the feature dimensions containing these small outliers. This in turn leads to large outliers in the model weights.

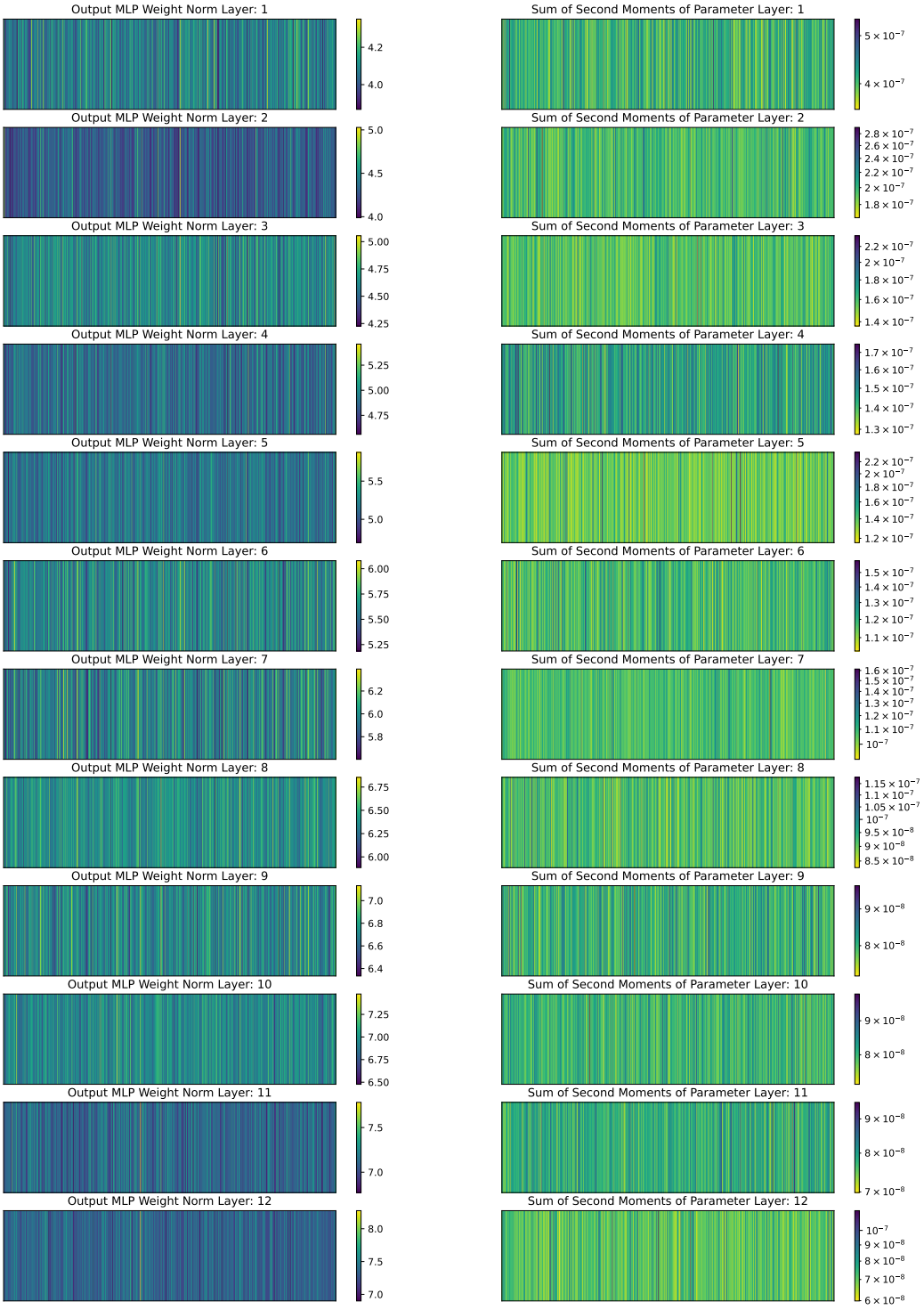

Figure 5: Left: MLP output feature weight euclidean norm for each layer in our GPT2-130M model training with softmax-1 and OrthoAdam. Right: Sum of gradient second moments for the corresponding MLP output feature weight when training with softmax-1 and OrthoAdam. Training with softmax-1 and OrthoAdam leads to gradient second moment moving averages to be considerably more uniform than training with softmax and Adam. This prevents large gradient steps from being taken removing outlier weights in the model.

Our observation of outlier activations in the hidden states of language models leads us to the conclusion that some aspect of the training process is "basis-dependent". "Basis-independent" functions are those which are equivariant under orthogonal transformations. A function $f(x)$ is basis-independent if $f(Qx) = Qf(x)$ for any orthogonal matrix Q. We remove biases from our linear layers and introduce a single-scale version of RMSNorm to remove "basis-dependent" effects from the model itself. It is straightforward to show that linear layers with biases (*i.e.*, affine transformations) and multi-scale RMSNorm (as is standard) are not basis-independent (*i.e.*, they are basis-dependent). SGD and SGD with momentum are basis-independent: We can show that the standard SGD update rule is basis-independent.

Given the standard SGD update rule $\theta_{t+1} = \theta_t - \eta \nabla L(\theta_t)$ transforms the parameters to $\rightarrow \hat{\theta}_t = \mathbf{Q}\theta_t$ where θ_t is an orthogonal matrix:

$$\hat{\theta}_{t+1} = \hat{\theta}_t - \eta \nabla \hat{L}(\hat{\theta}_t) \tag{E.1}$$

By chain rule:

$$\begin{aligned} \nabla \hat{L}(\hat{\theta}_t) = \nabla L(\theta_t) \cdot \frac{\partial \theta_t}{\partial \hat{\theta}_t} = \nabla L(\theta_t) \cdot \mathbf{Q}^T = \mathbf{Q} \nabla L(\theta_t), \\ \hat{\theta}_{t+1} = \hat{\theta}_t - \eta \mathbf{Q} \nabla L(\theta_t) = \mathbf{Q}(\theta_t - \eta \nabla L(\theta_t)) = \mathbf{Q}\theta_{t+1}. \end{aligned} \tag{E.2}$$

SGD with momentum follows similarly.

However, Adam and RMSProp are not basis-independent because of tracking the second-order moments. Let v_t be the second-order moment of the gradients at step t, then

$$v_{t+1} = \beta_2 v_t + (1 - \beta_2)(\nabla L(\theta_t) \odot \nabla L(\theta_t)) \tag{E.3}$$

$$\begin{aligned} \hat{v}_{t+1} &= \beta_2 \hat{v}_t + (1 - \beta_2)(\nabla \hat{L}(\hat{\theta}_t) \odot \nabla \hat{L}(\hat{\theta}_t)) \\ &= \beta_2 \hat{v}_t + (1 - \beta_2)(\mathbf{Q} \nabla L(\theta_t) \odot \mathbf{Q} \nabla L(\theta_t)) \\ &\neq \mathbf{Q} v_{t+1} \end{aligned} \tag{E.4}$$

The above shows that SGD and SGD with momentum provide a basis-independent update rule which is proportional to the gradient, on the other hand, Adam and RMSProp are not basis-independent and provide updates that depend on the element-wise root of the second-order moment of the gradient. We believe that basis-dependent functions (whether in the model or the optimizer) are the cause of the outlier activations we observe in the hidden states of language models. The second-order moment tracking in Adam and RMSProp allows disproportionately large gradient updates in certain weights of the model especially in the early steps of training where the moment moving averages are not well-calibrated (a result of adaptive per parameter learning rate scaling). Therefore we expect features for which the second-order moment is small to have disproportionately large weights in the model. Disproportionately large weights in particular dimensions of the model cause the outlier activations we observe. Using OrthoAdam, the moving average moments of the gradients are computed in a unique random orthogonal basis for each parameter, which prevents small values of the second-order moment from causing disproportionately large updates in the model parameters (outlier gradients large or small in the model basis are transformed to be likely of similar magnitude in the orthogonal basis).

The outlier weights in the model are generally contained in the output linear layer of each MLP block. To verify our intuition above, we have computed the norm of the output weights in output linear layer of each MLP block in a GPT2-130M model trained with/without softmax-1 and OrthoAdam. We additionally plot the sum of the second-order moments of the gradients of the output weights in each MLP block. We observe that in *all* cases of large outlier weights in the model, the sum of the second-order moments of the gradients of the output weights in the corresponding MLP block are *small* outliers. This is consistent with our intuition that the *small* outlier second-order moments of the gradients are causing the *large* outlier weights in the model. We show these plots in Figure 6 and Figure 5 respectively.

F TRANSFORMING THE OUTPUT INTO ORTHOGONAL BASIS

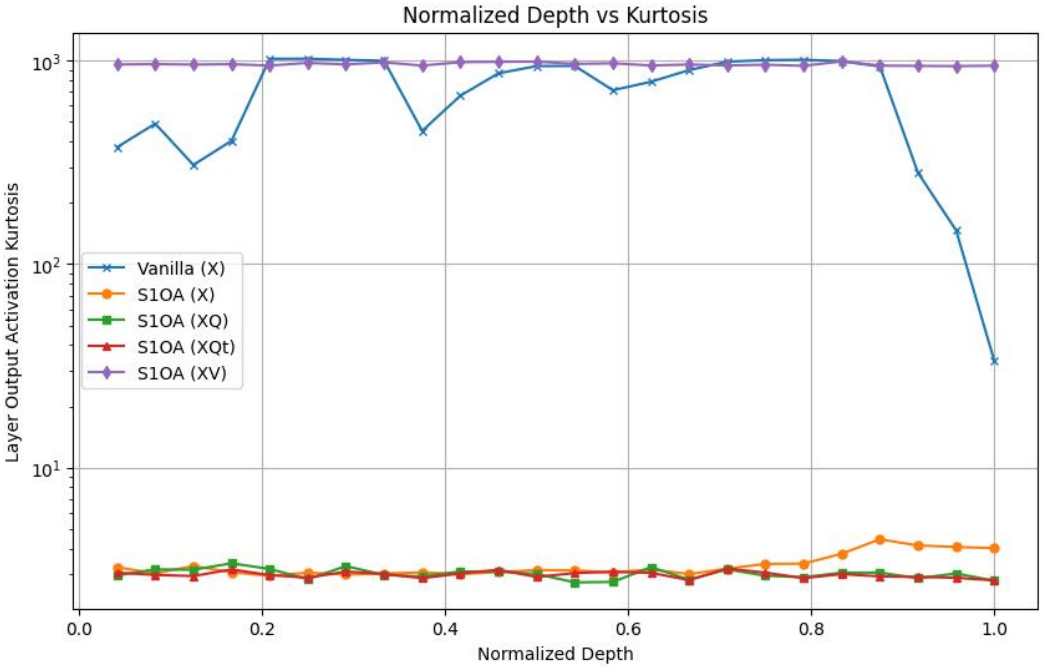

Figure 6: Transforming the output into orthogonal basis

We investigated transforming the output of each layer (*i.e.*, the added activations to the hidden states for said layer) into the orthogonal basis used by that output layer in OrthoAdam. We use our GPT2-350M model trained with Softmax-1 and OrthoAdam. Given layer i has output $\mathbf{X}_i \in \mathbb{R}^{L \times D}$ and the orthogonal basis for the output layer in OrthoAdam is $\mathbf{Q}_i \in \mathcal{O}^D$, where L is the sequence length, D is the hidden dimension and $\mathcal{O}^D$ is the set of $D \times D$ orthogonal matrices. We plot the activation kurtosis of $\mathbf{X}_i$, $\mathbf{X}_i\mathbf{Q}_i$, $\mathbf{X}_i\mathbf{Q}_i^T$ and $\mathbf{X}_i\mathbf{V}_i$ where $\mathbf{V}_i$ is the right singular vectors of $\mathbf{X}_i$. Using $\mathbf{V}_i$ as a transformed basis gives a baseline for how large one could increase the kurtosis of the activations by transforming them into an orthogonal basis. Additionally we give the activation kurtosis for our GPT2-350M vanilla model.

G KURTOSIS GROWS WITH THE NUMBER OF DIMENSIONS IN TRANSFORMERS

In this section, we use some observations from the hidden states of transformer models to illustrate how the kurtosis of the hidden states grows with the number of dimensions in the hidden states. This is something we observe empirically in the hidden states of transformer models and is a key motivation for our work. Table 2 shows the kurtosis of the hidden states of transformer models trained *without softmax-1 or OrthoAdam* grows as the model size increases, as does the maximum activation value in the hidden states.

To make this mathematically rigorous, we shall consider a simple example, in which we shall approximate the hidden states of a transformer model at a single token position as a D-dimensional vector comprising of the sum of a scaled one-hot vector and a standard normal vector.

Consider a D-dimensional vector $\mathbf{x}$ which is the sum of two D-dimensional vectors $\alpha\mathbf{e}_i$ and $\mathbf{z}$, where $\mathbf{e}_i$ is the ith unit vector in the standard basis, $\mathbf{x} \in \mathbb{R}^D$, $\alpha \in \mathbb{R}$ and $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_D)$. Therefore the elements of $\mathbf{x}$ are given by:

$$x_j = \alpha\delta_{ij} + z_j \quad \text{for } j = 1, 2, \ldots, D$$

where δ_{ij} is the Kronecker delta function. The mean is given by:

$$\begin{aligned}
\mu = \mathbb{E}[x_j] &= \mathbb{E}[\alpha\delta_{ij} + z_j] = \alpha\mathbb{E}[\delta_{ij}] + \mathbb{E}[z_j] \\
&= \frac{\alpha}{D} + 0 = \frac{\alpha}{D} \quad \text{as } \mathbb{E}[z_j] = 0 \text{ by definition of the standard normal distribution} \\
&\boxed{\mu = \frac{\alpha}{D}}
\end{aligned} \tag{G.1}$$

The variance is given by:

$$\begin{aligned}
\sigma^2 &= \text{Var}[x_j] = \text{Var}[\alpha\delta_{ij} + z_j] \\
&= \alpha^2\text{Var}[\delta_{ij}] + \text{Var}[z_j] \quad \text{as } \alpha\delta_{ij} \text{ and } z_j \text{ are independent in our model} \\
&= \alpha^2\text{Var}[\delta_{ij}] + 1 \quad \text{as Var}[z_j] = 1 \text{ by definition of the standard normal distribution}
\end{aligned}$$

$$\text{Var}[\delta_{ij}] = \mathbb{E}\left[\delta_{ij}^2\right] - \left(\mathbb{E}[\delta_{ij}]\right)^2 = \frac{1}{D}\left(1 - \frac{1}{D}\right)$$

Therefore:

$$\boxed{\sigma^2 = \frac{\alpha^2}{D}\left(1 - \frac{1}{D}\right) + 1} \tag{G.2}$$

The kurtosis of the elements of $\mathbf{x}$ is given by:

$$\text{Kurt}[x_j] = \mathbb{E}\left[\left(\frac{x_j - \mu}{\sigma}\right)^4\right] = \frac{\mathbb{E}\left[(x_j - \mu)^4\right]}{\sigma^4}$$

When $j \neq i$:

$$\begin{aligned}
\mathbb{E}\left[(x_j - \mu)^4\right] &= \mathbb{E}\left[\left(z_j - \frac{\alpha}{D}\right)^4\right] \\
&= \mathbb{E}\left[z_j^4 - 4z_j^3\frac{\alpha}{D} + 6z_j^2\left(\frac{\alpha}{D}\right)^2 - 4z_j\left(\frac{\alpha}{D}\right)^3 + \left(\frac{\alpha}{D}\right)^4\right] \\
&= \mathbb{E}\left[z_j^4\right] - 4\mathbb{E}\left[z_j^3\right]\frac{\alpha}{D} + 6\mathbb{E}\left[z_j^2\right]\left(\frac{\alpha}{D}\right)^2 - 4\mathbb{E}[z_j]\left(\frac{\alpha}{D}\right)^3 + \left(\frac{\alpha}{D}\right)^4
\end{aligned}$$

As $\mathbb{E}\left[z_j^3\right] = 0$ and $\mathbb{E}\left[z_j^4\right] = 3$:

$$= 3 + 6\left(\frac{\alpha}{D}\right)^2 + \left(\frac{\alpha}{D}\right)^4$$

When $j = i$:

$$\begin{aligned}
\mathbb{E}\left[(x_j - \mu)^4\right] &= \mathbb{E}\left[\left(\alpha + z_j - \frac{\alpha}{D}\right)^4\right] \\
&= \mathbb{E}\left[\left(\alpha\left(1 - \frac{1}{D}\right) + z_j\right)^4\right] \\
&= \mathbb{E}\left[\left(\alpha\left(1 - \frac{1}{D}\right)\right)^4 + 4\left(\alpha\left(1 - \frac{1}{D}\right)\right)^3 z_j \right. \\
&\qquad \left. + 6\left(\alpha\left(1 - \frac{1}{D}\right)\right)^2 z_j^2 + 4\left(\alpha\left(1 - \frac{1}{D}\right)\right) z_j^3 + z_j^4\right] \\
&= \left(\alpha\left(1 - \frac{1}{D}\right)\right)^4 + 6\left(\alpha\left(1 - \frac{1}{D}\right)\right)^2 + 3
\end{aligned}$$

