Spectral Conditioning of Attention Improves Transformer Performance

Hemanth Saratchandran
Australian Institute for Machine Learning
Adelaide University
hemanth.saratchandran@adelaide.edu.au

Simon Lucey
Australian Institute for Machine Learning
Adelaide University
simon.lucey@adelaide.edu.au

Abstract

We present a theoretical analysis of the Jacobian of a attention block within a transformer, showing that it is governed by the query, key, and value projections that define the attention mechanism. Leveraging this insight, we introduce a method that systematically alters the spectral properties of each attention layer to reduce the Jacobian's condition number, thereby improving the overall conditioning of the attention layers within a transformer network. We empirically show that this improved Jacobian conditioning translates to enhanced performance in practice. Our approach is simple, broadly applicable, and can be easily integrated as a drop-in replacement for a wide range of existing attention mechanisms. We validate its effectiveness across diverse transformer architectures and tasks, demonstrating consistent improvements in performance.

1 Introduction

Since its introduction in [34], the modern transformer has emerged as one of the most influential architectures in modern machine learning, yielding advances across a wide range of domains, including natural language processing (NLP) [34, 42, 41], computer vision [7, 20, 33, 5], and robotics [27, 23]. At the heart of its success lies the attention mechanism, which enables the model to capture complex relationships by computing pairwise interactions between all input tokens. By dynamically assigning importance to different tokens, attention allows transformers to model global dependencies, making them a foundational architecture in deep learning.

In this work, we investigate the conditioning of the Jacobian associated with the attention layer in transformer models. The conditioning of a matrix is measured by its condition number, the ratio of its largest to smallest singular value, with high values indicating ill-conditioning. Poorly conditioned Jacobians can hinder performance for gradient-based optimizers [25]. Recent advances in feedforward neural networks have shown that improving Jacobian conditioning can lead to better optimization and generalization performance. For instance, [30] proposed a weight normalization method to address ill-conditioning, while [19] demonstrated that reducing the condition number of the neural tangent kernel (NTK) leads to better convergence. Despite these insights, the conditioning of Jacobians in transformer models, particularly within attention layers remains largely unexamined. Addressing this gap is the central focus of our work.

We present a theoretical framework showing that the conditioning of the Jacobian in a attention layer is influenced by the conditioning of the queries, keys, and values matrices that constitute the attention matrix within that layer. Building on this insight, we introduce a method that modifies the spectrum of these matrices by adding a carefully designed correction term, yielding a new attention block that we call *spectral conditioned attention*, resulting in significantly improved Jacobian conditioning. Our analysis shows that the optimal correction can be derived using the singular value decomposition (SVD) of the query, key, and value matrices. However, computing the SVD directly can be prohibitively expensive in large-scale models. To address this, we also propose a computationally efficient approximation that achieves comparable improvements in conditioning with substantially lower overhead.

While a full theoretical proof linking our methodology to improved convergence in gradient-based optimization would require analyzing the NTK of a transformer, an extremely difficult task, we provide strong empirical evidence that our approach improves transformer performance across a wide range of applications, including image classification, object detection, instance segmentation, and natural language processing. A key advantage of our approach is its compatibility with a broad class of different attention mechanisms [4, 20, 33, 39, 6, 38], which we confirm empirically. Our main contributions include:

1. A theoretical framework that analyses the condition number of the Jacobian of attention showing its dependence on the condition number of the query, key and value matrices comprising an attention layer.

2. The introduction of spectral conditioned attention that adds a correction term to each of the query, key and value matrices resulting in a better conditioned attention Jacobian.

We validate the above contributions across a range of transformer architectures and applications, including image classification, object detection, instance segmentation, and natural language processing, consistently demonstrating improved performance in each setting.

2 Related Work

Conditioning. Several recent works have highlighted the critical role of conditioning in the optimization and generalization of neural networks. In [30], the authors focus on the condition numbers of weight matrices in feedforward architectures, showing that well-conditioned weights correlate with improved performance across tasks. They propose a preconditioning strategy that multiplies each weight matrix by a carefully constructed transformation, effectively reducing its condition number and facilitating more stable training. In a complementary line of work, [19] investigates neural network optimization from the perspective of the neural tangent kernel (NTK). The authors demonstrate that improving the conditioning of the NTK leads to better convergence, especially in the infinite-width regime where the NTK governs training dynamics [14]. In [21], it is shown that normalization improves the conditioning of various neural network architectures. Similarly, [40] demonstrate that normalizing attention weights enhances training convergence, which can be interpreted as an alternative means of improving the conditioning of attention layers. In a different approach, [1] shows that simply increasing the depth of feedforward networks can enhance their conditioning, thereby improving the effectiveness of gradient-based optimization methods. Recent work on transformers has examined conditioning from multiple perspectives, including tokenization, activation functions, and optimization [28, 17, 31, 15, 29], as well as within the context of fine-tuning [16, 3, 2, 12].

Attention. A range of methods have been proposed to improve the efficiency and scalability of Transformer architectures, often by rethinking the design of attention mechanisms or reducing computational complexity. For example, the Data-Efficient Image Transformer (DeiT) [33] introduces distillation tokens to achieve competitive performance in vision tasks without relying on large-scale datasets. The Cross-Covariance Image Transformer (XCiT) [4] reformulates attention by leveraging cross-covariances of spatial features, enabling efficient spatial interactions with reduced overhead. Swin Transformer [20] introduces a hierarchical architecture and a novel shifted window-based attention mechanism, significantly improving efficiency and effectiveness in vision tasks. DaViT [6] extends traditional vision transformers by incorporating both spatial and channel attention mechanisms, enhancing feature extraction across multiple dimensions. The Nyströmformer [38] takes a different approach by approximating full self-attention using the Nyström method, significantly lowering the quadratic complexity to near-linear while maintaining the core properties of attention, making it especially effective for long-sequence modeling.

3 Methodology

3.1 Preliminaries

In this section, we define the transformer architecture by describing the structure of a transformer layer, along with establishing notation for various mathematical quantities.

A transformer layer can be represented as a mapping

$$\mathbf{T} : \mathbb{R}^{N \times D} \to \mathbb{R}^{N \times D} \tag{1}$$

which is formally defined by

$$\mathbf{T}(X) = \mathbf{F}(\mathbf{A}(X) + X), \tag{2}$$

where $\mathbf{F}$ denotes a feed forward neural network with a residual connection, and $\mathbf{A}$ represents an attention mechanism. In general, layer normalization is added however for simplicity we omit layer normalization for this discussion.

For the theoretical framework we will primarily focus on self-attention, which is one of the most common forms of attention. Self-attention is composed of three learnable matrices, query $W_Q \in \mathbb{R}^{D \times d}$, key $W_K \in \mathbb{R}^{D \times d}$, and value $W_V \in \mathbb{R}^{D \times d}$, defined for an input sequence $X \in \mathbb{R}^{N \times D}$ by

$$\mathbf{A}(X) = \mathbf{softmax}(XW_Q W_K^T X^T)XW_V \tag{3}$$

where $\mathbf{softmax}$ is the softmax activation that acts row-wise on a matrix [26]. Note that then $A(X) \in \mathbb{R}^{N \times d}$. When training a transformer the self-attention parameters are given by the weight matrices W_Q, W_K and W_V. For further details on transformers readers may consult [26].

The self-attention map of a layer in a transformer $\mathbf{A}(X)$ has parameters given by those parameters in X from the previous layer and those given by W_Q, W_K and W_V that define $\mathbf{A}(X)$. Our work will consider the Jacobian of $\mathbf{A}(X)$ with respect to the parameters within the layer of $\mathbf{A}(X)$, namely W_Q, W_K and W_V. Therefore, when we speak of the Jacobian of $\mathbf{A}(X)$ it will be with respect to W_Q, W_K, W_V. We will denote this Jacobian by $J(\mathbf{A}(X))$ and note that it is defined by

$$J(\mathbf{A}(X)) = \left[\frac{\partial \mathbf{A}(X)}{\partial W_Q}, \frac{\partial \mathbf{A}(X)}{\partial W_K}, \frac{\partial \mathbf{A}(X)}{\partial W_V}\right]^T \tag{4}$$

Given a matrix $W \in \mathbb{R}^{m \times n}$ we denote the vectorization of W by $\mathbf{vec}(W) \in \mathbb{R}^{mn \times 1}$ [22]. Note that for such a matrix there is a transformation $T_{mn} \in \mathbb{R}^{mn \times mn}$ such that $T_{mn}\mathbf{vec}(W) = \mathbf{vec}(W^T)$ where W^T denotes the transpose of W. The matrix T_{mn} is known as a commutation matrix and is a permutation matrix [22]. The maximum singular value of a matrix W will be denoted by $\sigma_{\max}(W)$ and the minimum singular value by $\sigma_{\min}(W)$. We will use the standard terminology SVD to denote the singular value decomposition of a matrix. Given a vector $z \in \mathbb{R}^n$ the notation $||z||_2$ will denote the vector 2-norm of z.

3.2 Main Theorems

In this section, we analyze the conditioning of the Jacobian of the self-attention matrix in a transformer. We will show that the condition number of the Jacobian depends on the condition number of the queries, keys and values matrices. This then motivates our key insight: reducing the condition number of the queries, keys and values matrices can lead to a lower condition number for the Jacobian of the self-attention matrix in each layer of a transformer, which we empirically verify in Section 4. We have chosen to focus our theory on self-attention so as to yield concrete formulas. However, our theory goes through for more general attention mechanisms, see Section A.1 for a discussion. Proofs of all lemmas and theorems in this section can be found in Section A.1.

Definition 3.1. Let A be an $N \times d$ matrix of full rank. The condition number of A, denoted by κ, is defined as

$$\kappa(A) = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)} \tag{5}$$

where $\sigma_{\max}(A)$ denotes the maximum singular value of A and $\sigma_{\min}(A)$ the minimum singular value of A, which we know is non-zero as A is full rank.

Our objective is to analyze the condition number of the self-attention block within a transformer. We demonstrate that the condition number of its Jacobian is influenced by the condition numbers of the query, key, and value weight matrices. By appropriately conditioning these matrices, we show that the Jacobian's condition number can be significantly reduced, leading to a more stable and well-conditioned attention mechanism suitable for applications.

To establish this, we first examine the derivatives of the self-attention block with respect to the parameters W_Q, W_K, and W_V, yielding Theorem 3.3 which is then used to establish a condition number bound on the Jacobian given in Theorem 3.4. As a starting point, Theorem 3.2 presents the derivative of the $\mathbf{softmax}$ function, which plays a central role in the analysis.