Therefore, the *overall fourth moment* of the elements of $\mathbf{x}$ is given by:

$$\begin{aligned}
\mathbb{E}\left[(x_j - \mu)^4\right] = \frac{1}{D}\left(\left(\alpha\left(1 - \frac{1}{D}\right)\right)^4 + 6\left(\alpha\left(1 - \frac{1}{D}\right)\right)^2 + 3\right) \\
+ \frac{D-1}{D}\left(3 + 6\left(\frac{\alpha}{D}\right)^2 + \left(\frac{\alpha}{D}\right)^4\right)
\end{aligned}$$

And the kurtosis of the elements of $\mathbf{x}$ is given by:

$$\text{Kurt}\left[x_j\right] = \frac{\frac{1}{D}\left(\left(\alpha\left(1 - \frac{1}{D}\right)\right)^4 + 6\left(\alpha\left(1 - \frac{1}{D}\right)\right)^2 + 3\right) + \frac{D-1}{D}\left(3 + 6\left(\frac{\alpha}{D}\right)^2 + \left(\frac{\alpha}{D}\right)^4\right)}{\left(\frac{\alpha^2}{D}\left(1 - \frac{1}{D}\right) + 1\right)^2}$$

$$\boxed{\text{Kurt}\left[x_j\right] = \frac{3 + \frac{\alpha^4}{D} + \frac{6\alpha^2}{D} - \frac{4\alpha^4}{D^2} - \frac{6\alpha^2}{D^2} + \frac{6\alpha^4}{D^3} - \frac{3\alpha^4}{D^4}}{1 + \frac{2\alpha^2}{D} - \frac{2\alpha^2}{D^2} + \frac{\alpha^4}{D^2} - \frac{2\alpha^4}{D^3} + \frac{\alpha^4}{D^4}}} \tag{G.3}$$

At this point, we can see that Kurtosis is a function of α and D, however if we consider the limit as $D \to \infty$, we can see that Kurt$[x_j] \to 3$, *i.e.*, the kurtosis of a Gaussian distribution. However, this neglects the importance of the scaling factor α which we know empirically is larger than the dimensionality of the hidden states. The table below summarises the maximum activation values (analogous to α) and the dimension of the hidden states for the models we trained. Given this

Model	#Parameters	Model Size (D)	Max Activation (α)
GPT2	60M	512	1856
	130M	768	7018
	350M	1024	40196
	1.4B	2048	56798
Llama	130M	768	4623

Table 9: Model sizes and maximum activation values for the models used in our experiments.

empirical information, we make the *conservative* assumption that $\alpha = D$. Under this assumption

which is supported by our empirical observations, Equation (G.3) simplifies to:

$$\begin{aligned}
\text{Kurt}\left[x_j\right] &= \frac{3 + \frac{D^4}{D} + \frac{6D^2}{D} - \frac{4D^4}{D^2} - \frac{6D^2}{D^2} + \frac{6D^4}{D^3} - \frac{3D^4}{D^4}}{1 + \frac{2D^2}{D} - \frac{2D^2}{D^2} + \frac{D^4}{D^2} - \frac{2D^4}{D^3} + \frac{D^4}{D^4}} \\
&= \frac{3 + D^3 + 6D - 4D^2 - 6 + 6D - 3}{1 + 2D - 2 + D^2 - 2D + 1} \\
&= \frac{D^3 - 4D^2 + 12D - 6}{D^2} \\
\boxed{\text{Kurt}\left[x_j\right] = D - 4 + \frac{12}{D} - \frac{6}{D^2} = O(D)} &
\end{aligned} \tag{G.4}$$

Using our conservative assumption that $\frac{\alpha}{D} = 1$, we can see that the kurtosis of the hidden states grows linearly with the dimensionality of the hidden states when D is in the region of $10^3 - 10^5$ as is the case for transformer models.

This simple example serves as a mathematical illustration of the empirical observations we make in the hidden states of transformer models. We have shown that the kurtosis of the hidden states is expected to grow linearly with the dimensionality of the hidden states, and so the issue of outlier activations is expected to grow as the hidden states of transformer models grow in size.

H ORTHOGONAL TRANSFORMATIONS AND REDUCTION IN ℓ_∞–NORM AND KURTOSIS

From our simple model in Appendix G we have a simplified model of Transformer hidden states, $\mathbf{x} \in \mathbb{R}^D$, where the first element is α and the rest are standard normal random variables.

$$\mathbf{x} = \alpha \mathbf{e}_i + \mathbf{z} \quad \text{where } z_j \sim \mathcal{N}(0, 1)$$

From this model, we can compute the expected ℓ_2–norm:

$$\mathbb{E}\left[\|\mathbf{x}\|_2^2\right] = \sum_{j=1}^{D} x_j^2 = \alpha^2 + \sum_{j=1}^{D} z_j^2 = \alpha^2 + D\text{Var}\left[z_j\right] = \alpha^2 + D \tag{H.1}$$

Using the triangle inequality, we can compute a range for the ℓ_∞–norm:

$$\mathbb{E}\left[\|\mathbf{x}\|_\infty\right] = \mathbb{E}\left[\max_{1 \le j \le D}\left(\left|\alpha + z_i\right|, \max_{j \neq i}\left|z_j\right|\right)\right]$$

Given $\alpha \gg 1$, we can drop the terms for $j \neq i$ and compute the expected ℓ_∞–norm using the i^{th} element:

$$\begin{aligned} \mathbb{E}\left[\|\mathbf{x}\|_\infty\right] &= \mathbb{E}\left[\left|\alpha + z_i\right|\right] \\ \left|\alpha + z_i\right| &\le |\alpha| + |z_i| \end{aligned}$$

Using folded normal distribution properties, $\mathbb{E}\left[|z_i|\right] = \sqrt{\frac{2}{\pi}} \ll \alpha$, therefore:

$$\mathbb{E}\left[\|\mathbf{x}\|_\infty\right] \approx \alpha$$

Given that $\alpha \gg 1$, we can safely assume that $\|\mathbf{x}\|_\infty^2 \approx \alpha^2$. Therefore:

$$\mathbb{E}\left[\frac{\|\mathbf{x}\|_\infty}{\|\mathbf{x}\|_2}\right] \approx \frac{\alpha}{\sqrt{D + \alpha^2}}$$

Note from Table 9 that the maximum activation value, α, is generally much larger than the model size, D.

$$\mathbb{E}\left[\frac{\|\mathbf{x}\|_\infty}{\|\mathbf{x}\|_2}\right] \approx 1 \tag{H.2}$$

We find this empirically to be the case in the middle layers of the Transformer models we study (see plots in Appendix I.3).

The ∞-norm of $\mathbf{x}$ can be thought of as a proxy for the extent of outliers in a vector. If $\frac{\|\mathbf{x}\|_2}{\|\mathbf{x}\|_\infty} \approx 1$, then a vector has at least one large outlier and consequently a high kurtosis.

We will now show that applying an orthogonal transformation to a vector can reduce the ℓ_∞-norm constrained to a fixed ℓ_2-norm. Using the same definition of $\mathbf{x}$ as above, let $\mathbf{Q} \in \mathbb{R}^{D \times D}$ be an orthogonal matrix and let $\mathbf{y} = \mathbf{Q}\mathbf{x}$.

$$\begin{aligned} \|\mathbf{y}\|_2^2 = \mathbf{y}^T\mathbf{y} = \mathbf{x}^T\mathbf{Q}^T\mathbf{Q}\mathbf{x} = \mathbf{x}^T\mathbf{x} = \|\mathbf{x}\|_2^2 \\ \mathbb{E}[\|\mathbf{y}\|_2^2] = \mathbb{E}[\|\mathbf{x}\|_2^2] = \alpha^2 + D \end{aligned} \tag{H.3}$$

This standard proof shows that applying an orthogonal transformation to a vector does not change the ℓ_2–norm of the vector. It can however lead to a dramatic reduction in the ℓ_∞–norm of the vector. We will now show that for a vector, $\mathbf{y} \in \mathbb{R}^D$, constrained to have a fixed ℓ_2–norm, $\sqrt{\alpha^2 + D}$, the ℓ_∞–norm of a vector can be reduced significantly by applying an orthogonal transformation such that $y_j = \frac{\alpha}{\sqrt{D}} + z', \forall j \in [1, D]$, where $z' \sim \mathcal{N}(0, 1)$.

$$\mathbf{y} = \mathbf{Q}\mathbf{x} = \mathbf{Q}\left(\alpha \mathbf{e}_i + \mathbf{z}\right) = \alpha \mathbf{Q}\mathbf{e}_i + \mathbf{Q}\mathbf{z}$$

Select $\mathbf{Q}$ such that $\mathbf{Q}\mathbf{e}_i = \left(\frac{1}{\sqrt{D}}, \frac{1}{\sqrt{D}}, \ldots, \frac{1}{\sqrt{D}}\right)$, given $\mathbf{Q}$ is orthogonal, $\mathbf{Q}\mathbf{z} = \mathbf{z}' \sim \mathcal{N}(0, \mathbf{I}_D)$.

$$\begin{aligned}\mathbb{E}\left[\|\mathbf{y}\|_\infty\right] &\approx \frac{\alpha}{\sqrt{D}} + \sqrt{2\ln D}, \quad \text{using extreme value theory (Cramér, 1946)} \\ &= \frac{\alpha + \sqrt{2D\ln D}}{\sqrt{D}}\end{aligned}$$

The expected ratio of ℓ_∞–norm to ℓ_2–norm is:

$$\mathbb{E}\left[\frac{\|\mathbf{y}\|_\infty^2}{\|\mathbf{y}\|_2^2}\right] = \frac{\mathbb{E}\left[\|\mathbf{y}\|_\infty^2\right]}{\mathbb{E}\left[\|\mathbf{y}\|_2^2\right]} = \frac{\left(\alpha + \sqrt{2D\ln D}\right)^2}{D\left(\alpha^2 + D\right)}$$

Using the same *conservative* assumption as in Appendix G that $\alpha = D$, Table 9 shows empirically $\alpha > D$:

$$\mathbb{E}\left[\frac{\|\mathbf{y}\|_\infty^2}{\|\mathbf{y}\|_2^2}\right] = \frac{D^2 + 2D\ln D + 2D\sqrt{2D\ln D}}{D^3 + D^2} = \frac{D + 2\sqrt{2D\ln D} + 2\ln D}{D^2 + 1}$$

As D grows, the last term of the numerator and the 1 in the denominator become negligible:

$$\mathbb{E}\left[\frac{\|\mathbf{y}\|_\infty^2}{\|\mathbf{y}\|_2^2}\right] \approx \frac{1}{D} + \frac{D + 2\sqrt{2\ln D}}{D^{\frac{3}{2}}} = O\left(\frac{1}{D}\right)$$

Therefore, under an orthogonal transformation, the ℓ_∞–norm to ℓ_2–norm ratio can be reduced significantly. It is trivial to show that Kurt$[y_j] = 3$ and we see many of our experiments which use OrthoAdam and softmax-1 exhibit this behaviour (see plots in Appendix I.2).

$$\begin{aligned}&\mathbf{x} = \alpha\mathbf{e}_i + \mathbf{z}, \quad \mathbb{E}\left[\frac{\|\mathbf{x}\|_\infty^2}{\|\mathbf{x}\|_2^2}\right] \approx 1 \quad \rightarrow \quad \mathbf{y} = \mathbf{Q}\mathbf{x}, \quad \mathbb{E}\left[\frac{\|\mathbf{y}\|_\infty^2}{\|\mathbf{y}\|_2^2}\right] \approx \frac{1}{D} \\ &\text{Kurt}\left[x_j\right] = D - 4 + \frac{12}{D} - \frac{6}{D^2} = O(D) \quad \rightarrow \quad \text{Kurt}\left[y_j\right] = 3\end{aligned}$$

The exact form of $\mathbf{Q}$ can be computed numerically or constructed using appropriately normalised Hadamard matrices (Sylvester, 1867).

I LAYER PROGRESSION OF FIRST TOKEN ATTENTION DOMINANCE, KURTOSIS, ℓ_∞-NORM TO ℓ_2-NORM RATIO AND MAXIMUM ABSOLUTE ACTIVATION

For brevity, we give metrics for the first token attention dominance, hidden state kurtosis and absolute maximum activation *averaged over all layers* in Table 2 which gives the results of the main experiments in our work.

However, the layer-wise progression of these metrics is also of interest and can provide insights into the behaviour of the model. Additionally, we provide the same metrics for popular pretrained GPT2 and Llama models to show the similarity to our models *trained without softmax-1 and OrthoAdam*.

Finally, to establish a relationship between activation kurtosis and the ℓ_∞-norm to ℓ_2-norm ratio, we calculate the Pearson's correlation coefficients between per-layer kurtosis and per-layer ℓ_∞-norm to ℓ_2-norm ratio for all models in our main experimental results from Table 2.

All metrics are computed on the same validation set of the C4 dataset (Raffel et al., 2020) as in the main paper (Section 5).

I.1 FIRST TOKEN ATTENTION DOMINANCE

We begin by examining the progression of first token attention dominance across layers. We calculate the percentage of (head, query) pairs where the query token attends most to the first (key) token. Given different models have a different number of layers, we normalise the layer index to the range $[0, 1]$ for each model.

We find a general trend across our trained models which use the canonical softmax function where the first token attention dominance begins low in the initial layers where models do initial processing of all input tokens. The dominance rises to a peak in the middle layers where heads specialise to specific sub-tasks and so the first token is attended to as a default "no-op" (Bondarenko et al., 2023; Clark et al., 2019). Finally, the dominance decreases in the final layers where the model "detokenises" the features back into token space.

I.1.1 GPT2-60M

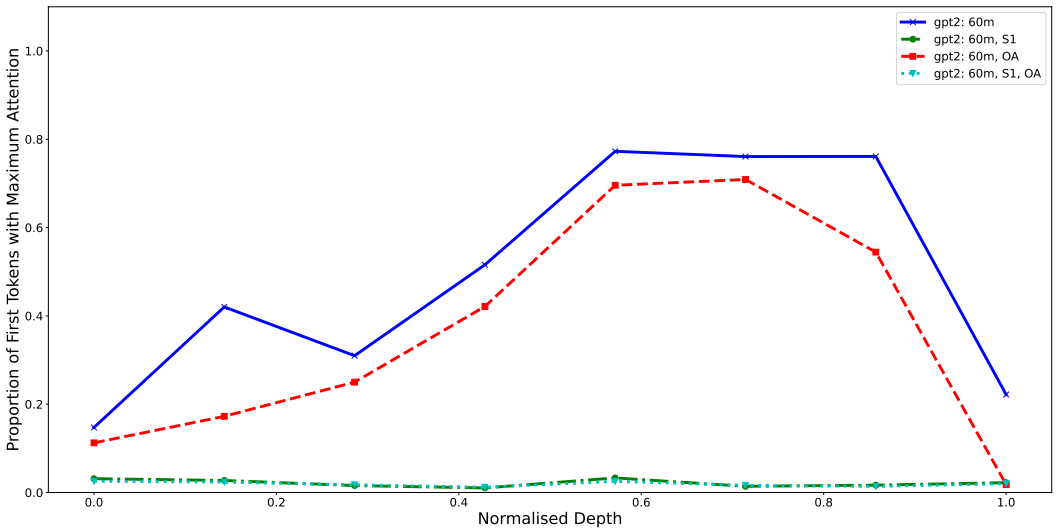

Figure 7: Layer-wise progression of first token attention dominance for GPT2-60M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.1.2 GPT2-130M

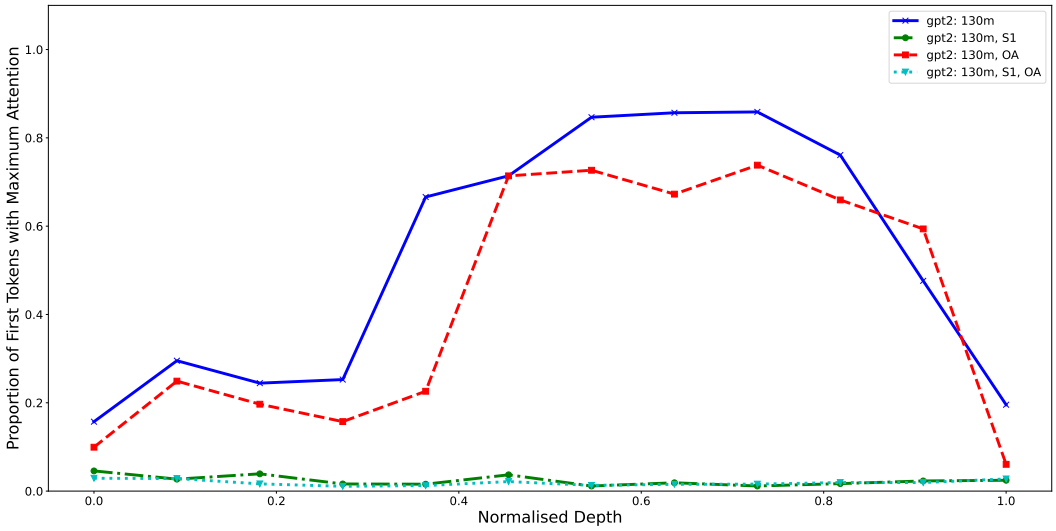

Figure 8: Layer-wise progression of first token attention dominance for GPT2-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.1.3 GPT2-350M AND GPT2-1.4B

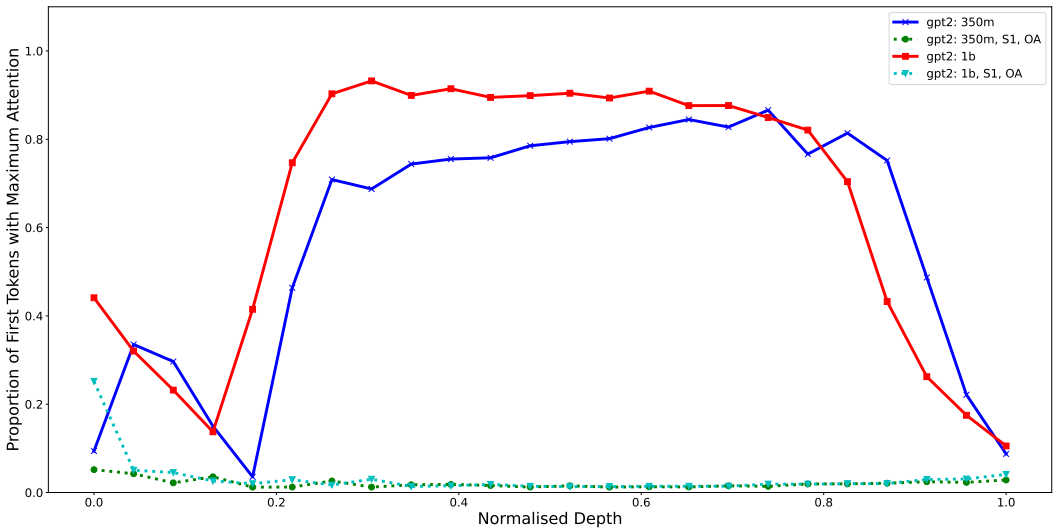

Figure 9: Layer-wise progression of first token attention dominance for GPT2-350M and GPT2-1.4B. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.1.4 LLAMA-130M

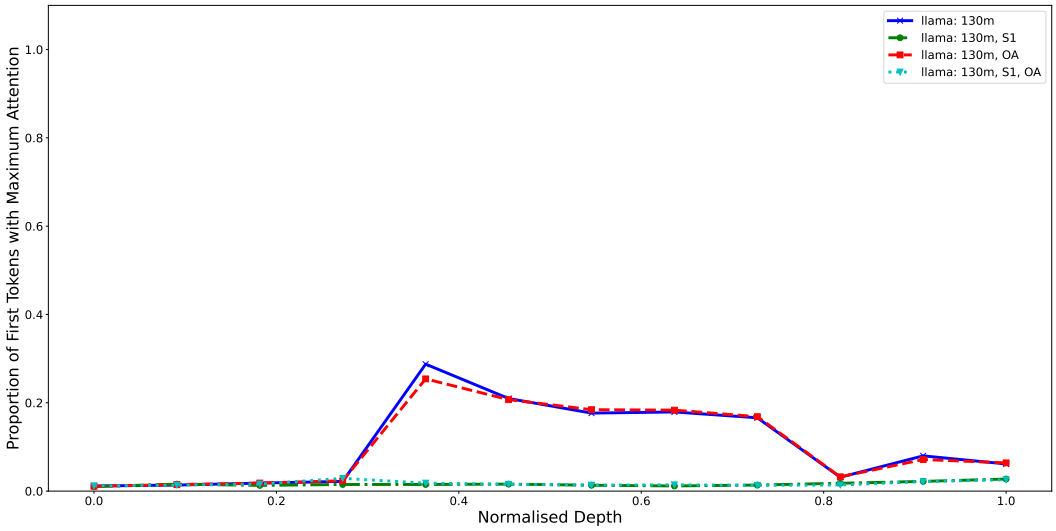

Figure 10: Layer-wise progression of first token attention dominance for Llama-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.1.5 POPULAR PRETRAINED MODELS—GPT2 AND LLAMA

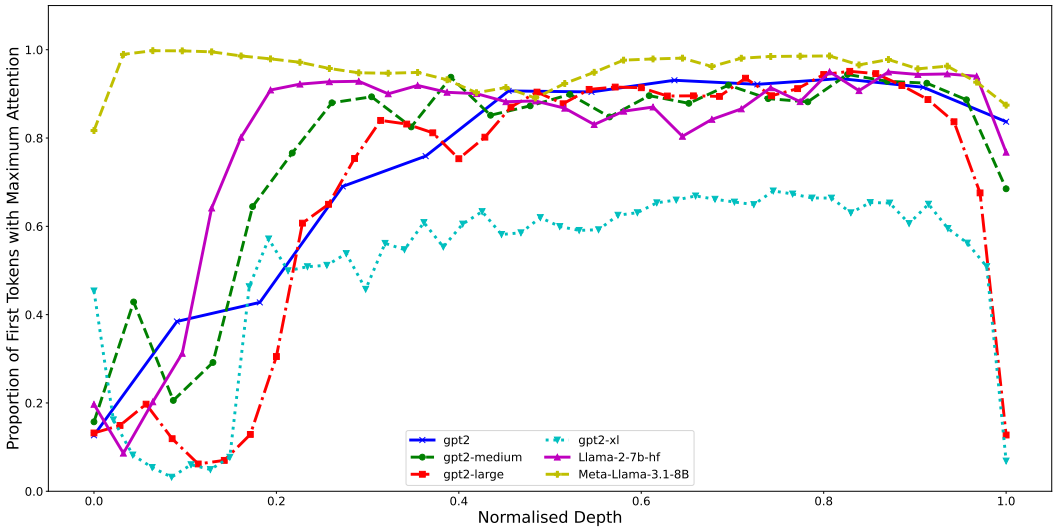

Figure 11: Layer-wise progression of first token attention dominance for popular pretrained GPT2 and Llama models. The x-axis is normalised to the range $[0, 1]$.

I.2 ACTIVATION KURTOSIS

Next, we examine the progression of activation kurtosis across layers. As observable in Table 2, the kurtosis of the first hidden state is significantly higher than the other hidden states and so we plot the kurtosis of the first hidden state only for brevity.

We observe in the plots below that models trained without OrthoAdam exhibit a general trend of increasing kurtosis as the hidden states progress through the layers. Demonstrating that multiple layers of the model contribute to the emergence of large activation values. Models trained with

OrthoAdam *but not softmax-1* exhibit a similar trend, but with lower kurtosis values initially. Finally, models trained with both OrthoAdam and softmax-1 exhibit a consistent small kurtosis across layers—around the value of 3 which is the kurtosis of a Gaussian distribution. Interestingly, GPT2-60M and GPT2-130M show small rises in the final layers—the cause of this is left for future work.

We find that some models show a reduction in kurtosis in the final layers, we again attribute this to the "detokenisation" of the features back into token space.

I.2.1 GPT2-60M

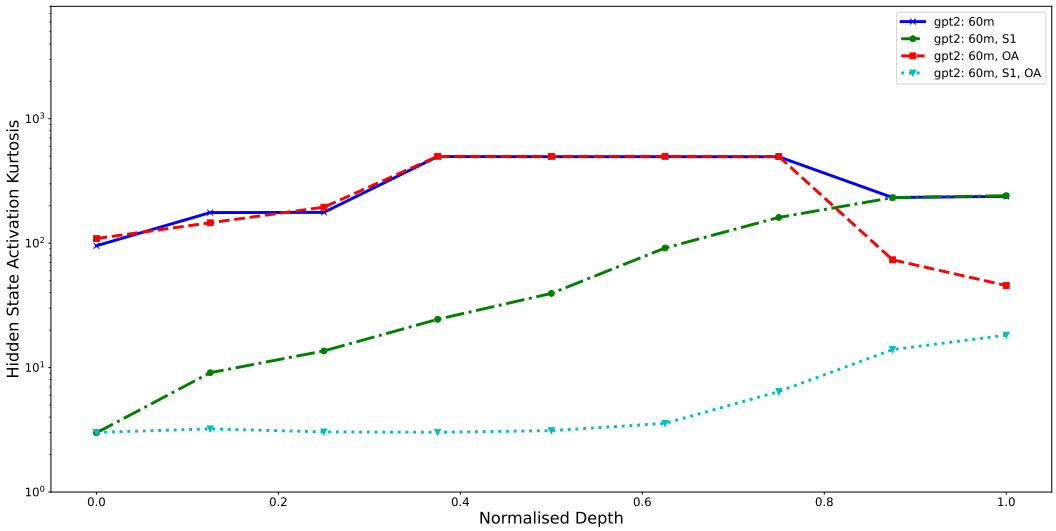

Figure 12: Layer-wise progression of activation kurtosis of the first token position for GPT2-60M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.2.2 GPT2-130M

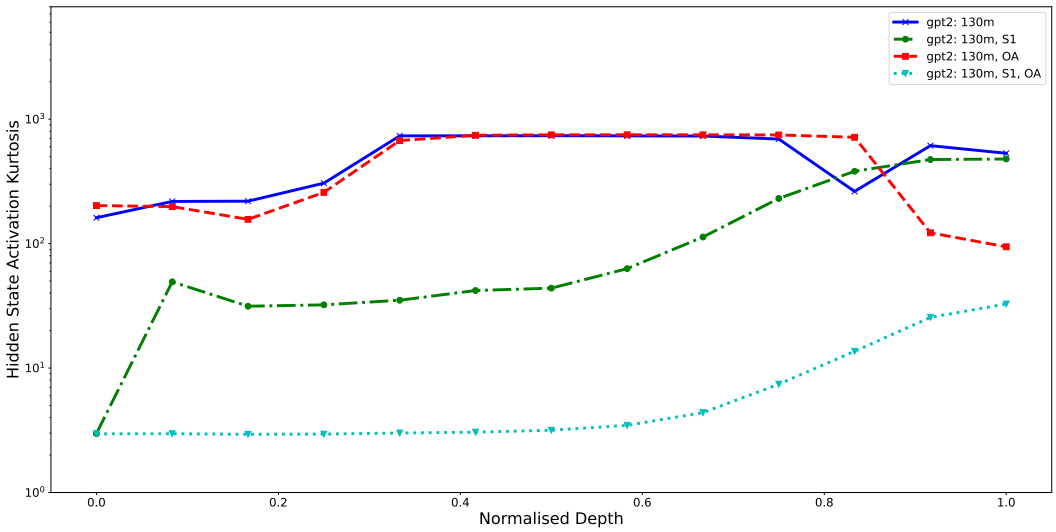

Figure 13: Layer-wise progression of activation kurtosis of the first token position for GPT2-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.2.3 GPT2-350M AND GPT2-1.4B

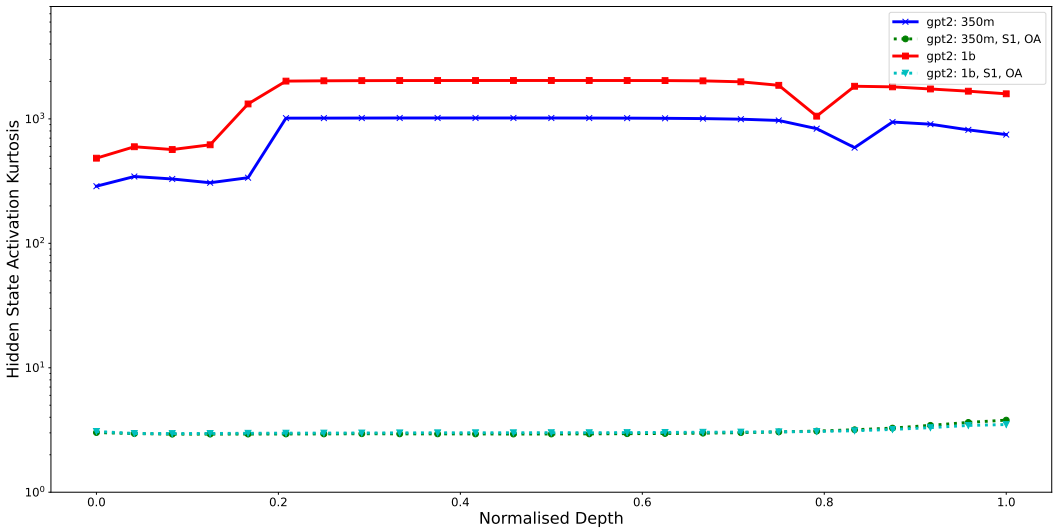

Figure 14: Layer-wise progression of activation kurtosis of the first token position for GPT2-350M and GPT2-1.4B. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.2.4 LLAMA-130M

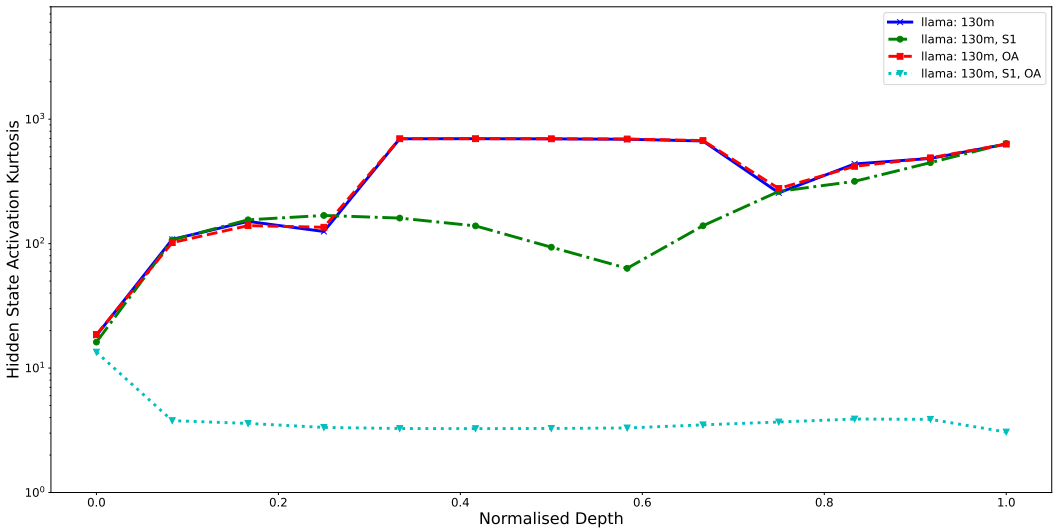

Figure 15: Layer-wise progression of activation kurtosis of the first token position for Llama-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.2.5 Popular Pretrained Models—GPT2 and Llama

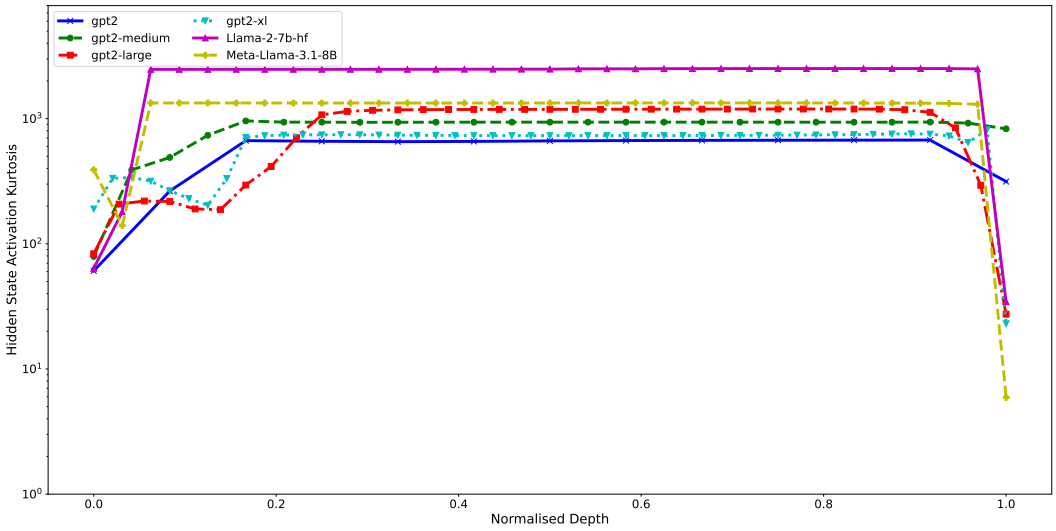

Figure 16: Layer-wise progression of activation kurtosis of the first token position for popular pretrained GPT2 and Llama models. The x-axis is normalised to the range $[0, 1]$.