Lemma 3.2. *Let $\Lambda : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ denote the function $\Lambda(z) = Diag(z) - z \cdot z^T$. We then have that*

$$\frac{\partial \mathbf{softmax}}{\partial x}(z) = \Lambda(\mathbf{softmax}(z)). \tag{6}$$

Theorem 3.3. *Let $\mathbf{A}(X)$ denote a self-attention matrix with input X as defined by Equation* (3)*. Then*

$$\frac{\partial \mathbf{A}(X)}{\partial W_Q} = (W_V^T X^T \otimes I_N)\Big(\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))\Big)(XW_K \otimes X) \tag{7}$$

$$\frac{\partial \mathbf{A}(X)}{\partial W_K} = (W_V^T X^T \otimes I_N)\Big(\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))\Big)(X \otimes XW_Q)\cdot T_{Dd} \tag{8}$$

$$\frac{\partial \mathbf{A}(X)}{\partial W_V} = I_d \otimes \mathbf{softmax}(XW_QW_K^TX^T)X \tag{9}$$

where Λ is defined in Theorem 3.2 and T_{Dd} is the commutation matrix satisfying $T_{Dd}\mathbf{vec}(W_K) = \mathbf{vec}(W_K^T)$ (see Section 3.1).

Theorem 3.4. *Let $\mathbf{A}(X)$ denote a self-attention matrix, as defined in Equation* (3)*, with input X and let $J(\mathbf{A}(X))$ denote its Jacobian with respect to the parameter matrices W_Q, W_K and W_V as defined in Equation* (4)*. Assume that $J(\mathbf{A}(X))$ has full rank so that $\kappa(J(\mathbf{A}(X)))$ is finite. Then*

$$\kappa(J(\mathbf{A}(X))) \leq \kappa(X)^3\kappa\big(\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))\big)\kappa(W_V)\big(\kappa(W_Q)+\kappa(W_K)\big) \tag{10}$$
$$+\, \kappa(X)\kappa(\mathbf{softmax}(XW_QW_K^TX^T))$$

where Λ is defined in Theorem 3.2.

Theorem 3.4 shows that $\kappa(J(\mathbf{A}(X)))$ is bounded above by a sum of two terms:

$$\kappa(X)^3 \cdot \kappa\big(\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))\big)\kappa(W_V)\big(\kappa(W_Q)+\kappa(W_K)\big) \tag{11}$$

$$\kappa\big(\mathbf{softmax}(XW_QW_K^\top X^\top)\big) \tag{12}$$

Observation: Directly conditioning the Jacobian of the self-attention map with respect to the parameters W_Q, W_K, and W_V during training would require computing the Jacobian at every iteration, which is prohibitively expensive. However, Theorem 3.4 offers a more efficient alternative: reducing the upper bound in Equation (10). Since we have access to W_Q, W_K, and W_V, lowering their condition numbers reduces the quantity in Equation (11), thereby tightening the bound in Equation (10). The following theorem shows how to do this.

Theorem 3.5. *Let W_Q, W_K and W_V denote the parameters of a self-attention matrix $\mathbf{A}(X)$. Then there exist matrices C_Q, C_K and C_V such that*

$$\kappa(W_Q + C_Q),\ \kappa(W_K + C_K),\ \kappa(W_V + C_V) \leq 2 \tag{13}$$

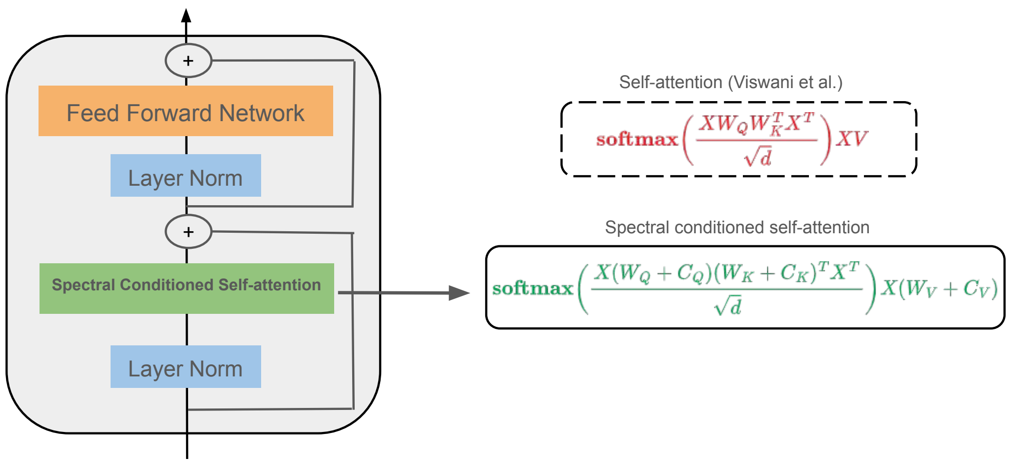
Feed Forward Network
Layer Norm
Spectral Conditioned Self-attention
Layer Norm
+
+
Self-attention (Viswani et al.)
$\mathbf{softmax}\left(\frac{XW_QW_K^TX^T}{\sqrt{d}}\right)XV$
Spectral conditioned self-attention
$\mathbf{softmax}\left(\frac{X(W_Q+C_Q)(W_K+C_K)^TX^T}{\sqrt{d}}\right)X(W_V+C_V)$

Figure 1: An illustration of spectrally conditioned self-attention within a transformer layer. At each layer, the self-attention weights W_Q, W_K, and W_V are modified by adding correction terms C_Q, C_K, and C_V, respectively. The correction terms C_Q, C_K, and C_V are initialized before training using Theorem 3.8 and remain fixed throughout training.

Overview of proof of Theorem 3.5: The same proof works for W_Q, W_K and W_V. Thus we will give the main steps for W_Q. As in Section 3.1 let $W_Q \in \mathbb{R}^{D\times d}$.

1. Using the SVD write $W_Q = USV^T$ where $U \in \mathbb{R}^{D\times D}$ is the left singular vectors, $S \in \mathbb{R}^{D\times d}$ is the matrix of singular values on the main diagonal and $V \in \mathbb{R}^{d\times d}$ are the right singular vectors.
2. Define a new matrix $C_Q = U\overline{S}V^T$ where $\overline{S}$ has $\sigma_{\max}(W_Q)$ on its main diagonal and zeros everywhere else.
3. The matrix $W_Q + C_Q$ then satisfies $\kappa(W_Q + C_Q) \leq 2$.

3.3 Spectral Conditioned Attention

In Theorem 3.5, we showed that adding correction terms C_Q, C_K, and C_V to the weight matrices W_Q, W_K, and W_V, respectively, significantly reduces their condition numbers $\kappa(W_Q)$, $\kappa(W_K)$, and $\kappa(W_V)$. This motivates the following definition.

Definition 3.6. We define *spectral conditioned attention* by

$$\mathbf{SpecA}(X) := \mathbf{softmax}(X(W_Q + C_Q)(W_K + C_K)^TX^T)X(W_V + C_V) \tag{14}$$

where C_Q, C_K and C_V are the correction terms given in Theorem 3.5.

We call the process of adding correction matrices C_Q, C_K and C_V to W_Q, W_K and W_V respectively to lower their condition number as spectral conditioning.

Remark 3.7. Theorem 3.5, together with Theorem 3.4, shows that spectrally conditioned attention reduces the upper bound in Equation (10), thereby yielding a tighter bound on the condition number of the Jacobian of the self-attention layer.

The proof overview for Theorem 3.5 showed the correction term C_Q (and C_K, C_V) is derived by taking an SVD. Computing the SVD of each of W_Q, W_K, W_V at each iteration of training would lead to a significant memory overhead. We therefore provide an implementation friendly form of spectral conditioned attention, motivated by the following theorem (proof given in Section A.1).

Theorem 3.8. *Let $A \in \mathbb{R}^{m \times n}$ and let $I_k \in \mathbb{R}^{m \times n}$ be the matrix that has 1 on its main $k \times k$ diagonal, where $k = \min\{m, n\}$, and zeros elsewhere. Let $\lambda \geq 2$ be a fixed constant and assume that $\frac{\sigma_{\max}(A)+\lambda}{\lambda-\sigma_{\min}(A)} \leq \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}$. Then $\kappa(A + \lambda I_k) < \kappa(A)$.*

The key difference between Theorem 3.5 and Theorem 3.8 when applied to W_Q, W_K, and W_V is that Theorem 3.5 guarantees a condition number strictly less than 2, whereas Theorem 3.8 only ensures a reduction relative to the original i.e. $\kappa(W_Q + \lambda I_k) < \kappa(W_Q)$, and similarly for W_K and W_V. However, the advantage of Theorem 3.8 over Theorem 3.5 is that the correction term λI_k does not require computing the SVD of W_Q, W_K or W_V and hence is much more memory efficient.

Implementation: We initialized correction matrices C_Q, C_K, and C_V as λI_k with $\lambda = 10$ (see Section A.2.1 for an ablation study on λ), and used spectral conditioned attention where the modified weights were $W_Q + C_Q$, $W_K + C_K$, and $W_V + C_V$. These correction matrices are fixed during training and are not updated, incurring no additional memory overhead during backpropagation. An overview of the resulting attention architecture is shown in Figure 1.

For a comparison of spectral conditioned attention using Theorem 3.5 and Theorem 3.8 on vision transformers see Section A.2.1.

4 Experiments

In this section, we evaluate our insights from Section 3 on a variety of transformer applications. For each application we will consider an original transformer architecture used within the literature and compare it with one employing spectral conditioning on its attention layer.

Implementation. In all cases, we used the implementation described in Section 3.3, where fixed correction terms C_Q, C_K, and C_V are added to the query, key and value matrices W_Q, W_K and W_V respectively within each attention layer, with $\lambda = 10$ (see Section A.2.1 for an ablation on λ). These terms are not updated during training and therefore do not introduce any additional trainable parameters. For more details see Section A.2.

4.1 Image Classification

Vision transformers. We applied spectral conditioned attention to a variety of modern vision transformers: the original vision transformer (ViT-B) [7], Swin transformer (Swin-B) [20], Cross-Covariance image transformer (XCiT-M) [4], Data efficient image transformer (DeiT-B) [33], Dual attention vision transformer (DaViT-B) [6] for image classification on ImageNet-1k. Each of these vision transformers uses a different attention block to the standard self-attention used in the ViT-B architecture. However, spectral conditioned attention (see Theorem 3.6) works by adding a correction term to each query, key and value matrix and hence can easily be plugged into these modern variants of self-attention. We include these latter models to illustrate that, while our theoretical analysis in Section 3 centers on self-attention, spectral conditioned attention can be readily incorporated into modern transformer architectures, including those with more complex designs.