I.3 ℓ_∞-Norm to ℓ_2-Norm Ratio

The plots below show the progression of the ℓ_∞-norm to ℓ_2-norm ratio across layers. We observe that models trained without OrthoAdam exhibit a general trend of increasing ratio as the hidden states progress through the layers. Once again as this ratio is maximal in the first hidden state, we plot the ratio of the first hidden state only for brevity (as done for kurtosis).

The trends are similar to the kurtosis plots and so the same commentary applies.

I.3.1 GPT2-60M

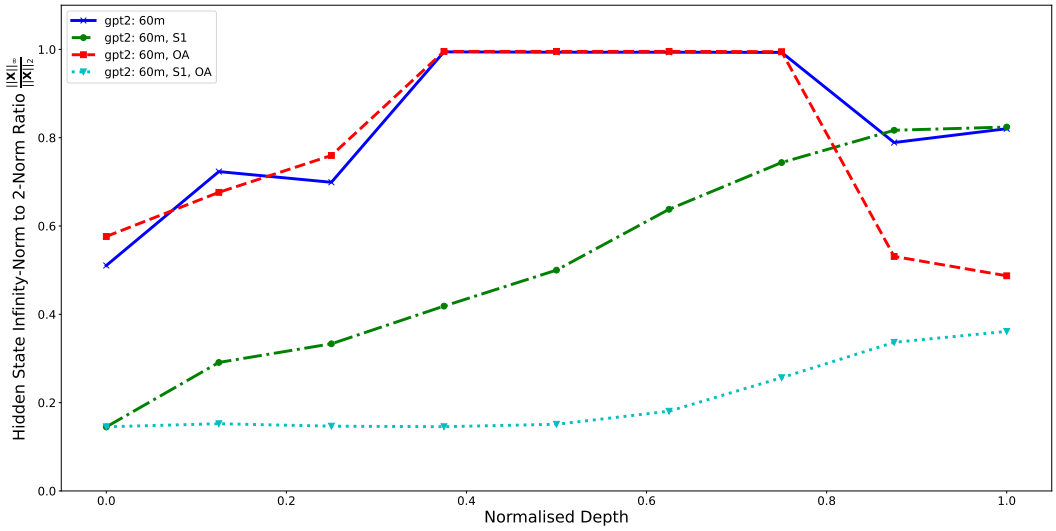

Figure 17: Layer-wise progression of the ℓ_∞-norm to ℓ_2-norm ratio in the hidden states of the first token position for GPT2-60M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.3.2 GPT2-130M

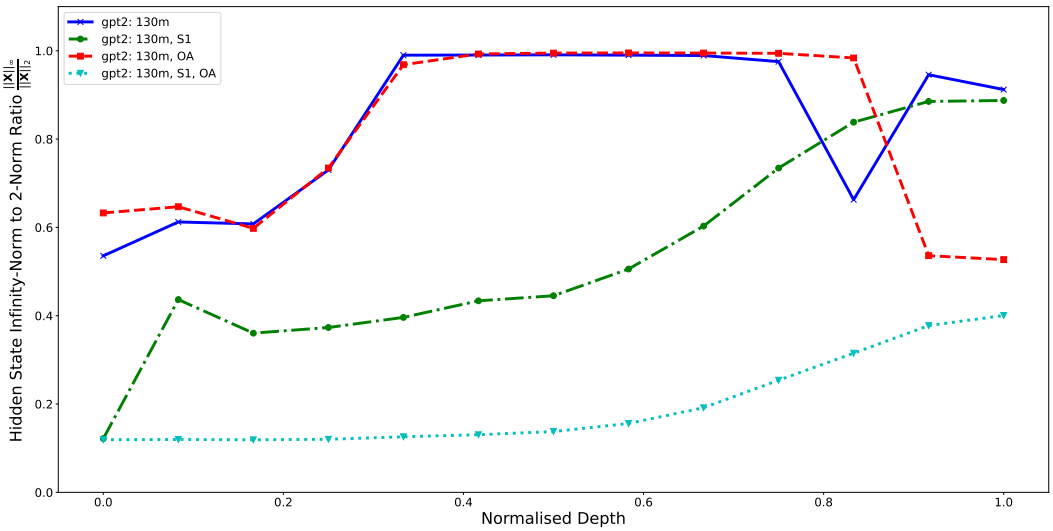

Figure 18: Layer-wise progression of the ℓ_∞-norm to ℓ_2-norm ratio in the hidden states of the first token position for GPT2-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.3.3 GPT2-350M AND GPT2-1.4B

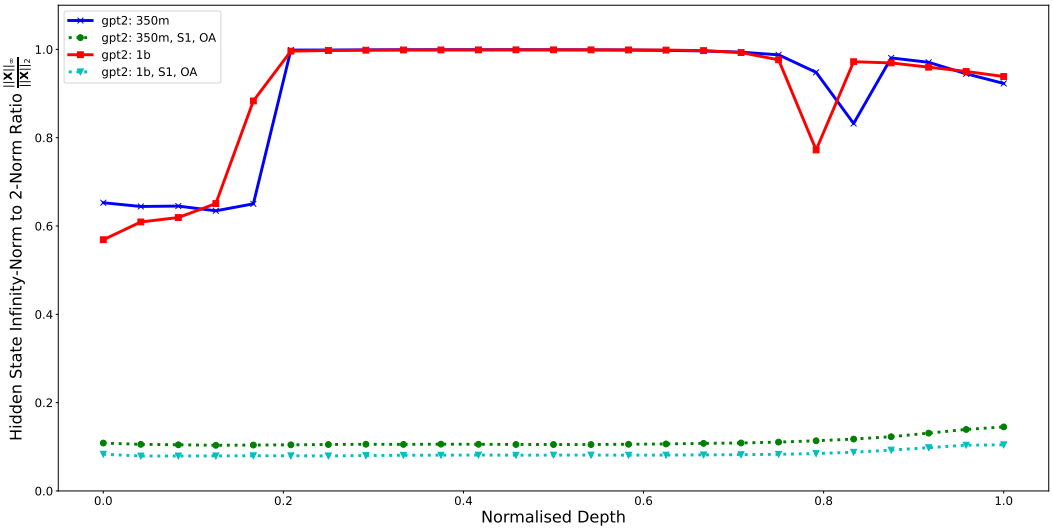

Figure 19: Layer-wise progression of the ℓ_∞-norm to ℓ_2-norm ratio in the hidden states of the first token position for GPT2-350M and GPT2-1.4B. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.3.4 LLAMA-130M

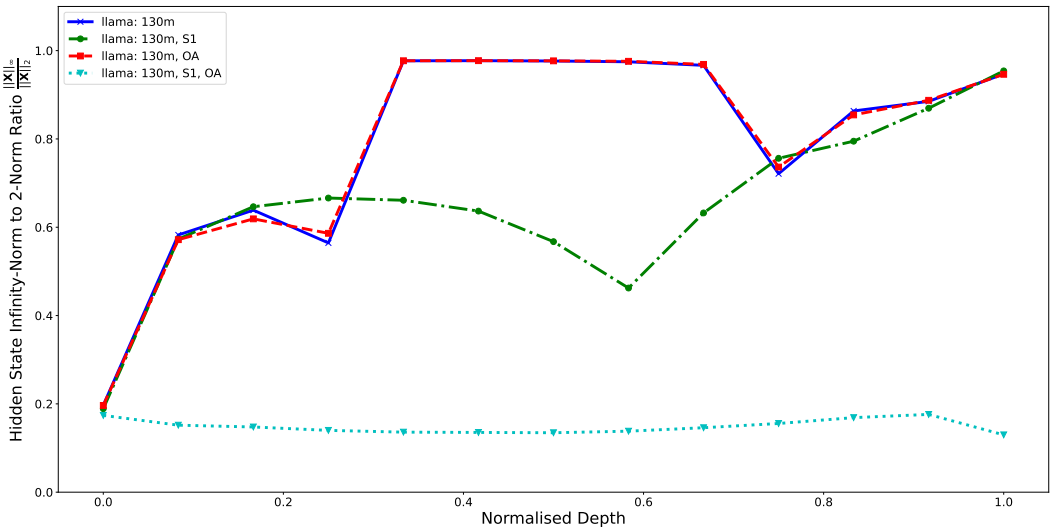

Figure 20: Layer-wise progression of the ℓ_∞-norm to ℓ_2-norm ratio in the hidden states of the first token position for Llama-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.3.5 POPULAR PRETRAINED MODELS—GPT2 AND LLAMA

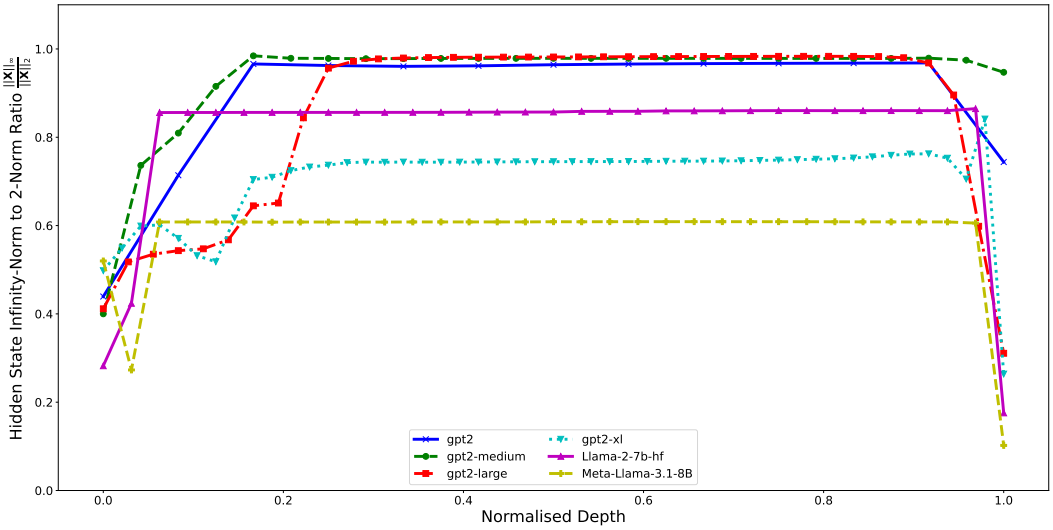

Figure 21: Layer-wise progression of the ℓ_∞-norm to ℓ_2-norm ratio in the hidden states of the first token position for popular pretrained GPT2 and Llama models. The x-axis is normalised to the range $[0, 1]$.

To further clarify that in Transformer models the ℓ_∞-norm to ℓ_2-norm ratio is a proxy for activation kurtosis, we calculate the Pearson's correlation coefficients between the two metrics for all models in our main experimental results from Table 2 and public GPT2 and Llama models. The results are shown in Table 10. We find a strong positive correlation between the two metrics across all models which reinforces our intuition that using orthogonal matrices to transform the gradients in the optimiser is an effective way to mitigate the emergence of large activation values, as an orthogonal transformation can reduce the ℓ_∞-norm of a vector substantially for a given ℓ_2-norm.

Model	#Parameters	Softmax+1?	OrthoAdam?	Correlation of Kurtosis to Norm Ratio First Token	Other Tokens
GPT2 (Ours)	60M			0.961	0.932
		✓		0.932	0.934
			✓	0.986	0.972
		✓	✓	0.968	0.970
	130M			0.988	0.932
		✓		0.927	0.924
			✓	0.992	0.962
		✓	✓	0.935	0.953
	350M			0.990	0.929
		✓	✓	0.998	0.997
	1.4B			0.988	0.952
		✓	✓	0.994	0.995
Llama2 (Ours)	130M			0.931	0.903
		✓		0.864	0.877
			✓	0.931	0.905
		✓	✓	0.560	0.975
GPT2 (Public)	137M			0.985	0.944
GPT2-Medium (Public)	350M			0.969	0.846
GPT2-Large (Public)	812M			0.985	0.896
GPT2-XL (Public)	1.6B			0.956	0.939
Llama2-7B (Public)	6.7B			0.987	0.902
Llama3.1-8B (Public)	8B			0.928	0.915

Table 10: Correlation of the kurtosis and norm-ratio of the hidden states of our trained models and popular pretrained models.

I.4 Maximum Absolute Activation

Finally, we examine the progression of the maximum absolute activation across layers.

I.4.1 GPT2-60M

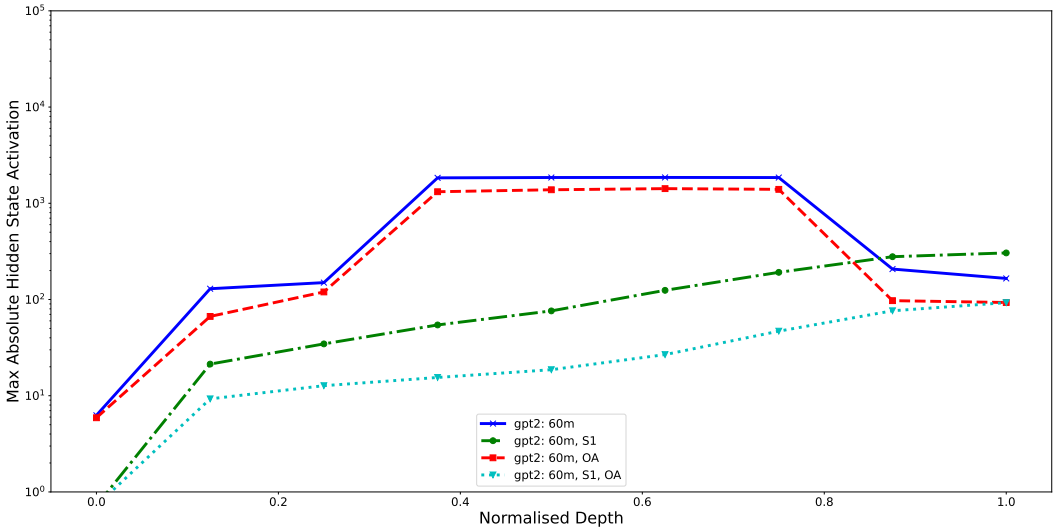

Figure 22: Layer-wise progression of the maximum absolute activation in the hidden states of the first token position for GPT2-60M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.4.2 GPT2-130M

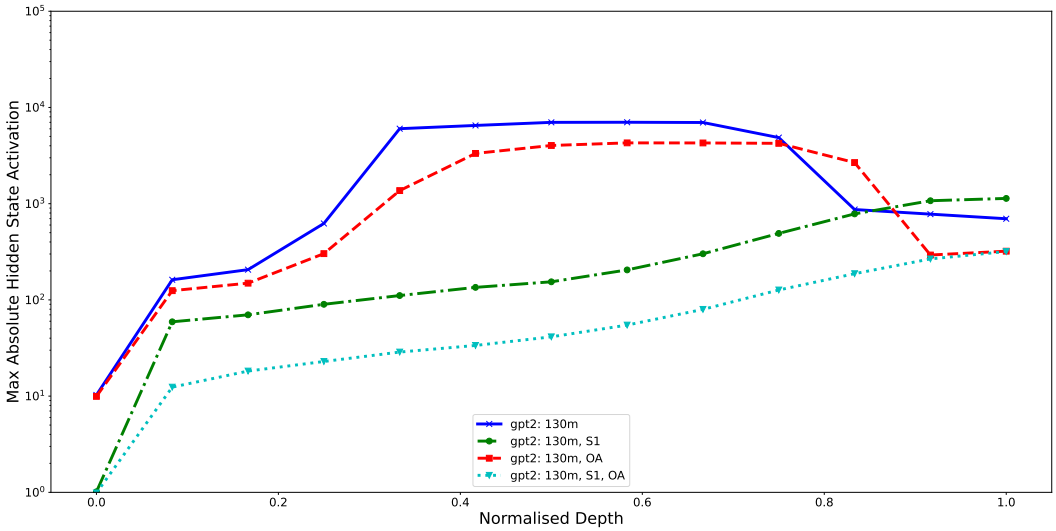

Figure 23: Layer-wise progression of the maximum absolute activation in the hidden states of the first token position for GPT2-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.4.3 GPT2-350M AND GPT2-1.4B