Validating the theory. We validate the theoretical results from Section 3 on a ViT-B model trained on the ImageNet-1k dataset. For each experiments we ran five trials and took the mean and standard deviations. For each attention head in every transformer layer, we compute the minimum singular value of the query (W_Q), key (W_K), and value (W_V) weight matrices, and then average these values across all heads and layers. To improve the conditioning, we add correction terms C_Q, C_K, and C_V as described after Theorem 3.8, resulting in spectral conditioned attention matrices $W_Q + C_Q$, $W_K + C_K$, and $W_V + C_V$. The left figure in Figure 2 shows the average minimum singular value during training and in each case we see that the minimum singular value of the corrected weights is higher. Similar plots for the maximum singular value is shown in Section A.2.1. The middle plot in Figure 2 compares the average condition numbers of the original and spectral conditioned query, key and value matrices, demonstrating that the latter are substantially better conditioned, thus empirically verifying the claim of Theorem 3.8. Finally, the right plot reports the average condition number of the

Jacobian of the self-attention matrices, both for the standard ViT-B and for the spectral conditioned version. It also includes the theoretical upper bound from Theorem 3.4. While the spectral correction was motivated by this theoretical bound, the results clearly show that it leads to better-conditioned Jacobians in practice, supporting its use in designing more stable attention mechanisms. Note that each plot shows the mean over five trials. Standard deviations cannot be seen due to the log plot of the y-axis but can be found in separate plots in Section A.2.1. We repeated the experiment using the XCiT-M transformer [4]. As shown in Figure 3, the results exhibit the same trend observed with ViT-B, further supporting the theoretical findings in Section 3.

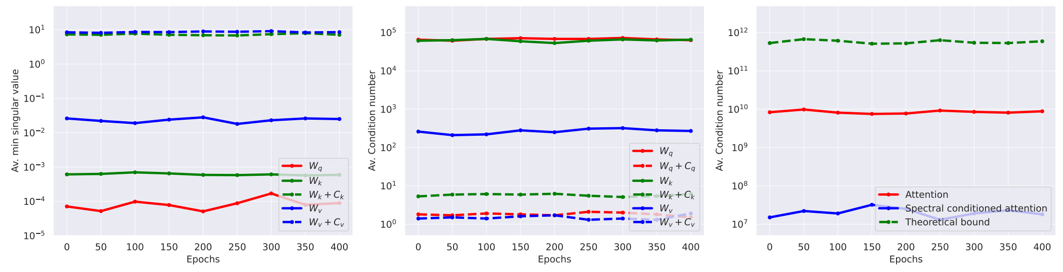

Figure 2: Analysis for ViT-B. **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) and their spectrally conditioned counterparts ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) throughout training. **Middle:** Condition numbers of W_Q, W_K, and W_V, and their spectrally conditioned forms during training. **Right:** Average condition number of the self-attention Jacobian over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10).

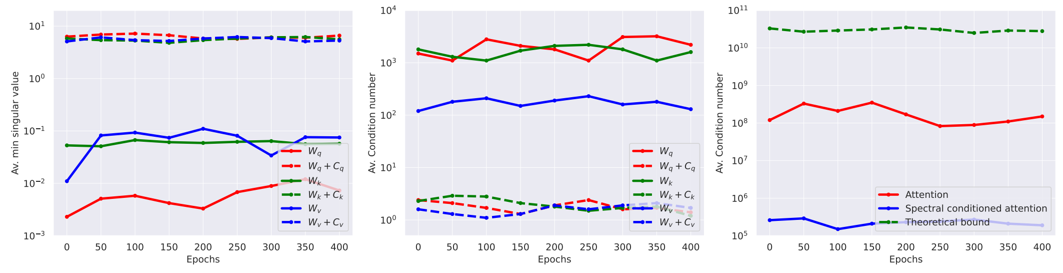

Figure 3: Analysis for XCiT-M. **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) and their spectrally conditioned counterparts ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) throughout training. **Middle:** Condition numbers of W_Q, W_K, and W_V, and their spectrally conditioned forms during training. **Right:** Average condition number of the self-attention Jacobian over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10).

Rank assumption. In the statement of Theorem 3.4, we assumed that the Jacobian is full rank, ensuring the condition number in Equation (10) remains finite. Empirically, we observed this assumption holds across all vision transformer architectures considered. For both ViT-B and XCiT-M, the final plots in Figure 2 and Figure 3 confirm that the average condition number remains finite and stable throughout training, indicating no rank deficiency in the Jacobian of each layer.

Results on ImageNet-1k. We trained two versions of each transformer: a baseline model and one incorporating spectral conditioned attention. For each base architecture, we observed that the minimum singular values of the query, key, and value matrices remained below 1 throughout training (see Figure 2 for ViT-B and Figure 3 for XCiT-M), allowing us to apply Theorem 3.8 and implement spectral conditioning as described in Section 3.3. All models were trained using the AdamW optimizer, see Section A.2.1. We trained each model five times with five different random seeds. The final results, summarized in Table 1, show the mean test accuracy with the standard deviation in brackets. We observe that in every case, the spectral conditioned variant achieves higher test accuracy than its unmodified counterpart. Hardware, memory cost and FLOPS analysis is shown in Section A.2.1. An ablation on λ is given in Section A.3. Furthermore, spectral conditioning was applied together with layer normalization as shown in Figure 1. A discussion comparing the two is given in Section A.3.

Table 1: Comparison of Vision Transformers with their original attention architecture versus one with spectrally conditioned attention (Spec. cond.), pre-trained on the ImageNet-1k dataset. We report Top-1% classification accuracy. In each case, spectral conditioning improves performance over the original attention mechanism.

	ViT-B	DeiT-B	Swin-B	XCiT-M	DaViT-B
Original	80.7 ($\pm$0.41)	81.6 ($\pm$0.30)	83.4 ($\pm$0.28)	82.6 ($\pm$0.39)	84.3 ($\pm$0.26)
Spec. cond.	81.7 ($\pm$0.38)	82.6 ($\pm$0.32)	84.1 ($\pm$0.25)	83.5 ($\pm$0.35)	84.9 ($\pm$0.21)

4.2 Object Detection and Instance Segmentation

In this section, we evaluate our methodology on two downstream tasks: object detection and instance segmentation. The goal is to assess the effectiveness of spectral conditioned attention in a fine-tuning setting. We begin by pretraining an XCiT architecture [4] on the ImageNet-1k dataset, followed by fine-tuning on the COCO 2017 dataset [18]. The XCiT models are used as backbones within the Mask R-CNN framework [13], enhanced with a Feature Pyramid Network (FPN) to improve multi-scale feature representation. To integrate XCiT with FPN, we modify its column structure to extract intermediate features from multiple layers. Specifically, we use the 12-layer XCiT-Small (XCiT-S) adapting their its strides from the original fixed stride of 16 to $[4, 8, 16, 32]$ to match the FPN hierarchy. Downsampling is performed via max pooling, and upsampling is achieved using a single transposed convolution layer. We conduct this procedure for both the original XCiT-S and one incorporating spectral conditioned attention blocks.

Downstream results. The results for both object detection and instance segmentation are shown in Table 2. We ran five trials each with a different seed and plotted the mean metric with standard deviation in brackets. The reported metrics include AP^b (Average Precision for bounding box predictions), $AP^b_{50/75}$ (Average Precision at IoU thresholds of 0.50 and 0.75 for bounding boxes), AP^m (Average Precision for mask predictions), and $AP^m_{50/75}$ (Average Precision at IoU thresholds of 0.50 and 0.75 for mask predictions). In each case we see the XCiT employing spectral conditioned (spec. cond.) out performs the original architecture. Hardware and memory overheads are discussed in Section A.2.2.

Table 2: Performance evaluation of object detection and instance segmentation on the COCO dataset. For each metric, our spectrally conditioned architecture (Spec. cond.) outperforms the original.

Model	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
original	44.9 ($\pm$0.33)	66.1 ($\pm$0.31)	48.9 ($\pm$0.35)	40.1 ($\pm$0.29)	63.1 ($\pm$0.31)	42.8 ($\pm$0.34)
Spec. cond.	45.6 ($\pm$0.31)	66.7 ($\pm$0.36)	49.6 ($\pm$0.32)	40.5 ($\pm$0.33)	63.4 ($\pm$0.28)	43.3 ($\pm$0.32)

4.3 Nyströmformer on LRA Benchmark

To evaluate the effectiveness of our method, we applied it to the Nyströmformer architecture [38], which is specifically designed to handle long-range dependencies efficiently. Experiments were conducted on the Long-Range Arena (LRA) benchmark [32], a suite of tasks targeting the ability of models to process extended input sequences. We trained two variants of the Nyströmformer: one baseline and one with spectral conditioned attention.

Validating the theory. We validate the theoretical results from Section 3 on a Nyströmformer trained on the LRA text classification task. Each experiment was repeated five times, reporting the mean and standard deviation. We computed the minimum singular values and condition numbers of W_Q, W_K, W_V, and their spectral-conditioned forms $W_Q + C_Q$, $W_K + C_K$, and $W_V + C_V$. We also evaluated the average condition number of the attention Jacobian, its conditioned variant, and the theoretical bound from Equation (10). The observed trends closely follow those for ViT-B and XCiT-M in Section 4.1, supporting our theoretical findings. Results are shown in Figure 4; standard deviations and maximum singular value plots are provided in Section A.2.1.

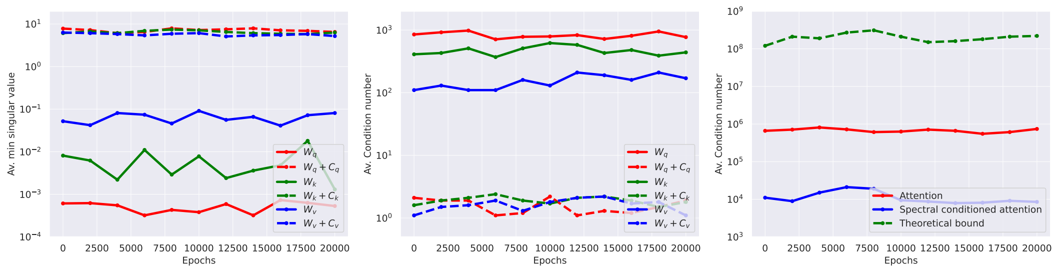

Figure 4: Analysis for Nyströmformer on text classification task. **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) and their spectrally conditioned counterparts ($W_Q + C_Q, W_K + C_K, W_V + C_V$) throughout training. **Middle:** Condition numbers of W_Q, W_K, and W_V, and their spectrally conditioned forms during training. **Right:** Average condition number of the attention Jacobian over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10).

Results. Table 3 shows the results of the experiment. As can be seen from the table the Nyströmformer that employed spectral conditioned attention outperformed the original Nyströmformer from [38] in every task in the LRA benchmark. We used the exact training setup and hyperparameters from [38]. Hardware and memory overheads are discussed in Section A.3 and an ablation on λ is given in Section A.3. A discussion comparing spectral conditioning and layer normalization is given in Section A.3.

Table 3: Comparison of a Nyströmformer using its original architecture (original) with a Nyströmformer using spectral conditioned attention (Spec. cond.) on the LRA benchmark. We report evaluation accuracy (%). As shown, our methodology consistently improves performance across all tasks.