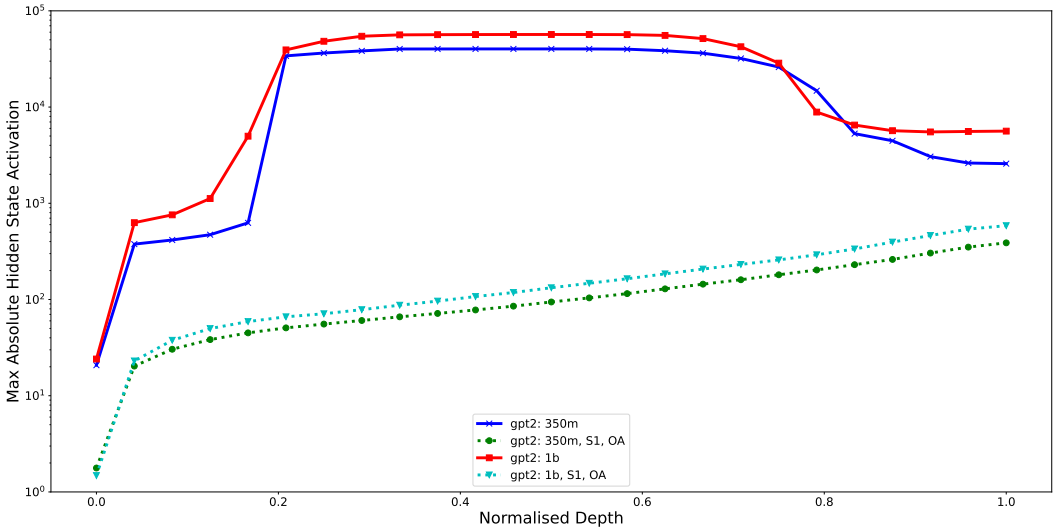

Figure 24: Layer-wise progression of the maximum absolute activation in the hidden states of the first token position for GPT2-350M and GPT2-1.4B. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.4.4 LLAMA-130M

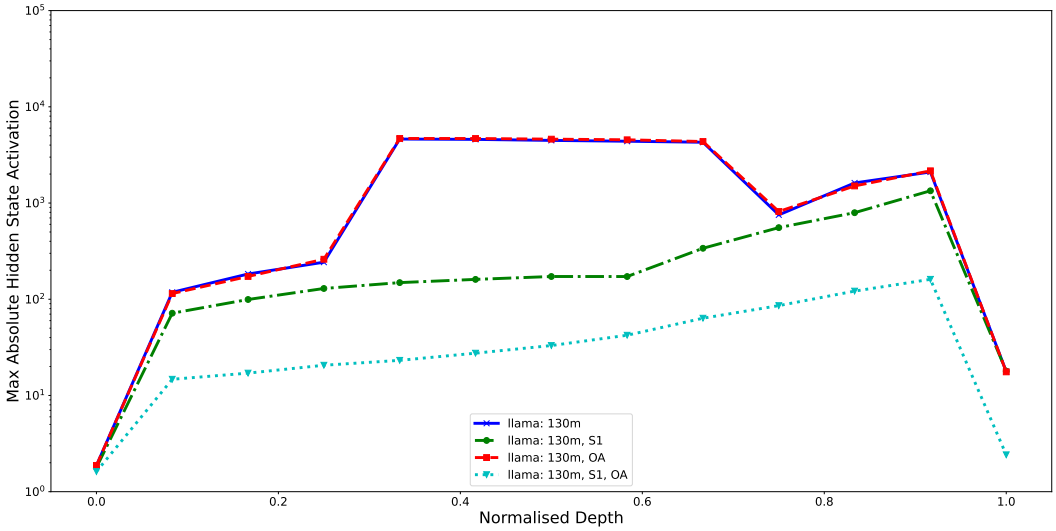

Figure 25: Layer-wise progression of the maximum absolute activation in the hidden states of the first token position for Llama-130M. The x-axis is normalised to the range $[0, 1]$. S1/OA denote models trained with softmax-1 and/or OrthoAdam.

I.4.5 POPULAR PRETRAINED MODELS—GPT2 AND LLAMA

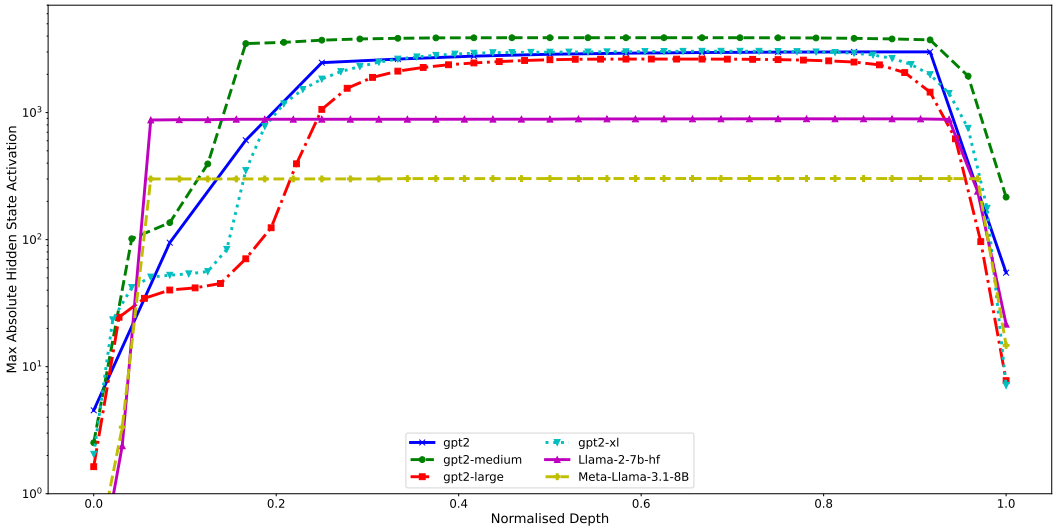

Figure 26: Layer-wise progression of the maximum absolute activation in the hidden states of the first token position for popular pretrained GPT2 and Llama models. The x-axis is normalised to the range $[0, 1]$.

J HIDDEN STATES OF PRETRAINED MODELS

In this section, we present the progression of hidden states of popular pretrained models. This shows how models establish outlier activations and how they persist in the same feature dimensions across layers. For each model we show the *absolute* activation values in the features containing the largest activations. We show the mean across layers, the first layer, $\frac{1}{4}$ and $\frac{3}{4}$ of the layers.

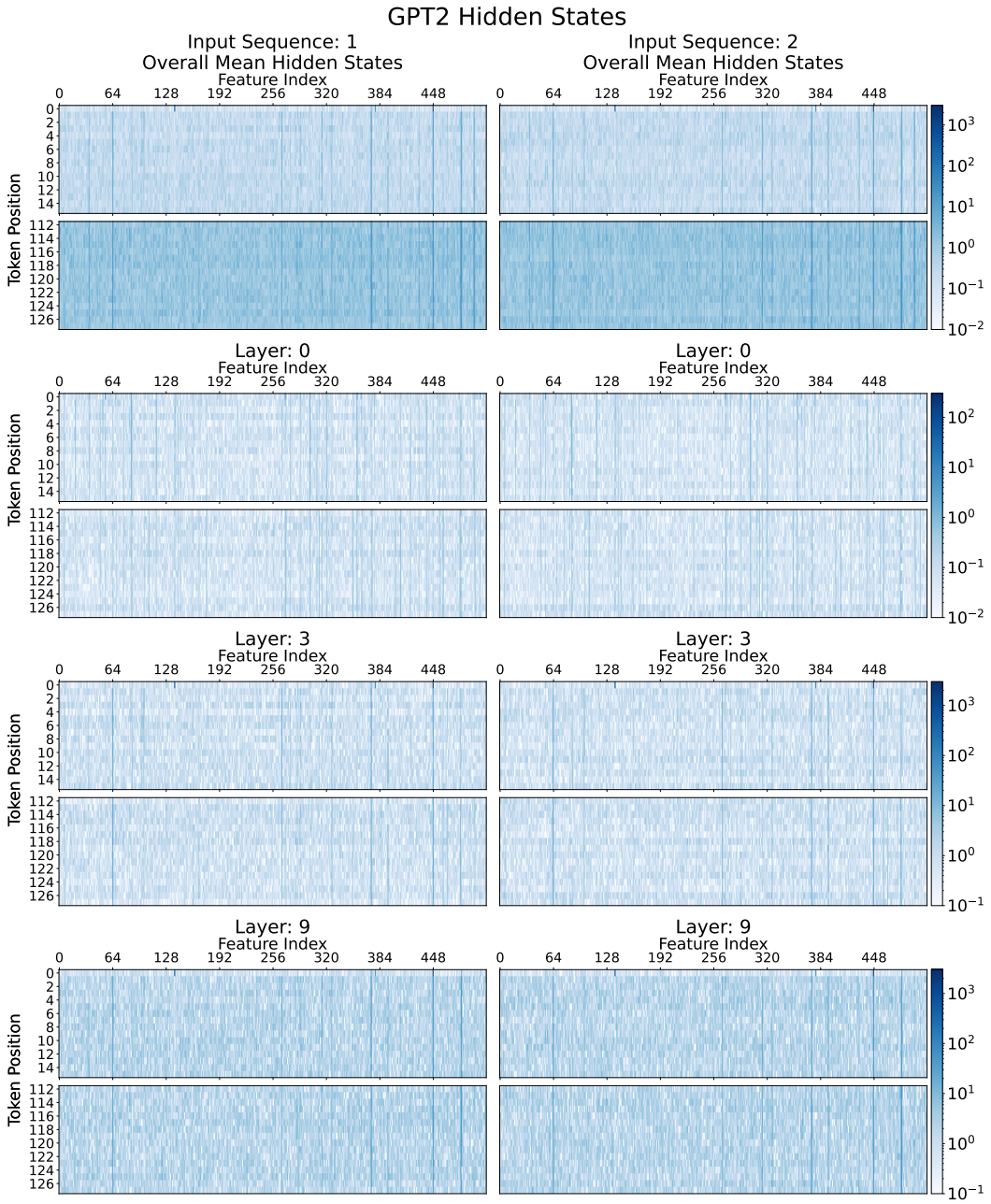

Figure 27: Example hidden state plots for a GPT2-Small model.

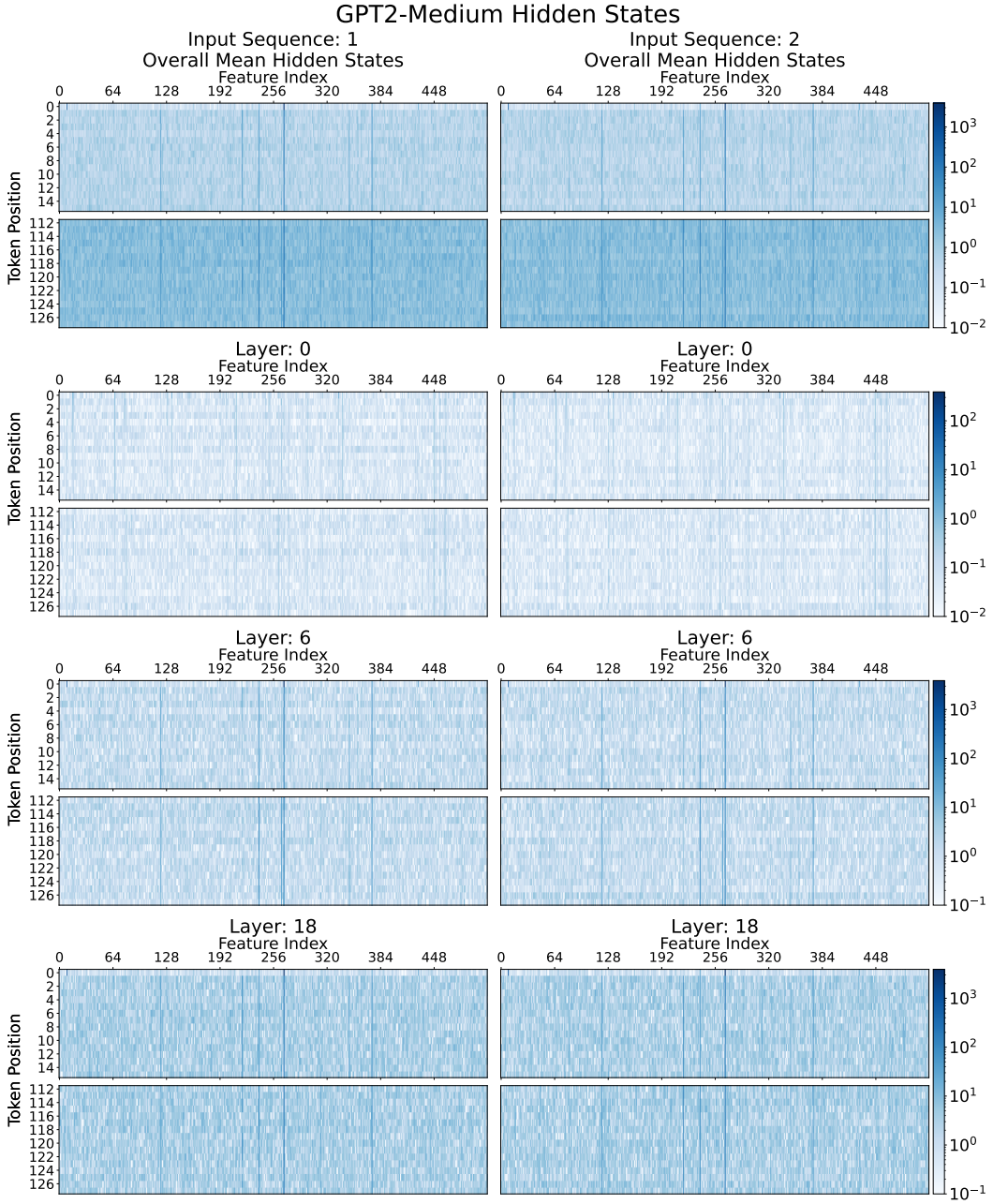

Figure 28: Example hidden state plots for a GPT2-Medium model.

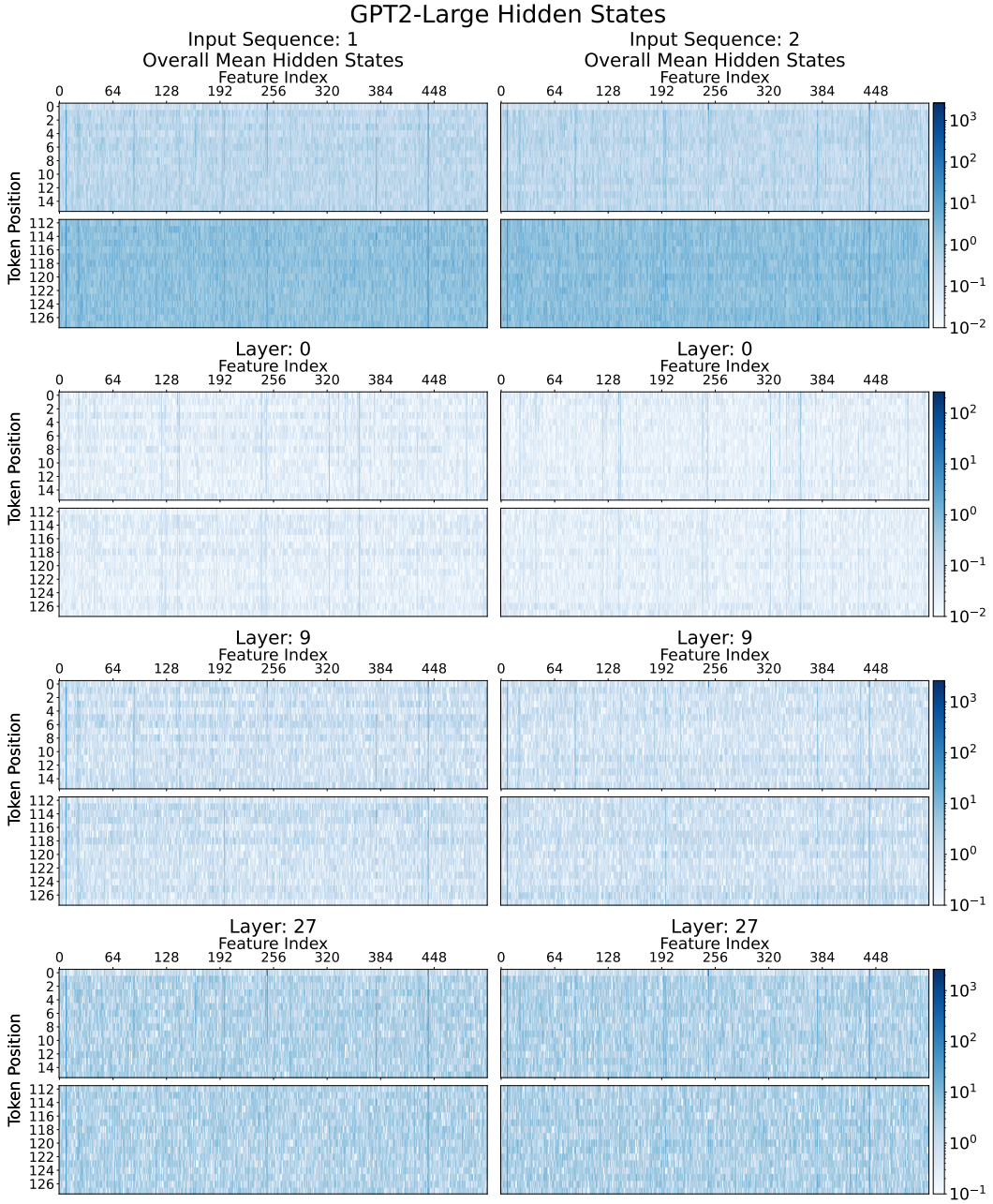

Figure 29: Example hidden state plots for a GPT2-Large model.

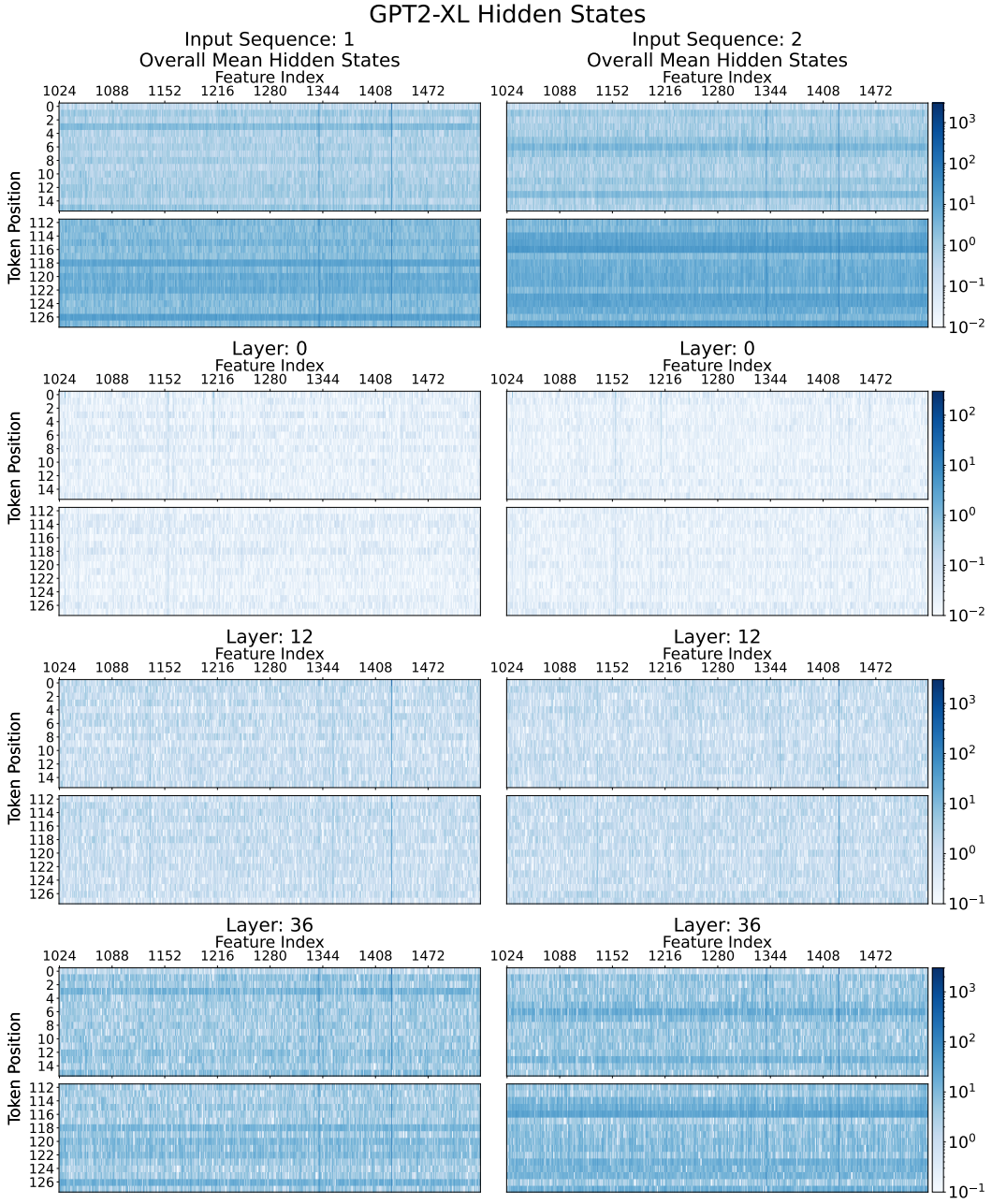

Figure 30: Example hidden state plots for a GPT2-XL model.

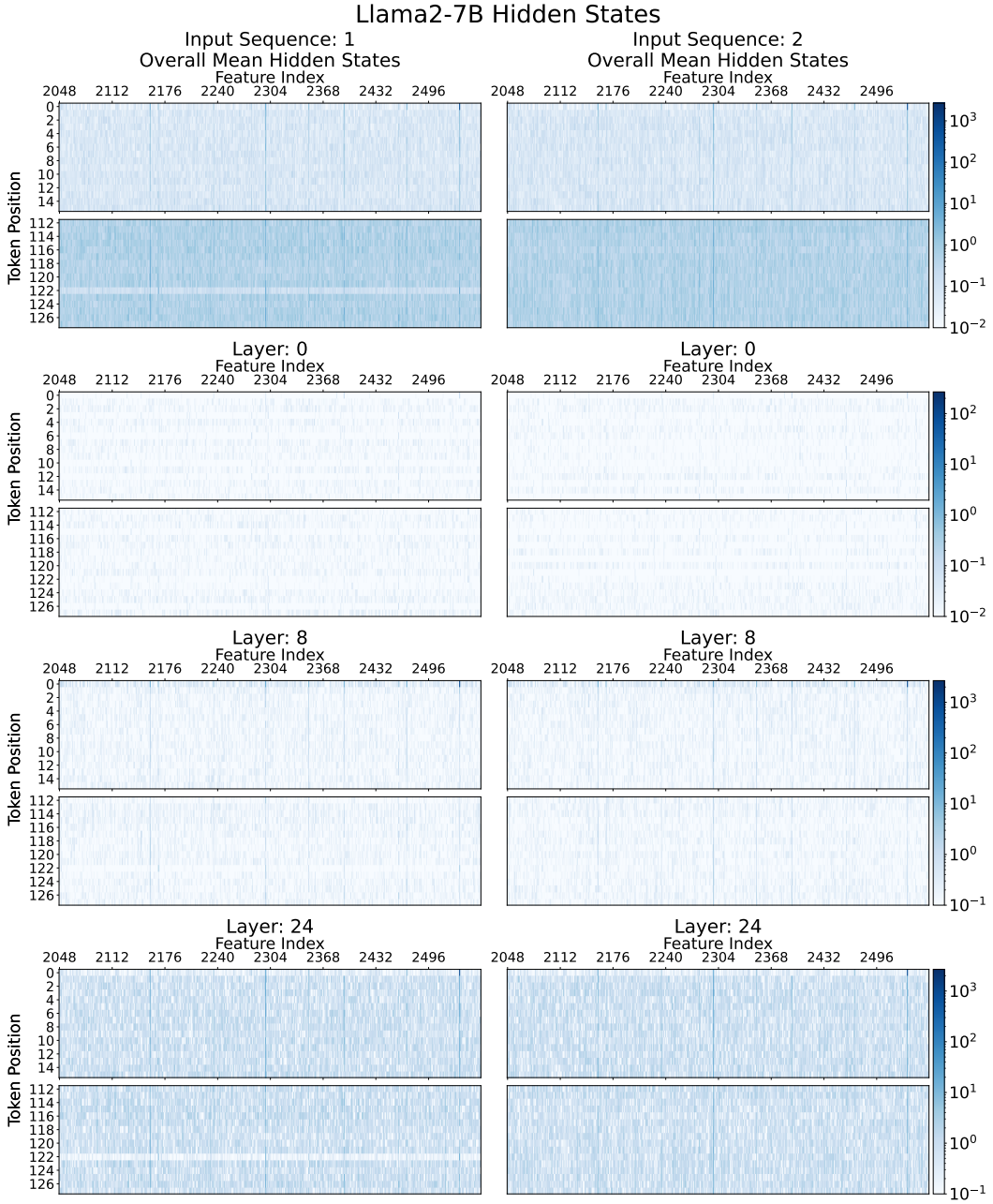

Figure 31: Example hidden state plots for a Llama2-7B model.

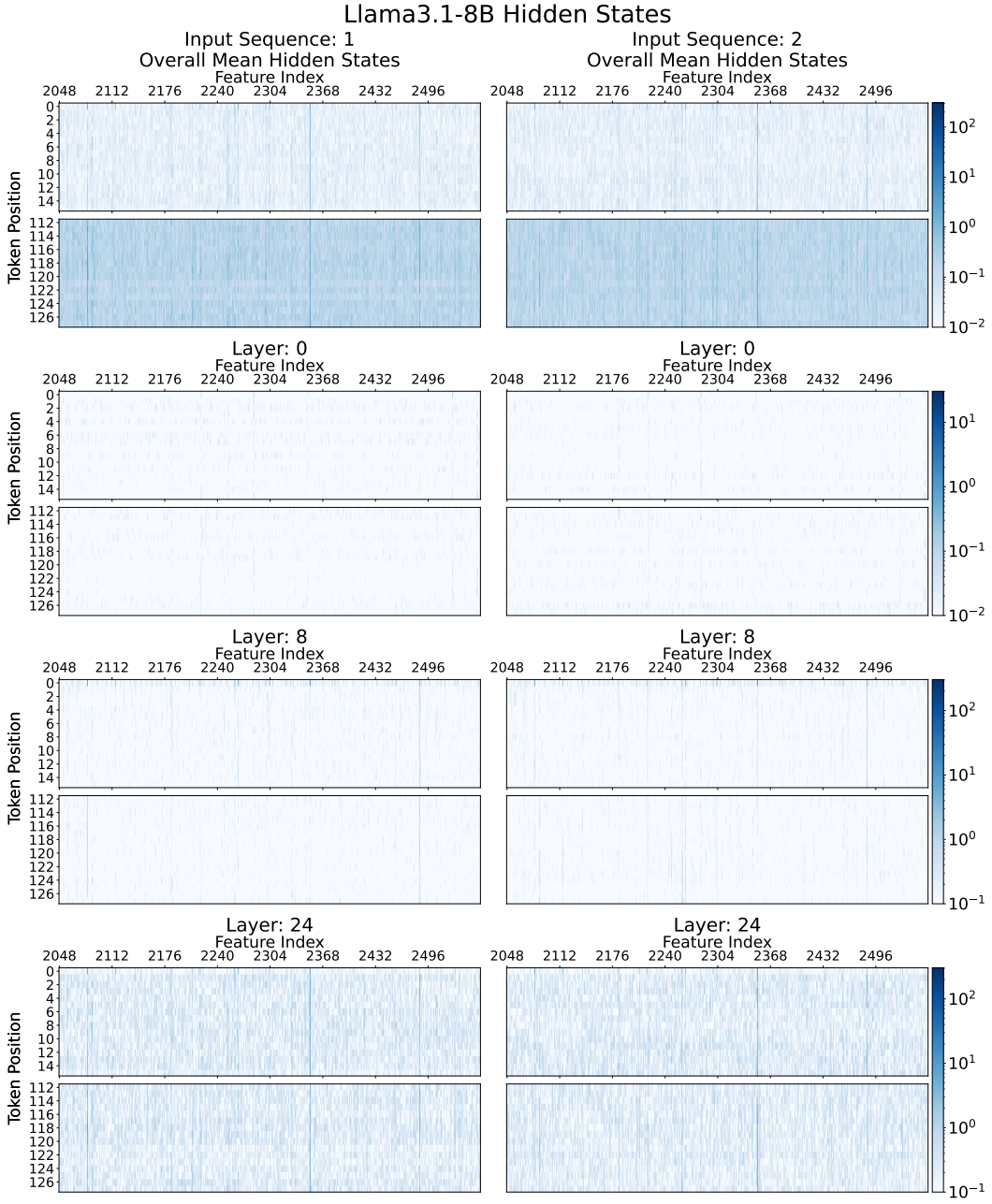

Figure 32: Example hidden state plots for a Llama3.1-8B model.

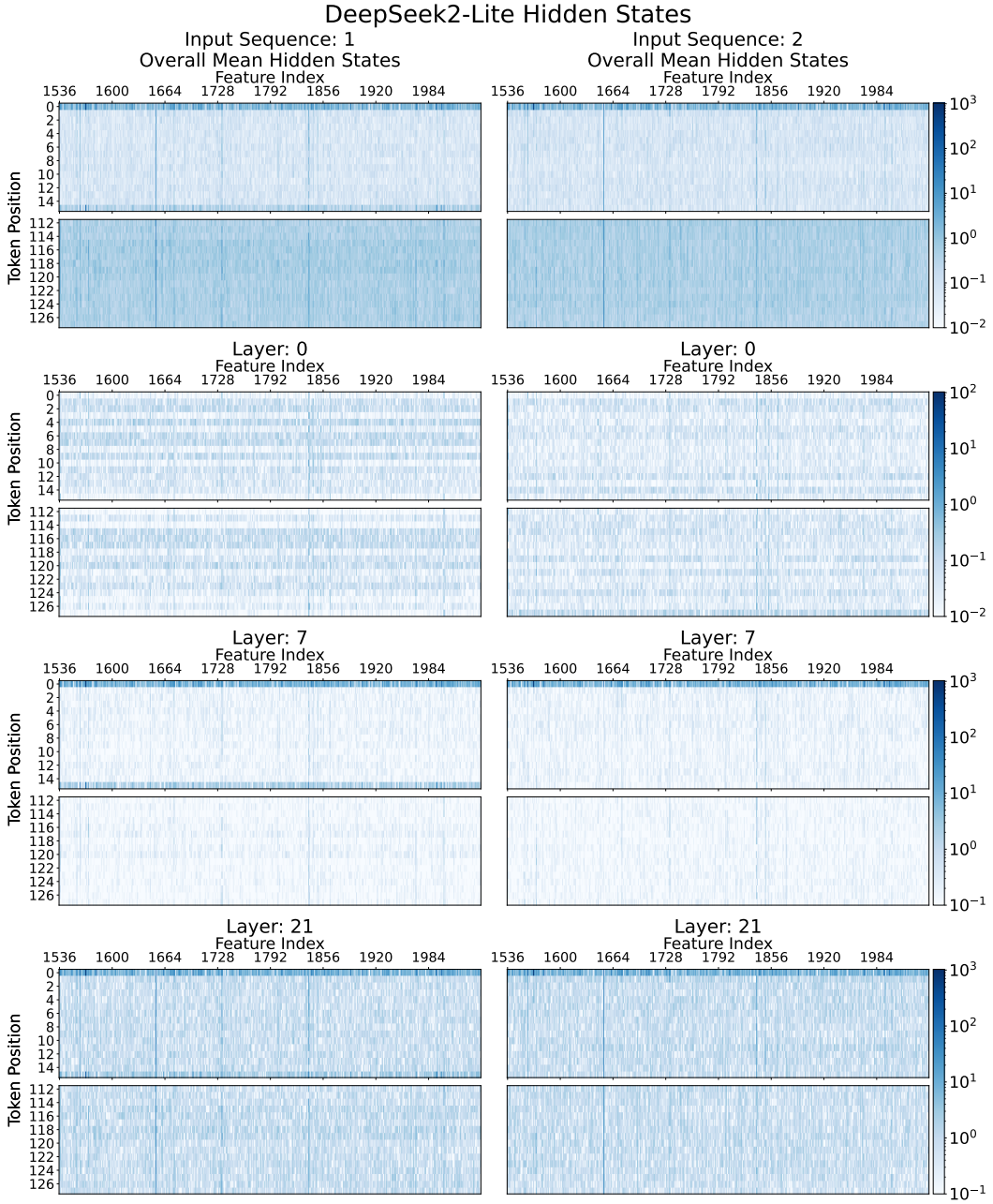

Figure 33: Example hidden state plots for a DeepSeekv2-Lite model.

K ATTENTION MAPS OF PRETRAINED MODELS

In this section, we present the attention maps of popular pretrained models. This shows how models establish attention patterns and how they persist after initial layers. This shows that generally after the first or second layer, first token attention dominance is highly established and persists across layers.

We show the mean across layers, the first layer, $\frac{1}{4}$ and $\frac{3}{4}$ of the layers—averaging over all heads in each case.