Model	ListOps	Text	Retrieval	Image	Pathfinder
Original	37.1 ($\pm$0.21)	63.8 ($\pm$0.24)	79.8 ($\pm$0.26)	39.9 ($\pm$0.20)	72.9 ($\pm$0.25)
Spec. cond.	37.8 ($\pm$0.23)	64.8 ($\pm$0.20)	80.6 ($\pm$0.27)	40.2 ($\pm$0.22)	73.7 ($\pm$0.23)

4.4 Language Modeling with Crammed BERT

We applied our insights from Section 3 to a Crammed BERT language model, trained entirely from scratch using masked language modeling following the approach in [11]. The model consists of 110 million parameters, with 12 transformer layers and 12 attention heads per layer. For this experiment, we train two versions from scratch: the original Crammed BERT baseline model from [11] and a variant incorporating spectral conditioned attention in each layer, following the implementation from Section 3.3. Both models are trained on The Pile dataset [9], a large-scale corpus designed for language model training, following the pretraining regime carried out in [11]. After pretraining, we evaluate the performance on the GLUE benchmark [35] following the evaluation methodology outlined in [11]. Each model was trained five times on the Pile dataset [9] with five different random seeds. Table 4 shows the mean results with standard deviations shown in brackets. As can be seen from the table, spectral conditioned attention outperforms the baseline model in every downstream task in the GLUE benchmark. For the Hardware and memory overheads we refer the reader to Section A.4.

Table 4: Evaluation of a pre-trained Crammed BERT on the GLUE benchmark using the original architecture (Original) and our spectrally conditioned variant (Spec. cond.). Spectral conditioning improves performance across all tasks.

	MNLI	SST-2	STSB	RTE	QNLI	QQP	MRPC	CoLA	Avg.
Original	83.8 ($\pm$0.21)	92.3 ($\pm$0.18)	86.3 ($\pm$0.21)	55.1 ($\pm$0.20)	90.1 ($\pm$0.23)	87.3 ($\pm$0.15)	85.0 ($\pm$0.25)	48.9 ($\pm$0.26)	78.6
Spec. cond.	84.3 ($\pm$0.20)	92.7 ($\pm$0.19)	86.6 ($\pm$0.22)	55.5 ($\pm$0.24)	91.0 ($\pm$0.22)	87.5 ($\pm$0.23)	86.1 ($\pm$0.23)	51.7 ($\pm$0.24)	79.4

5 Limitations

Our spectral conditioning approach is derived by optimizing an upper bound on the condition number of the self-attention Jacobian, rather than directly minimizing the Jacobian's condition number itself. While this bound provides useful theoretical guidance and aligns with empirical improvements, it remains an indirect proxy. Developing methods to efficiently estimate and control the exact Jacobian conditioning during training would be a valuable direction for future work. Additionally, due to computational resource constraints, our experiments were limited to models with up to 100 million parameters. Whether spectral conditioning offers similar benefits for large-scale transformer models with billions of parameters remains an open question.

6 Conclusion

We presented a theoretical framework linking the conditioning of self-attention Jacobians to the spectral properties of the query, key, and value matrices in transformer architectures. Building on this insight, we introduced spectral conditioned self-attention, a simple and broadly applicable method that modifies these matrices with carefully designed correction terms to improve Jacobian conditioning. Our empirical results demonstrate that this approach consistently yields good performance across diverse transformer models and tasks, including image classification, object detection, language modeling, and long-range sequence learning.

References

[1] Naman Agarwal, Pranjal Awasthi, and Satyen Kale. A deep conditioning treatment of neural networks. In *Algorithmic Learning Theory*, pages 249–305. PMLR, 2021.

[2] Paul Albert, Frederic Z Zhang, Hemanth Saratchandran, Anton van den Hengel, and Ehsan Abbasnejad. Towards higher effective rank in parameter-efficient fine-tuning using khatri–rao product. *arXiv preprint arXiv:2508.00230*, 2025.

[3] Paul Albert, Frederic Z Zhang, Hemanth Saratchandran, Cristian Rodriguez-Opazo, Anton van den Hengel, and Ehsan Abbasnejad. Randlora: Full-rank parameter-efficient fine-tuning of large models. *arXiv preprint arXiv:2502.00987*, 2025.

[4] Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34:20014–20027, 2021.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[6] Mingyu Ding, Bin Xiao, Noel Codella, Ping Luo, Jingdong Wang, and Lu Yuan. Davit: Dual attention vision transformers. In *European conference on computer vision*, pages 74–92. Springer, 2022.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Joel N Franklin. *Matrix theory*. Courier Corporation, 2000.

[9] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Aadi Thite, Eric Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2021.

[10] Jonas Geiping. Cramming. `https://github.com/JonasGeiping/cramming`, 2023.

[11] Jonas Geiping and Tom Goldstein. Cramming: Training a language model on a single gpu in one day. In *International Conference on Machine Learning*, pages 11117–11143. PMLR, 2023.

[12] Cameron Gordon, Yiping Ji, Hemanth Saratchandran, Paul Albert, and Simon Lucey. Compressing sine-activated low-rank adapters through post-training quantization. *arXiv preprint arXiv:2505.21895*, 2025.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[14] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[15] Yiping Ji, James Martens, Jianqiao Zheng, Ziqin Zhou, Peyman Moghadam, Xinyu Zhang, Hemanth Saratchandran, and Simon Lucey. Cutting the skip: Training residual-free transformers. *arXiv preprint arXiv:2510.00345*, 2025.

[16] Yiping Ji, Hemanth Saratchandran, Cameron Gordon, Zeyu Zhang, and Simon Lucey. Sine activated low-rank matrices for parameter efficient learning. *CoRR*, 2024.

[17] Yiping Ji, Hemanth Saratchandran, Peyman Moghadam, and Simon Lucey. Always skip attention. *arXiv preprint arXiv:2505.01996*, 2025.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[19] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. *Applied and Computational Harmonic Analysis*, 59:85–116, 2022.

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[21] Lachlan MacDonald, Jack Valmadre, Hemanth Saratchandran, and Simon Lucey. On skip connections and normalisation layers in deep optimisation. *Advances in Neural Information Processing Systems*, 36:14705–14724, 2023.

[22] Jan R Magnus and Heinz Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 2019.

[23] Abhisek Maiti, Sander Oude Elberink, and George Vosselman. Transfusion: Multi-modal fusion network for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6536–6546, 2023.

[24] Matterport. Mask r-cnn implementation, 2017.

[25] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

[26] Simon JD Prince. *Understanding deep learning*. MIT press, 2023.

[27] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control. *arXiv preprint arXiv:2001.03093*, 2, 2020.

[28] Hemanth Saratchandran and Simon Lucey. Enhancing transformers through conditioned embedded tokens. *arXiv preprint arXiv:2505.12789*, 2025.

[29] Hemanth Saratchandran, Damien Teney, and Simon Lucey. Leaner transformers: More heads, less depth. *arXiv preprint arXiv:2505.20802*, 2025.

[30] Hemanth Saratchandran, Thomas X Wang, and Simon Lucey. Weight conditioning for smooth optimization of neural networks. In *European Conference on Computer Vision*, pages 310–325. Springer, 2025.

[31] Hemanth Saratchandran, Jianqiao Zheng, Yiping Ji, Wenbo Zhang, and Simon Lucey. Rethinking softmax: Self-attention with polynomial activations. *arXiv preprint arXiv:2410.18613*, 2024.

[32] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.

[33] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[35] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[36] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

[37] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Fei Tan, Glenn Fung, Vikas Singh, Xiaodong Yuan, Sungsoo Ahn Wang, Dimitris Papailiopoulos, and Katerina Fragkiadaki. Github repository, 2021.

[38] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14138–14148, 2021.

[39] Li Yuan, Qibin Hou, Zihang Jiang, Jiashi Feng, and Shuicheng Yan. Volo: Vision outlooker for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 45(5):6575–6586, 2022.

[40] Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M Susskind. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, pages 40770–40803. PMLR, 2023.

[41] Q Zhen, W Sun, H Deng, D Li, Y Wei, B Lv, J Yan, L Kong, and Y Zhong. cosformer: rethinking softmax in attention. In *International Conference on Learning Representations*, 2022.

[42] Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. A robustly optimized bert pre-training approach with post-training. In *Proceedings of the 20th chinese national conference on computational linguistics*, pages 1218–1227, 2021.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Claims stated in the introduction and abstract are shown in Section 3 with proofs given in Section A.1. Experimental validations are given in Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 5 clearly talking about limitations of the theory and experimental validation.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Each theorem in the theory Section 3 clearly states all assumptions with proofs being clearly shown in Section A.1.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The papers experiments are built on experiments already within the literature whose papers we explicitly cite. This is clearly mentioned through the paper and in the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We use publicly available code from works within the literature that we clearly cite.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We use the exact setup from works in the literature that we explicitly cite and whose GitHubs are publicly available. We also clearly state the hardware used in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We ran five trials for each experiment and showed the mean result along with standard deviations. Error plots are then shown in the Section A.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute and hardware is explicitly stated for each experiment in Section A.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We read the NeurIPS Code of Ethics and can confirm that our paper coforms to those ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: For this work there is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper has no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have clearly cited those works whose code we used. Their code is all publicly available on their GitHub.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines: The paper does not involve crowdsourcing nor research with human subjects.

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Appendix

A.1 Theoretical Framework

In this section we will give the proofs of the lemma and theorems in Section 3.2 and Section 3.3.

To begin with we will need the following standard facts on derivatives of matrices.

Lemma A.1. *Let $A \in R^{n\times m}$, $B \in \mathbb{R}^{k\times l}$ and $C \in \mathbb{R}^{m\times k}$. Then*

$$\frac{\partial ACB}{\partial C} = B^T \otimes A. \tag{15}$$

Proof. We start by using a well known vectorization identity [22]

$$\mathbf{vec}(ACB) = (B^T \otimes A)\mathbf{vec}(C) \tag{16}$$

where **vec** denotes the vectorization operator which takes a matrix and maps it to a vector by stacking its columns on top of each other, see [22]. We then differentiate Equation (16) to obtain

$$\frac{\partial \mathbf{vec}(ACB)}{\partial \mathbf{vec}(C)} = B^T \otimes A. \tag{17}$$

The result of the lemma follows. □

Lemma A.2. *Let $A \in \mathbb{R}^{n\times m}$ so that $A^T \in \mathbb{R}^{m\times n}$. Then*

$$\mathbf{vec}(A^T) = T_{mn}\mathbf{vec}(A) \tag{18}$$

$$\frac{\partial \mathbf{vec}(A^T)}{\partial \mathbf{vec}(A)} = T_{mn} \tag{19}$$

where T_{mn} is a commutation matrix.

Proof. The first equation follows from the definition of the transpose of a matrix [22]. The second equation then follows from the first. □

We now give the proof of Theorem 3.2.

Proof of Theorem 3.2. Let $[z_1, \dots, z_n]$ denote a row vector in $\mathbb{R}^n$. Then by definition

$$\mathbf{softmax}([z_1, \dots, z_n]) = \left[\frac{e^{z_1}}{\sum_{i=1}^n e^{z_i}}, \dots, \frac{e^{z_n}}{\sum_{i=1}^n e^{z_i}}\right]. \tag{20}$$

From the above equation we can compute the partial derivative and find

$$\frac{\partial\, \mathrm{softmax}(z)_i}{\partial z_j} = \mathrm{softmax}(z)_i \left(\delta_{ij} - \mathrm{softmax}(z)_j\right). \tag{21}$$

The term $\frac{\partial\, \mathrm{softmax}(z)_i}{\partial z_j}$ is precisely the ij component of the matrix $\frac{\partial \mathbf{softmax}(z)}{\partial z}$. Putting each of these ij terms into an $n \times n$ matrix we find

$$\frac{\partial \mathbf{softmax}(z)}{\partial z} = Diag(z) - z \cdot z^T \tag{22}$$

which proves the lemma. □

We can use the above lemmas to give the proof of Theorem 3.3.

Proof of Theorem 3.3. We start by establishing the derivative formula for the term $\frac{\partial \mathbf{A}(X)}{\partial W_Q}$. We will use the notation used in Section 3.1. Note that by definition $\frac{\partial \mathbf{A}(X)}{\partial W_Q} \in \mathbb{R}^{dN\times dD}$. Using Equation (3)

$$\mathbf{A}(X) = I_N \mathbf{softmax}(XW_Q W_K^T X^T)XW_V \tag{23}$$

where I_N is the $N \times N$ identity matrix. This is done so that we can apply Theorem A.1. We then compute

$$\begin{aligned}
\frac{\partial \mathbf{A}(X)}{\partial W_Q} &= \frac{\partial (I_N \mathbf{softmax}(XW_QW_K^TX^T)XW_V)}{\partial W_Q} && (24)\\
&= (W_V^TX^T \otimes I_N)\frac{\partial \mathbf{softmax}(XW_QW_K^TX^T)}{\partial W_Q} \text{ using } Theorem\ A.1 && (25)\\
&= (W_V^TX^T \otimes I_N)\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))\frac{\partial (XW_QW_K^TX^T)}{\partial W_Q} \text{ using } Theorem\ 3.2 && (26)\\
&= (W_V^TX^T \otimes I_N)\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))(XW_K \otimes X) && (27)
\end{aligned}$$

which proves the firs equality in Theorem 3.3.

To compute $\frac{\partial \mathbf{A}(X)}{\partial W_K} \in \mathbb{R}^{dN \times dD}$ we proceed in a similar way.

$$\begin{aligned}
\frac{\partial \mathbf{A}(X)}{\partial W_K} &= \frac{\partial (I_N \mathbf{softmax}(XW_QW_K^TX^T)XW_V)}{\partial W_K} && (28)\\
&= (W_V^TX^T \otimes I_N)\frac{\partial \mathbf{softmax}(XW_QW_K^TX^T)}{\partial W_K} \text{ using } Theorem\ A.1 && (29)\\
&= (W_V^TX^T \otimes I_N)\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))\frac{\partial (XW_QW_K^TX^T)}{\partial W_K} \text{ using } Theorem\ 3.2 && (30)\\
&= (W_V^TX^T \otimes I_N)\Lambda(\mathbf{softmax}(XW_QW_K^TX^T))(X \otimes XW_Q)T_{Dd} && (31)
\end{aligned}$$

where the last equality follows from Theorems A.1 and A.2 This establishes the second equality in Theorem 3.3.

To prove the identity for $\frac{\partial \mathbf{A}(X)}{\partial W_V} \in \mathbb{R}^{dN \times dD}$ we write

$$\mathbf{A}(X) = \mathbf{softmax}(XW_QW_K^TX^T)XW_VI_d \tag{32}$$

where I_d is the $d \times d$ identity matrix. Then we simply apply Theorem A.1 to obtain

$$\begin{aligned}
\frac{\partial \mathbf{A}(X)}{\partial W_V} &= \frac{\partial (\mathbf{softmax}(XW_QW_K^TX^T)XW_VI_d)}{\partial W_V} && (33)\\
&= I_d \otimes \mathbf{softmax}(XW_QW_K^TX^T)X && (34)
\end{aligned}$$

which proves the final equality in Theorem 3.3. □

Proof of Theorem 3.4. The proof of Theorem 3.4 follows from using Theorem 3.3 and the definition of the Jacobian of $\mathbf{A}(X)$ with respect to W_Q, W_K and W_V given by

$$J(\mathbf{A}(X)) = \left[\frac{\partial(\mathbf{A}(X))}{\partial W_Q}, \frac{\partial(\mathbf{A}(X))}{\partial W_K}, \frac{\partial(\mathbf{A}(X))}{\partial W_V}\right]^T. \tag{35}$$

We recall that the condition number is defined as $\kappa(J(\mathbf{A}(X))) = \frac{\sigma_{\max}(J(\mathbf{A}(X)))}{\sigma_{\min}(J(\mathbf{A}(X)))}$ where $\sigma_{\max}(J(\mathbf{A}(X)))$ is the maximum singular value of $J(\mathbf{A}(X))$ and $\sigma_{\min}(J(\mathbf{A}(X)))$ the minimum singular value which we know is non-zero because of the assumption that $J(\mathbf{A}(X))$ has full rank. Note that, using the notation in Section 3.1, we have that $J(\mathbf{A}(X)) \in \mathbb{R}^{3dN \times dD}$ as $\frac{\partial \mathbf{A}(X)}{\partial W_Q}$, $\frac{\partial \mathbf{A}(X)}{\partial W_K}$, $\frac{\partial \mathbf{A}(X)}{\partial W_V} \in \mathbb{R}^{dN \times dD}$. For each of notation we will write $\mathbf{A}_Q := \frac{\partial \mathbf{A}(X)}{\partial W_Q}$, $\mathbf{A}_K := \frac{\partial \mathbf{A}(X)}{\partial W_K}$, $\mathbf{A}_V := \frac{\partial \mathbf{A}(X)}{\partial W_V}$.

We will start by computing a bound for the maximum singular value. We have for any vector $z \in \mathbb{R}^{dD}$ we have

$$||J(\mathbf{A}(X))z||_2^2 = \left\| \left[\frac{\partial(\mathbf{A}(X))}{\partial W_Q}(z), \frac{\partial(\mathbf{A}(X))}{\partial W_K}(z), \frac{\partial(\mathbf{A}(X))}{\partial W_V}(z) \right]^T \right\|_2^2 \tag{36}$$

$$= \left\| \left[\mathbf{A}_Q(z), \mathbf{A}_K(z), \mathbf{A}_V(z) \right] \right\|_2^2 \tag{37}$$

$$= \|\mathbf{A}_Q(z)\|_2^2 + \|\mathbf{A}_K(z)\|_2^2 + \|\mathbf{A}_V(z)\|_2^2 \tag{38}$$

$$\leq \left(\sigma_{\max}(\mathbf{A}_Q)^2 + \sigma_{\max}(\mathbf{A}_K)^2 + \sigma_{\max}(\mathbf{A}_V)^2 \right) ||z||_2^2. \tag{39}$$

This implies that

$$\sigma_{\max}(J(\mathbf{A}(X))) := \max_{z \neq 0} \frac{||J(\mathbf{A}(X))z||_2^2}{||z||_2^2} \tag{40}$$

$$\leq \sqrt{\sigma_{\max}(\mathbf{A}_Q)^2 + \sigma_{\max}(\mathbf{A}_K)^2 + \sigma_{\max}(\mathbf{A}_V)^2} \tag{41}$$

$$\leq \sigma_{\max}(\mathbf{A}_Q) + \sigma_{\max}(\mathbf{A}_K) + \sigma_{\max}(\mathbf{A}_V). \tag{42}$$

The next step is to compute a lower bound for the minimum singular value $\sigma_{\min}(J(\mathbf{A}(X)))$. The approach is similar to the above, using the fact that $\sigma_{\min}(J(\mathbf{A}(X))) = \min_{||z||_2=1} J(\mathbf{A}(X)(z)$. We can then use the inequality

$$\|J(\mathbf{A}(X)(z)\|_2^2 = \left\| \left[\frac{\partial(\mathbf{A}(X))}{\partial W_Q}(z), \frac{\partial(\mathbf{A}(X))}{\partial W_K}(z), \frac{\partial(\mathbf{A}(X))}{\partial W_V}(z) \right]^T \right\|_2^2 \tag{43}$$

$$\geq \max \left\{ \left\| \frac{\partial(\mathbf{A}(X))}{\partial W_Q}(z) \right\|, \left\| \frac{\partial(\mathbf{A}(X))}{\partial W_K}(z) \right\|, \left\| \frac{\partial(\mathbf{A}(X))}{\partial W_V}(z) \right\| \right\} \tag{44}$$

Then minimizing the above over the constraint $z \in \mathbb{R}^{dD}$ such that $||z|| = 1$ we obtain

$$\sigma_{\min}(J(\mathbf{A}(X))) \geq \max \left\{ \sigma_{\min}\left(\frac{\partial(\mathbf{A}(X))}{\partial W_Q} \right), \sigma_{\min}\left(\frac{\partial(\mathbf{A}(X))}{\partial W_K} \right), \sigma_{\min}\left(\frac{\partial(\mathbf{A}(X))}{\partial W_V} \right) \right\}. \tag{45}$$

Combing the bounds on $\sigma_{\max}(J(\mathbf{A}(X)))$ and $\sigma_{\min}(J(\mathbf{A}(X)))$ we obtain

$$\kappa(J(\mathbf{A}(X))) \leq \kappa\left(\frac{\partial(\mathbf{A}(X))}{\partial W_Q} \right) + \kappa\left(\frac{\partial(\mathbf{A}(X))}{\partial W_K} \right) + \kappa\left(\frac{\partial(\mathbf{A}(X))}{\partial W_V} \right). \tag{46}$$

The final step is to get a bound on the condition numbers for each term on the right hand side in the above inequality. This is done using two facts: Firstly, given two matrices C and D such that the product CD is full rank then $\kappa(CD) \leq \kappa(C)\kappa(D)$ and the second that $\kappa(C \otimes D) = \kappa(C)\kappa(D)$. Using these two facts we can then use Theorem 3.3 to obtain the bound of Theorem 3.4 and the proof is finished. □

We will now give the proof of Theorem 3.5.

Proof of Theorem 3.5. We will give the proof for W_Q as the proof for W_K and W_V are exactly analogous.