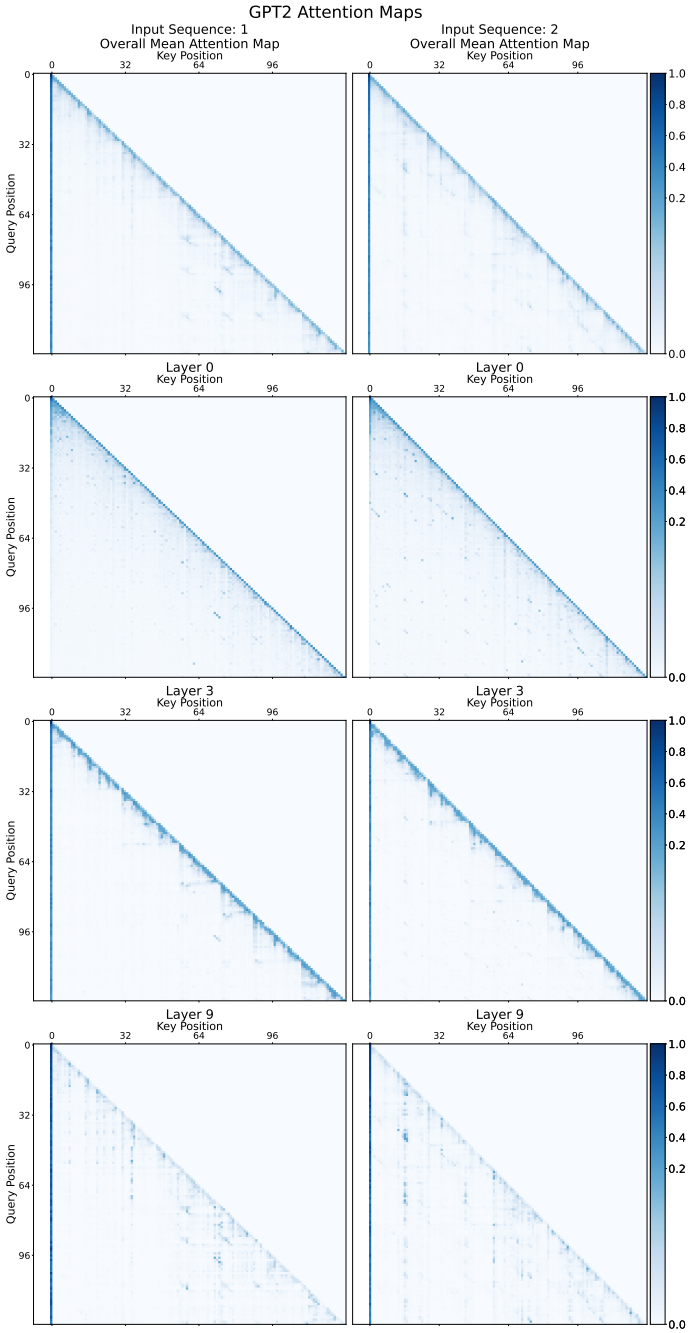

Figure 34: Example attention maps for a GPT2-Small model.

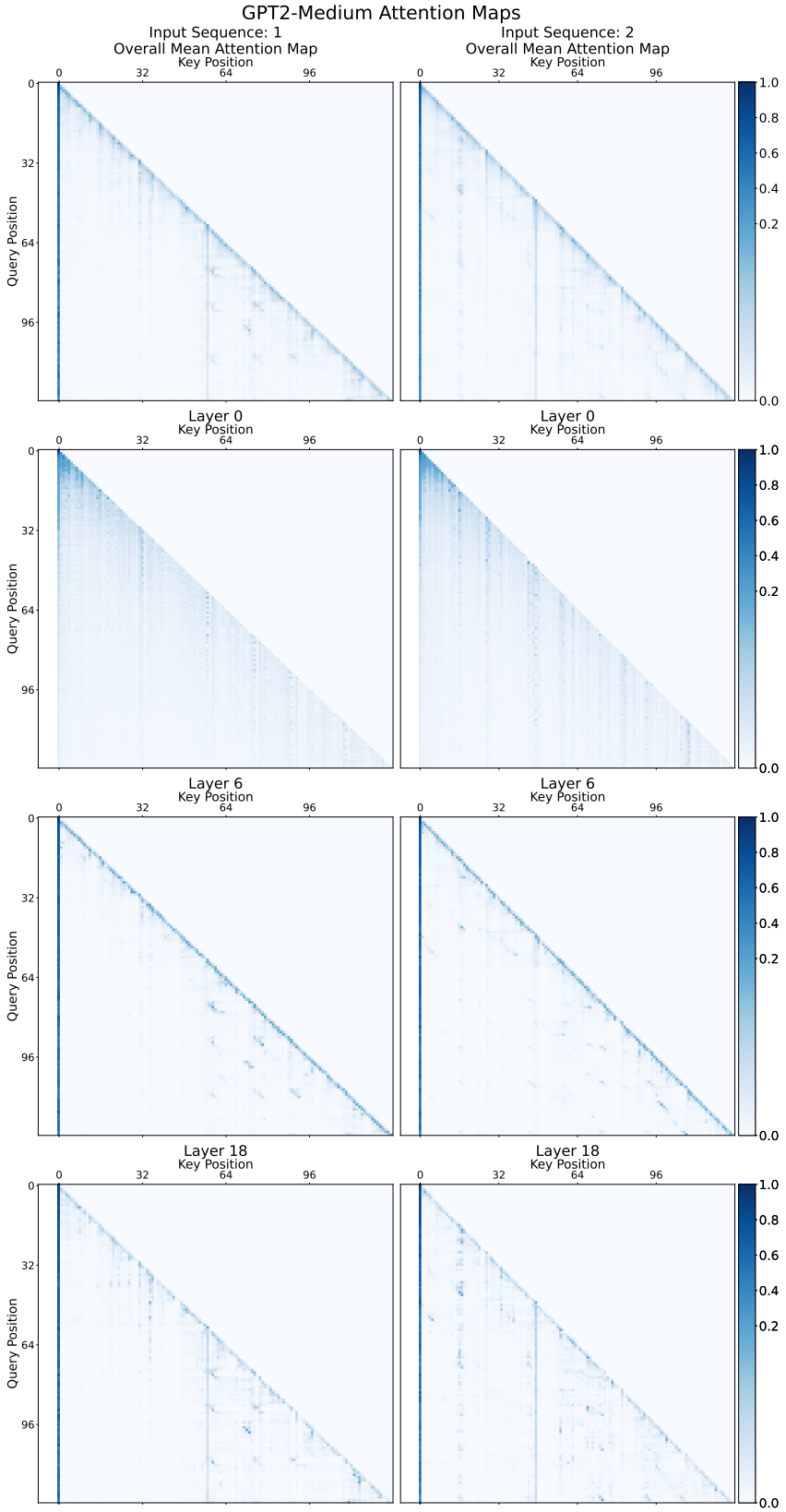

Figure 35: Example attention maps for a GPT2-Medium model.

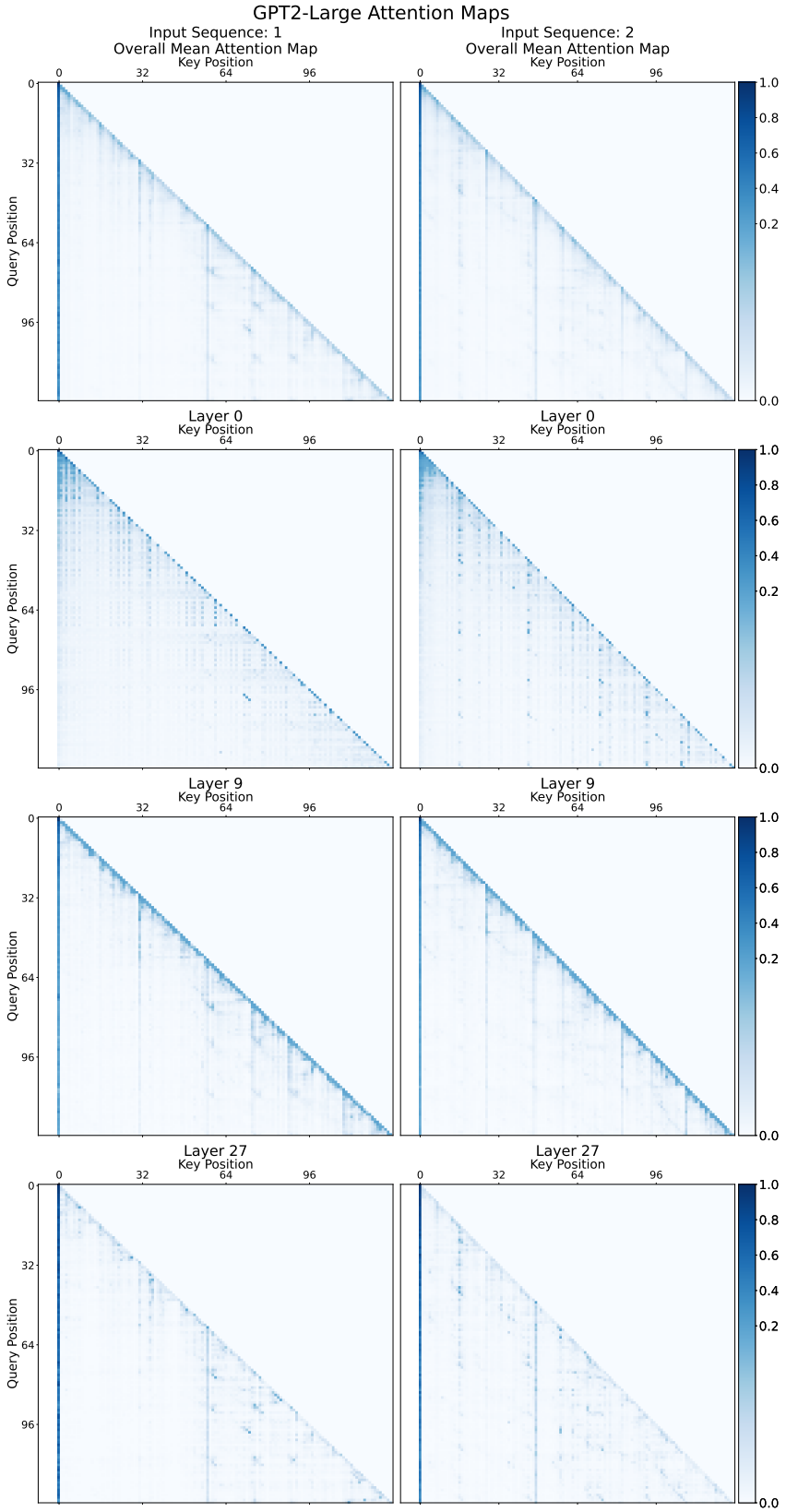

Figure 36: Example attention maps for a GPT2-Large model.

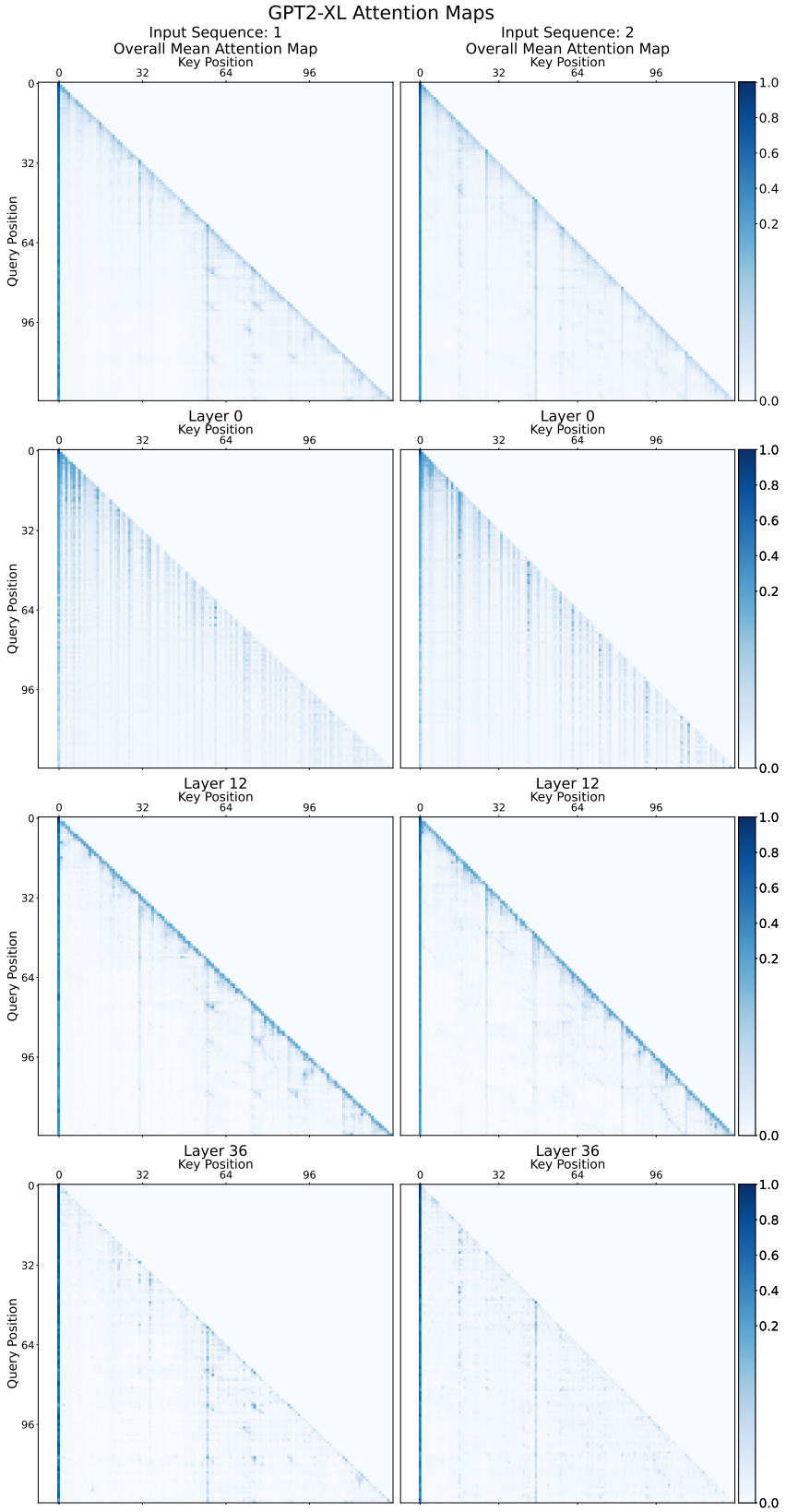

Figure 37: Example attention maps for a GPT2-XL model.

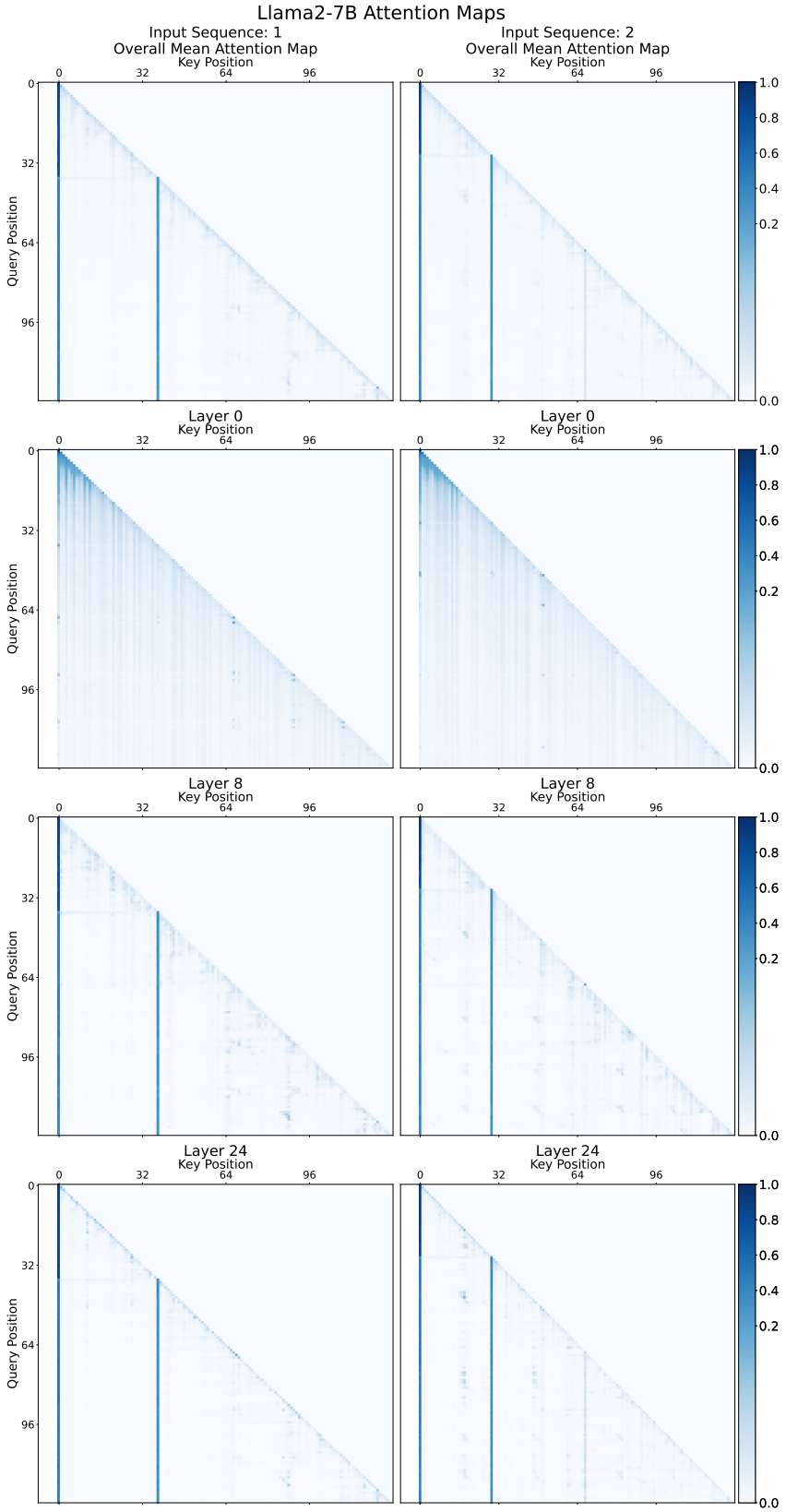

Figure 38: Example attention maps for a Llama2-7B model.