Take the SVD of W_Q to obtain $W_Q = USV^T$ where $U \in \mathbb{R}^{D \times D}$ is the left singular vectors, $S \in \mathbb{R}^{D \times d}$ is the matrix of singular values having the singular values of W_Q on its main diagonal, $V \in \mathbb{R}^{d \times d}$ is the right singular vectors. We then define a new matrix C_Q by the formula

$$C_Q := U\overline{S}V^T \tag{47}$$

where $\overline{S} \in \mathbb{R}^{D \times d}$ has $\sigma_{\max}(W_Q)$ on its main diagonal and zeros elsewhere. We then have that

$$W_Q + C_Q = U(S + \overline{S})V^T. \tag{48}$$

We can then compute $\sigma_{\max}(W_Q + C_Q) = 2\sigma_{\max}(W_Q)$ by definition of C_Q and $\sigma_{\min}(W_Q + C_Q) = \sigma_{\min}(W_Q) + \sigma_{\max}(W_Q)$. This implies

$$\kappa(W_Q + C_Q) = \frac{\sigma_{\max}(W_Q + C_Q)}{\sigma_{\min}(W_Q + C_Q)} \tag{49}$$

$$= \frac{2\sigma_{\max}(W_Q)}{\sigma_{\min}(W_Q) + \sigma_{\max}(W_Q)} \tag{50}$$

$$< 2 \tag{51}$$

which finishes the proof of the theorem. □

A key issue for implementing the approach in Theorem 3.5 into the attention layer is that it requires the computation of the SVD of each of the matrices W_Q, W_K and W_V. Computing the SVD each time for a large matrix is extremely memory intensive. This is exactly the premise of Theorem 3.8 whose proof we now give.

Proof of Theorem 3.8. The proof starts by making use of the Weyl inequalities for singular values [8] to obtain

$$\sigma_{\max}(A + \lambda \cdot I_k) \leq \sigma_{\max}(A) + \sigma_{\max}(\lambda \cdot I_k). \tag{52}$$

$$\sigma_{\min}(A + \lambda \cdot I_k) \geq \sigma_{\min}(\lambda \cdot I_k) - \sigma_{\max}(A). \tag{53}$$

We then compute

$$\kappa(A + \lambda \cdot I_k) := \frac{\sigma_{\max}(A + \lambda \cdot I_k)}{\sigma_{\min}(A + \lambda \cdot I_k)} \tag{54}$$

$$\leq \frac{\sigma_{\max}(A) + \sigma_{\max}(\lambda \cdot I_k)}{\sigma_{\min}(\lambda \cdot I_k) - \sigma_{\min}(A)} \tag{55}$$

$$= \frac{\sigma_{\max}(A) + \lambda}{\lambda - \sigma_{\min}(A)} \tag{56}$$

$$\leq \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)} \text{ by our assumption} \tag{57}$$

$$= \kappa(A) \tag{58}$$

which completes the proof. □

Other forms of Attention. The theory developed in Section 3 was presented for the self-attention mechanism defined in Section 3.1, to allow for concrete analytical expressions. However, as evident from the derivations, the proposed corrections to the weight matrices W_Q, W_K, and W_V apply to any attention layer. Since the results depend only on the derivatives of the attention function with respect to these matrices, analogous theorems can be established for more general mechanisms such as cross-attention. Moreover, in Section 4, we empirically demonstrate that our theoretical results extend to a broad class of attention architectures.

A.2 Experimental Analysis

Details of implementation of spectral conditioned attention. For all of the experiments in Section 4 we implemented spectral conditioned attention following the strategy outlined in Section 3.3 via Theorem 3.8. We describe this implementation in detail. For each attention layer in a transformer fix the query, key and value weights as W_Q, W_K, W_V. We then form three matrices denoted by C_Q, C_K and C_V with each being defined by the equation

$$C_Q = \lambda I_Q \tag{59}$$

$$C_K = \lambda I_K \tag{60}$$

$$C_V = \lambda I_V \tag{61}$$

with $\lambda = 10$ and where C_Q, C_K and C_V have the same shape as W_Q, W_K and W_V and I_Q, I_K and I_V has the same shape as C_Q, C_K and C_V with 1's on their main diagonal and zeros elsewhere.

The matrices C_Q, C_K and C_V, which we call the correction matrices, are fixed throughout training the transformer and never change. They are not updated and hence do not add any memory to the backpropagation algorithm and the only memory comes from storing them. During the training of the transformer, we add each of these correction matrices to the weights W_Q, W_K and W_V during each forward pass forming. Thus after initializing the weights W_Q^0, W_K^0 and W_V^0 we add the correction terms C_Q, C_K and C_V as follows

$$1.W_Q^0 \to W_Q^0 + C_Q \tag{62}$$

$$2.W_K^0 \to W_K^0 + C_K \tag{63}$$

$$3.W_V^0 \to W_V^0 + C_V \tag{64}$$

perform a forward pass using $W_Q^0 + C_Q$, $W_K^0 + C_K$ and $W_V^0 + C_V$, then perform a backward pass and during the backward pass only the weights W_Q^0, W_K^0 and W_V^0 are updated forming W_Q^1, W_K^1 and W_V^1. This process then repeats, we add the fixed correction terms to W_Q^1, W_K^1 and W_V^1 forming

$$1.W_Q^1 \to W_Q^1 + C_Q \tag{65}$$

$$2.W_K^1 \to W_K^1 + C_K \tag{66}$$

$$3.W_V^1 \to W_V^1 + C_V \tag{67}$$

then a backward pass updates W_Q^1, W_K^1 and W_V^1 and this process continues. Note that Theorem 3.8 allows any $\lambda \geq 2$ when forming Through ablations we found the best value is $\lambda = 10$.

A.2.1 Vision transformers on Imagenet-1k

Validating the theory. In Section 4.1 we validated the theory given in Section 3 on a ViT-B and XCiT-M architecture. For each experiment we ran five trials with five different random seeds and plotted the mean. The plots in Section 4.1 were shown in a log scale where the standard deviations were not visible. We have plotted each plot in a new scale that clearly shows the standard deviations. The plots for ViT-B can be seen in Figures 5 to 8 and the plots for XCiT-M can be seen from Figures 9 to 12. Furthermore, Figure 6 validates the assumption in Theorem 3.8 on the maximum singular value for ViT-B and Figure 10 for the maximum singular for XCiT justifying the use of Theorem 3.8.

Ablation on λ from Theorem 3.8. In Section 4 we used the implementation for the correction terms given in Section 3. This required choosing a factor $\lambda \geq 2$. For the experiments in Section 4 we chose λ through a grid search treating it as a hyperparameter. An ablation for λ on the ViT-B architecture trained on ImageNet-1k is shown in Table 5. What we found was that $\lambda \geq 10$ gives the best result.

Table 5: Ablation on λ for ViT-B

λ	2	4	6	8	10	12	14	16
ViT-B Acc.	80.8	81.1	81.4	81.6	81.7	81.7	81.6	81.5

Relation to Normalization. Our conditioning methodology can be seen as being related to normalization. By adding correction terms to the weights W_Q, W_K and W_V to improve their spectrum we are in effect producing a normalization for their spectrum. We therefore decided to test our spectral conditioning when we remove layer normalization in the vision transformers to understand whether spectral conditioning can replace layer normalization. Table 6 shows the results on the ViTs on ImageNet-1k. We ran each experiments 5 times and plot the mean and standard deviation in brackets. We found that if we remove layer normalization and keep spectral conditioning (with $\lambda = 10$) the performance drops. We noticed that this was due to the fact that the weight values in the corrected terms $W_Q + C_Q$, $W_K + C_K$ and $W_V + C_V$ were much larger than without layer normalization which can lead to issues with backpropagation and suggesting that layer normalization was still crucial to maintain weights from getting too large.

Table 6: Performance comparison of spectral conditioning with and without layer normalization.

	ViT-B	DeiT-B	Swin-B	XCiT-M	DaViT-B
Original (with only layer norm.)	80.7 ($\pm$0.41)	81.6 ($\pm$0.30)	83.4 ($\pm$0.28)	82.6 ($\pm$0.39)	84.3 ($\pm$0.26)
Spec. cond. + layer norm.	81.7 ($\pm$0.38)	82.6 ($\pm$0.32)	84.1 ($\pm$0.25)	83.5 ($\pm$0.35)	84.9 ($\pm$0.21)
Spec. cond. - layer norm.	79.8 ($\pm$0.31)	80.4 ($\pm$0.33)	80.5 ($\pm$0.23)	80.0 ($\pm$0.35)	80.3 ($\pm$0.21)

Comparing Theorem 3.5 and Theorem 3.8. In Section 3, we saw that Theorem 3.5 gave a methodology that could reduce the condition number of the query weights, key weights and value weights to less than 2. However, the reason we did not use this theorem as the basis of our experiments is that it requires an SVD computation which would significantly increase training times when compared to the implementation friendly theorem Theorem 3.8. We carried out a comparison on the vision transformer for both methods. As can be seen from Table 7 the vision transformer employing the methodology from Theorem 3.5 take much longer to train.

Table 7: Comparison of accuracy and training time across transformer variants with spectral conditioning of attention following Theorem 3.5 and Theorem 3.8.

Model	**Acc.**	**Training time (hrs:mins)**
ViT-B (original)	80.7	29:29
ViT-B spec. cond. (Thm. 3.5)	82.0	41:38
ViT-B spec. cond. (Thm. 3.8)	81.7	29:33
DeiT-B (original)	81.6	26:16
DeiT-B spec. cond. (Thm. 3.5)	82.9	37:54
DeiT-B spec. cond. (Thm. 3.8)	82.6	26:24
Swin-B (original)	83.4	53:12
Swin-B spec. cond. (Thm. 3.5)	84.3	68:32
Swin-B spec. cond. (Thm. 3.8)	84.1	53:26
XCiT-M (original)	82.6	91:03
XCiT-M spec. cond. (Thm. 3.5)	83.9	109:12
XCiT-M spec. cond. (Thm. 3.8)	83.5	91:18
DaViT-M (original)	84.3	75:09
DaViT-M spec. cond. (Thm. 3.5)	85.2	93:12
DaViT-M spec. cond. (Thm. 3.8)	84.9	75:28

Hardware and implementation. The image classification experiments in Section 4.1 of the paper were done on Nvidia A100 GPUs. The implementation of the ViTs were all done using the Timm code base [36]. The architectures were all trained from scratch on the ImageNet-1k dataset using the AdamW optimizer following the hyperparameters used in the original papers [7, 20, 4, 33, 6].

Memory Cost. We analyze the memory overhead introduced by using spectrally conditioned weight matrices $W_Q + C_Q$, $W_K + C_K$, and $W_V + C_V$ in the forward pass for the ViT-B architecture. A similar analysis gives a similar profile for all the other ViT architectures we consider in Section 4.1. The correction matrices C_Q, C_K, and C_V are fixed diagonal matrices with constant entries (value 10) on the main diagonal, are non-trainable, and have gradients disabled (`requires_grad = False`).