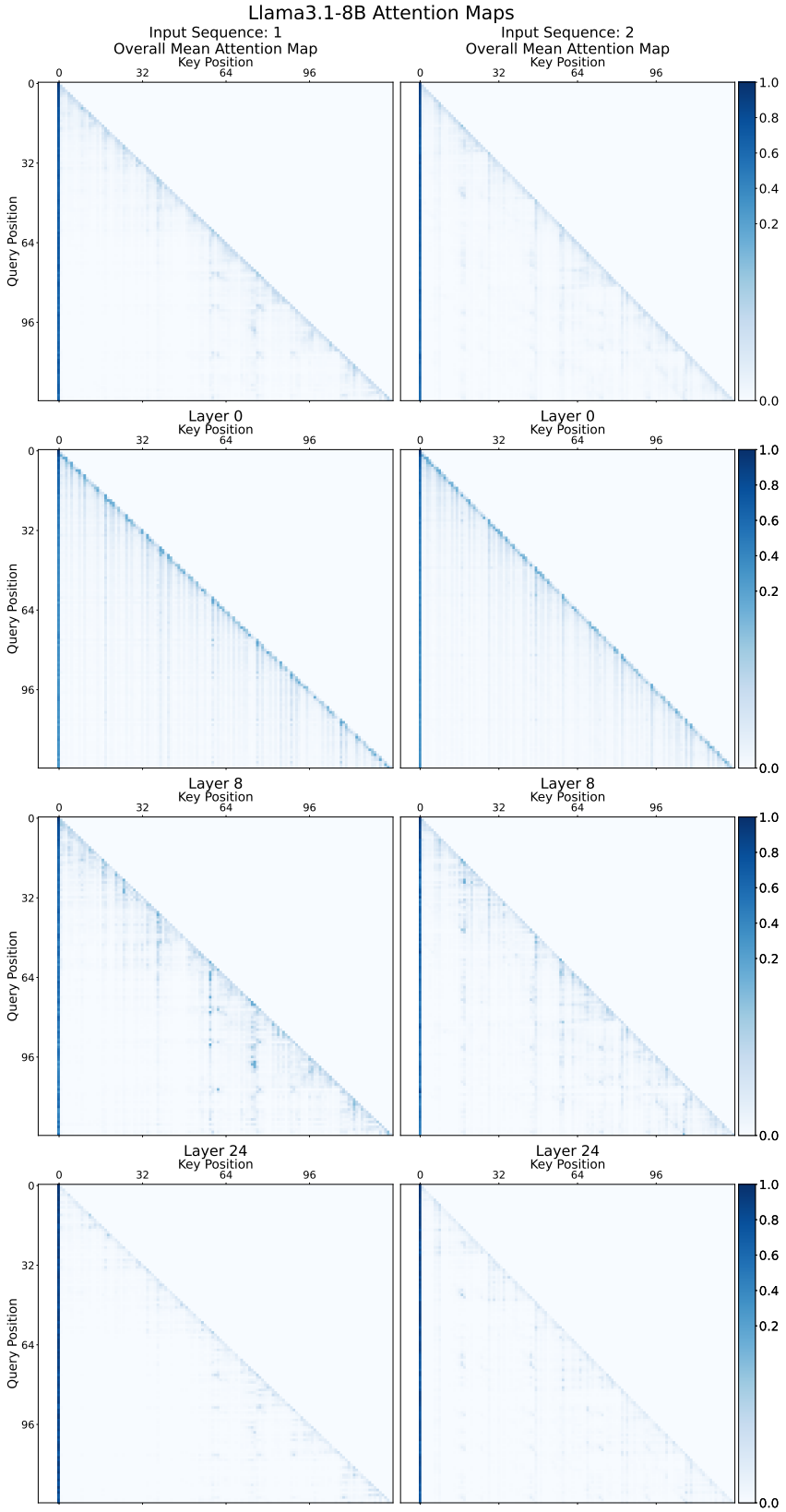

Figure 39: Example attention maps for a Llama3.1-8B model.

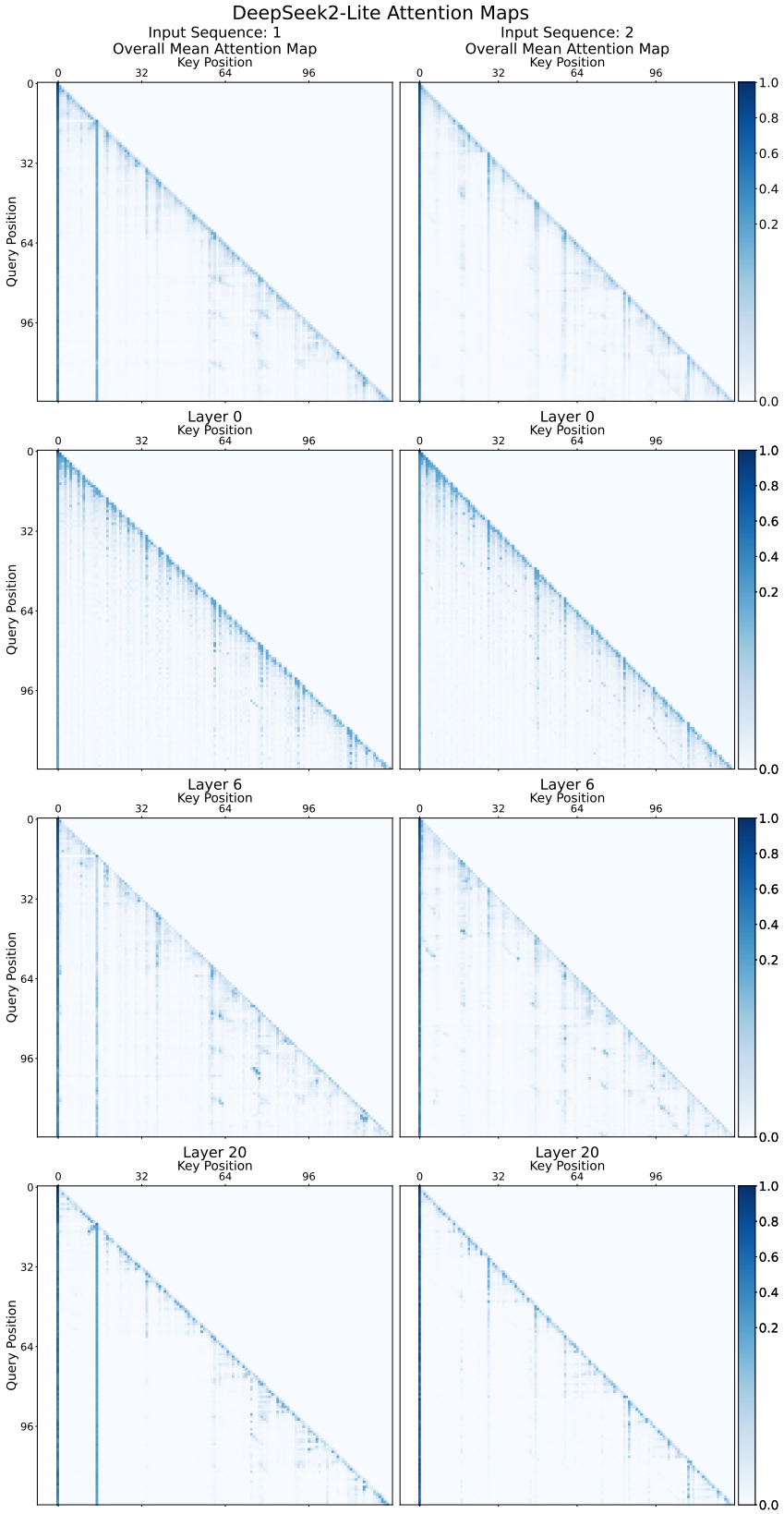

Figure 40: Example attention maps for a DeepSeekv2-Lite model.

L TRAINING CURVES

To demonstrate that our proposed methods, *i.e.*, replacing the canonical softmax function with *softmax-1* and using our proposed optimiser, *OrthoAdam*, do not negatively impact the training of large language models, we provide the training curves for our models here. One can observe that the training curves for models using either or both of our proposed changes are stable and converge to a similar loss value as the baseline models.

L.1 GPT2-60M

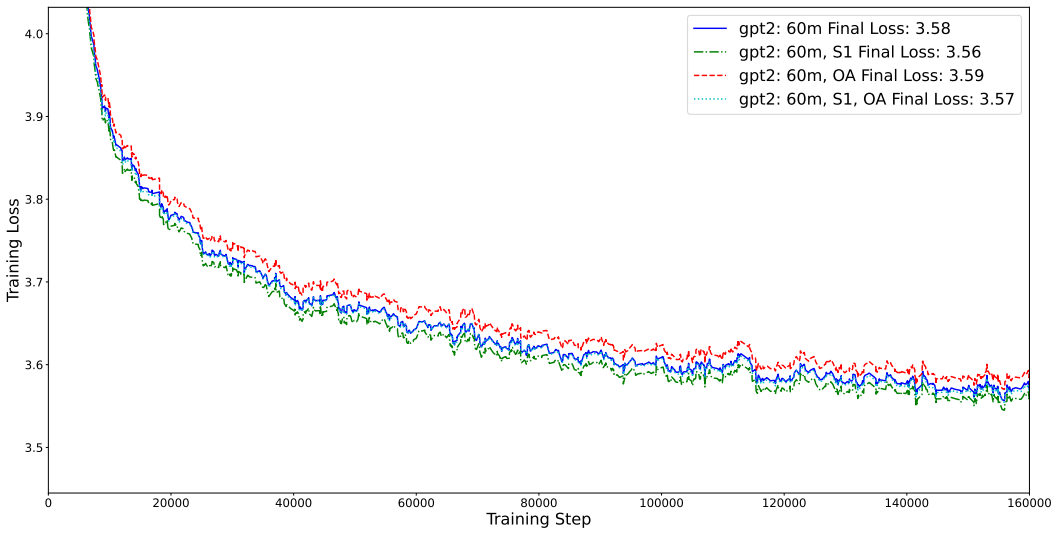

Figure 41: Training curves for GPT2-60M models with different optimisers and softmax functions. The models using *OrthoAdam* and *softmax-1* are stable and converge to a similar loss value as the baseline models. S1/OA denotes the model using softmax-1 and/or OrthoAdam.

L.2 GPT2-130M

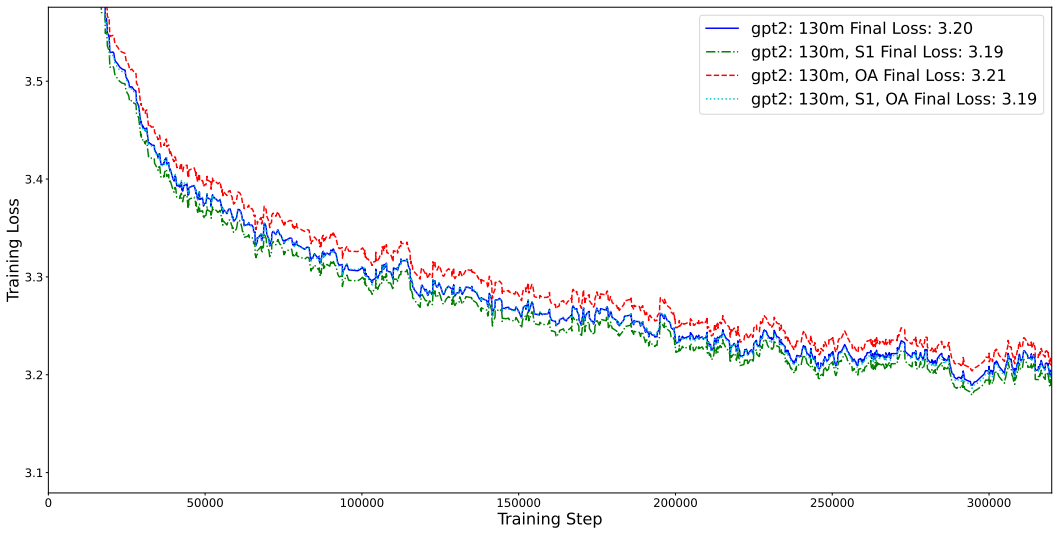

Figure 42: Training curves for GPT2-130M models with different optimisers and softmax functions. The models using *OrthoAdam* and *softmax-1* are stable and converge to a similar loss value as the baseline models. S1/OA denotes the model using softmax-1 and/or OrthoAdam.

L.3 GPT2-350M AND GPT2-1.4B

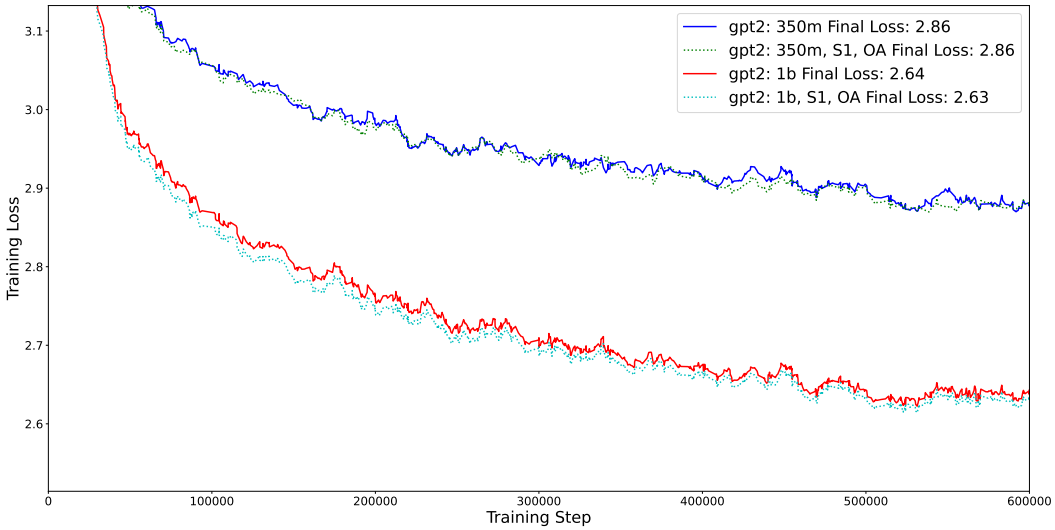

Figure 43: Training curves for GPT2-350M and GPT2-1.4B models with different optimisers and softmax functions. The models using *OrthoAdam* and *softmax-1* are stable and converge to a similar loss value as the baseline models. S1/OA denotes the model using softmax-1 and/or OrthoAdam.

L.4 LLAMA-130M

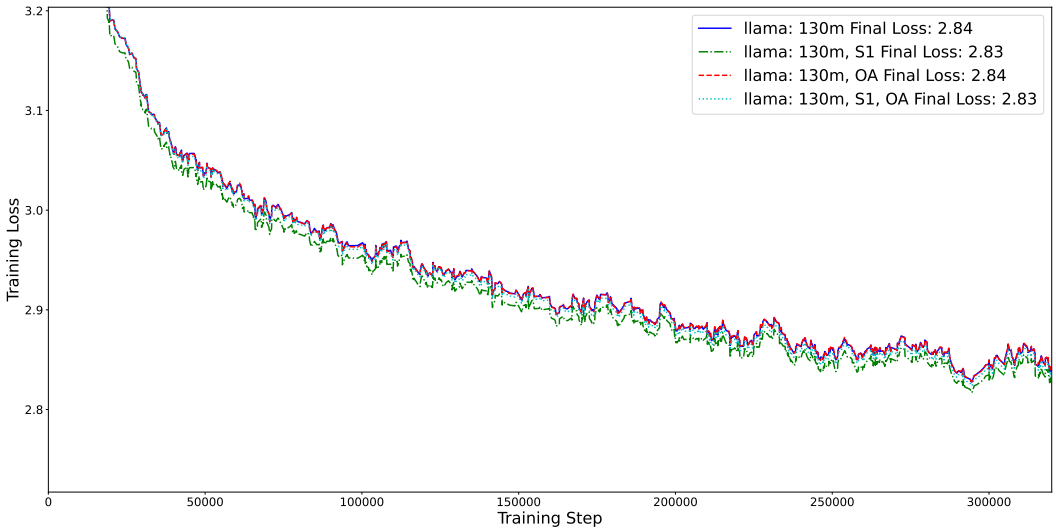

Figure 44: Training curves for Llama-130M models with different optimisers and softmax functions. The models using *OrthoAdam* and *softmax-1* are stable and converge to a similar loss value as the baseline models. S1/OA denotes the model using softmax-1 and/or OrthoAdam.