Computing $X(W_Q + C_Q)$ expands as:

$$X(W_Q + C_Q) = XW_Q + XC_Q, \tag{68}$$

where XW_Q is a standard matrix multiplication and XC_Q corresponds to scaling each column of X by 10. This operation is efficiently implemented as a column-wise scaling, requiring no explicit storage of C_Q beyond a scalar or a small constant vector.

- **Parameters:** No new trainable parameters are introduced; C_Q, C_K, and C_V are fixed and non-trainable.
- **Gradient Storage:** Since the correction matrices do not participate in backpropagation, no additional gradient memory is used.

- **Activation Memory:** The computation of XC_Q involves element-wise scaling and summation, which reuses existing output tensors without requiring new memory allocations.

In practice, the additional memory overhead from spectral conditioning is negligible, as it involves no new trainable parameters, no gradient storage, and no extra activation tensors.

Table 8: Comparison of memory overhead for spectral conditioning relative to standard attention projections.

Memory Aspect	**Original Attention**	**With Spectral Conditioning**
Trainable Parameters	$3 \times Dd$	$3 \times Dd$
Gradient Storage	$3 \times Dd$	$3 \times Dd$
Correction Matrices (C_Q, C_K, C_V)	–	negligible (constant scalar)
Activation Memory	$3 \times Nd$	$3 \times Nd$ (no extra tensors)

FLOPS Cost. We analyze the additional FLOPS overhead introduced by using spectrally conditioned weight matrices $W_Q + C_Q$, $W_K + C_K$, and $W_V + C_V$ in the forward pass of the attention mechanism in the forward pass for the ViT-B architecture. A similar analysis gives a similar profile for all the other ViT architectures we consider in Section 4.1. Consider the self-attention formulation:

$$\textbf{softmax}(X(W_Q + C_Q)(W_K + C_K)^\top X^\top) \cdot X(W_V + C_V), \tag{69}$$

where $X \in \mathbb{R}^{N \times D}$ is the input, and $W_Q, W_K, W_V \in \mathbb{R}^{D \times d}$ are the learnable projection matrices. The correction matrices C_Q, C_K, and C_V are fixed diagonal matrices with constant entries 10 along their main diagonal.

Computing the term $X(W_Q + C_Q)$ expands as:

$$X(W_Q + C_Q) = XW_Q + XC_Q. \tag{70}$$

The first term, XW_Q, corresponds to a standard matrix multiplication with computational complexity $\mathcal{O}(NDd)$, requiring $2NDd$ FLOPS. The second term, XC_Q, corresponds to scaling each column of X by 10, which requires only Nd FLOPS (one multiplication per entry).

Thus, the total FLOPS for computing $X(W_Q + C_Q)$ is:

$$2NDd + Nd. \tag{71}$$

An identical cost arises for $X(W_K + C_K)$ and $X(W_V + C_V)$. Therefore, the additional FLOPS introduced by the correction terms across all three projections is:

$$3Nd. \tag{72}$$

Compared to the original cost of $6NDd$ FLOPS for the query, key, and value projections without spectral conditioning, the overhead introduced by spectral conditioning is negligible:

$$\frac{3Nd}{6NDd} = \frac{1}{2D}, \tag{73}$$

which is insignificant for typical values of D used in practice (e.g., $D = 512, 768, 1024$ are common dimension for ViTs).

Table 9: FLOPS comparison for the forward pass of the attention projection layers with and without spectral conditioning. N denotes the sequence length, D the embedding dimension, and d the projection dimension per head.

Operation	**FLOPS (Original)**	**FLOPS (With Spectral Conditioning)**
Query projection (XW_Q)	$2NDd$	$2NDd + Nd$
Key projection (XW_K)	$2NDd$	$2NDd + Nd$
Value projection (XW_V)	$2NDd$	$2NDd + Nd$
Total Projection FLOPS	$6NDd$	$6NDd + 3Nd$

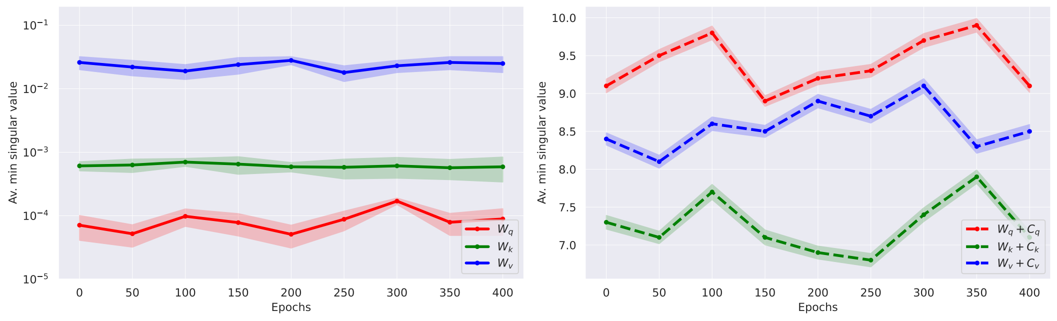

Figure 5: **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a ViT-B during training. We plot the mean over five trials and the standard deviation. **Right:** Average minimum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a ViT-B during training. We plot the mean over five trials and the standard deviation.

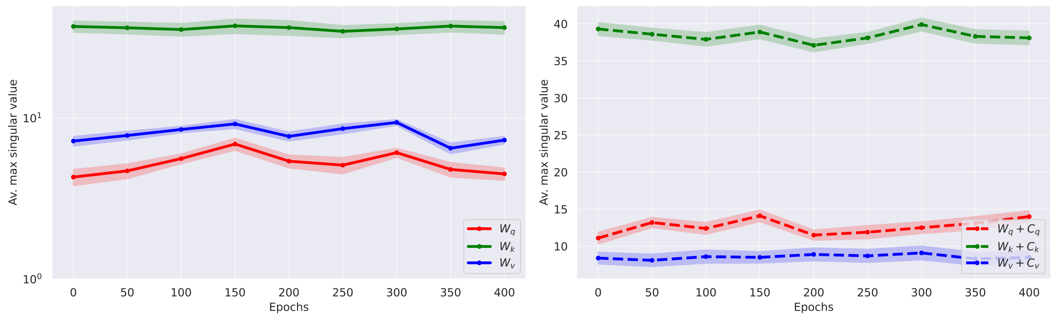

Figure 6: **Left:** Average maximum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a ViT-B during training. We plot the mean over five trials and the standard deviation. **Right:** Average maximum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a ViT-B during training. We plot the mean over five trials and the standard deviation.

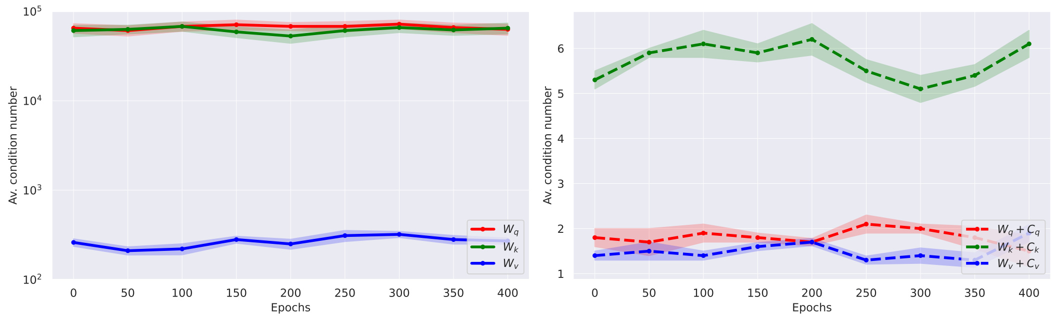

Figure 7: **Left:** Average condition number of the query, key, and value projection matrices (W_Q, W_K, W_V) for a ViT-B during training. We plot the mean over five trials and the standard deviation. **Right:** Average condition number of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a ViT-B during training. We plot the mean over five trials and the standard deviation.

A.2.2 Object detection and instance segmentation

Hardware and Implementation: The experiments for Section 4.2 of the paper on object detection and instance segmentation were carried out on Nvidia A100 GPUs. The implementation followed [13]. We used the code base given by the GitHub [24] following their exact training regime.

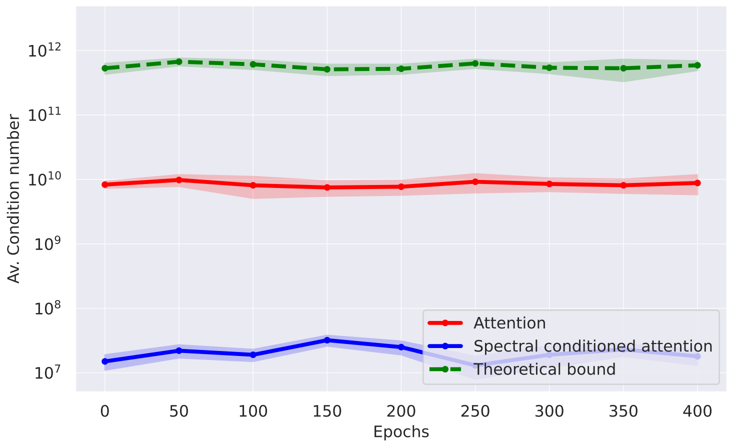

Figure 8: Average condition number of the self-attention Jacobian of a ViT-B over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10). We ran five trials with five different random seeds with plots showing the mean and standard deviations.

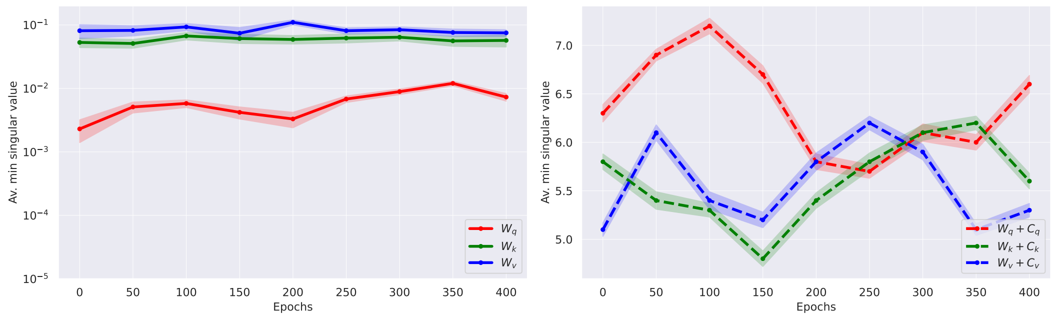

Figure 9: **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a XCiT-M during training. We plot the mean over five trials and the standard deviation. **Right:** Average minimum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a XCiT-M during training. We plot the mean over five trials and the standard deviation.

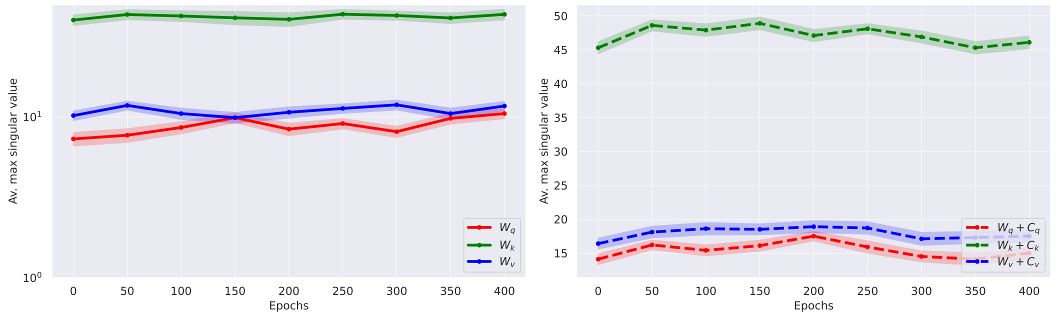

Figure 10: **Left:** Average maximum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a XCiT-M during training. We plot the mean over five trials and the standard deviation. **Right:** Average maximum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a XCiT-M during training. We plot the mean over five trials and the standard deviation.

Memory and Overheads: For this experiment we used a pre-trained XCiT-S and since, as explained in Section A.2.1, the correction terms are not trained and fixed throughout training there was no real memory overhead.

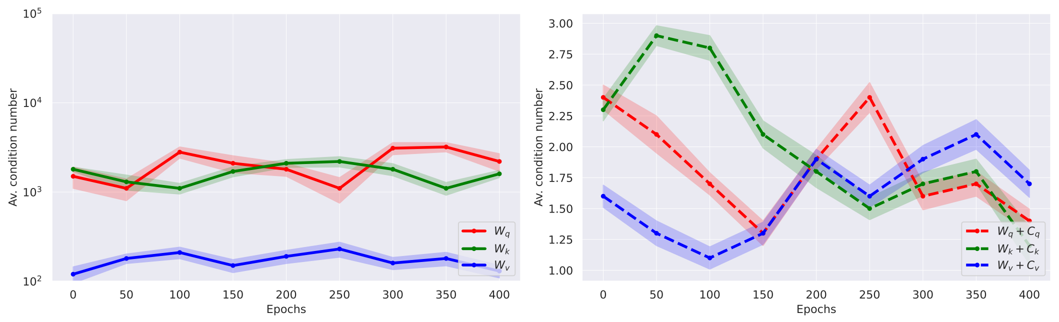

Figure 11: **Left:** Average condition number of the query, key, and value projection matrices (W_Q, W_K, W_V) for a XCiT-M during training. We plot the mean over five trials and the standard deviation. **Right:** Average condition number of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a XCiT-M during training. We plot the mean over five trials and the standard deviation.

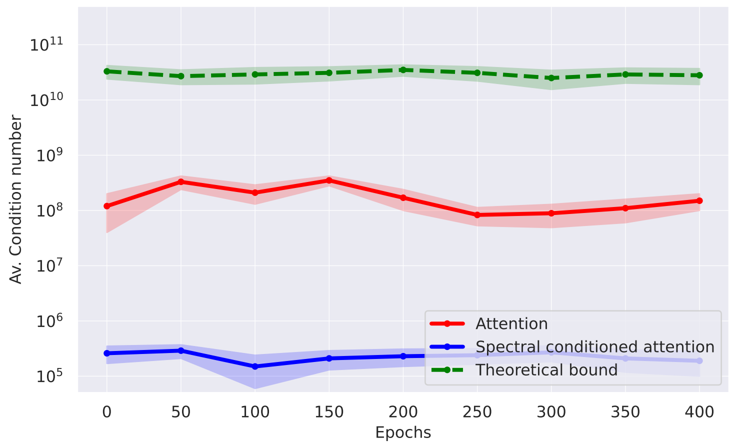

Figure 12: Average condition number of the self-attention Jacobian of a XCiT-M over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10). We ran five trials with five different random seeds with plots showing the mean and standard deviations.

Further results: We also ran the object detection and instance segmentation results using a Swin-base vision transformer as the backbone. The result are shown in Table 10.

Table 10: Comparison of Swin-base and spectrally conditioned (spec. cond.) Swin-base on COCO benchmarks.

Model	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
Swin-base	45.9	67.1	50.1	41.1	63.7	43.8
Spec. cond. Swin-base	**46.8**	**68.1**	**50.7**	**41.8**	**64.2**	**44.6**

A.3 Nyströmformer on LRA benchmark

Validating the theory. In Section 4.3 we validated the theory given in Section 3 on a Nyströmformer on the text classification task in the LRA benchmark. For each experiment we ran five trials with five different random seeds and plotted the mean. The plots in Section 4.3 were shown in a log scale where the standard deviations were not visible. We have plotted each plot in a new scale that clearly shows the standard deviations. The plots are shown from Figures 13 to 16. We also ran a similar empirical analysis for the listops task in the LRA benchmark. The results of which can be seen in

Figures 17 to 21. Furthermore, Figure 14 and Figure 19 validates the assumption in Theorem 3.8 on the maximum singular value justifying the use of Theorem 3.8.

Ablation on λ from Theorem 3.8. For the experiments in Section 4.3 we chose λ through a grid search treating it as a hyperparameter. An ablation for λ on the Nyströmformer trained on the text classification task on the LRA benchmark is shown in Table 11. Once again we found that $\lambda \geq 10$ gives the best result.

Table 11: Ablation on λ for Nyströmformer on text classification task.

λ	2	4	6	8	10	12	14	16
Acc.	63.7	64.1	64.3	64.6	64.8	64.8	64.7	64.7

Relation to Normalization. We compared spectral conditioning on a Nyströmformer with and without layer normalization. As can be seen in Table 12 we see that layer normalization is important for spectral conditioning. Once again when we removed layer normalization and just applied spectral conditioning we found that the weights were large resulting in poorer performance.

Table 12: Comparing spectral conditioning with and without layer normalization for the Nyströmformer on text classification.

	Acc.(%)
Original (with only layer norm.)	63.8 ($\pm$0.24)
Spec. cond. + layer norm.	64.8 ($\pm$0.20)
Spec. cond. - layer norm	62.3 ($\pm$ 0.23)

Hardware. All the experiments for the Nyströmformer on LRA benchmark results in Section 4.3 were carried out on Nvidia A100 GPUs following the implementation and hyperparameter settings given in [37].

Memory and Overheads: The analysis carried out on FLOPS and memory in Section A.2.1 apply in this case as once again the correction matrices C_Q, C_K and C_V are fixed throughout training and only added for forward passes.

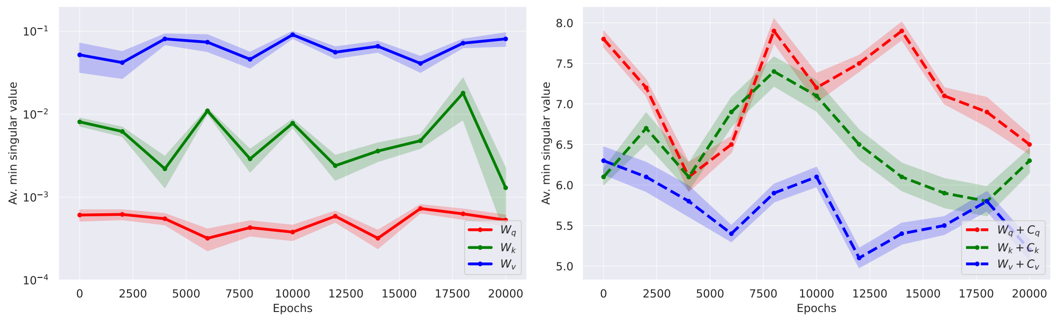

Figure 13: **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a Nyströmformer on the text classification task during training. We plot the mean over five trials and the standard deviation. **Right:** Average minimum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a Nyströmformer on the text classification during training. We plot the mean over five trials and the standard deviation.

A.4 Language Modeling with Crammed BERT

Hardware. The language modeling experiment in Section 4.4 were all carried out on a Nvidia A6000 GPU. The Crammed-Bert was implemented following the original paper [11] and the original GitHub [10]. The training regime follows [10].

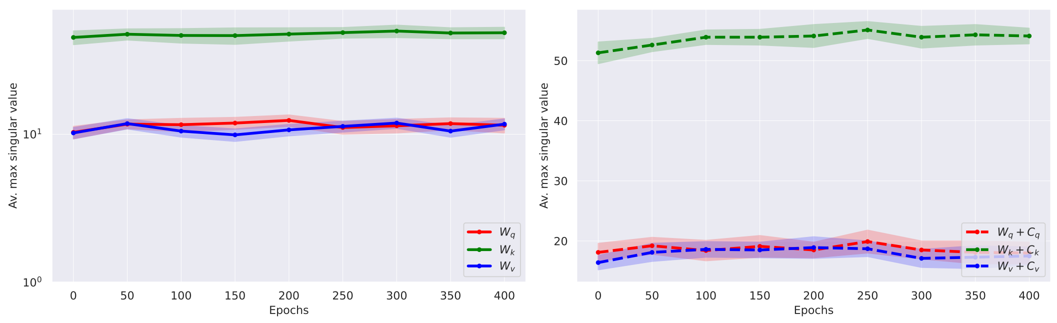

Figure 14: **Left:** Average maximum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a Nyströmformer on the text classification task during training. We plot the mean over five trials and the standard deviation. **Right:** Average maximum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a Nyströmformer on the text classification during training. We plot the mean over five trials and the standard deviation.

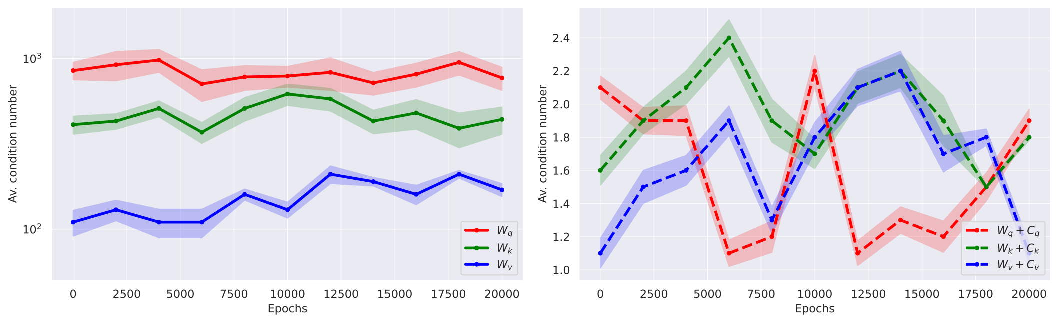

Figure 15: **Left:** Average condition number of the query, key, and value projection matrices (W_Q, W_K, W_V) for a Nyströmformer on the text classification task during training. We plot the mean over five trials and the standard deviation. **Right:** Average condition number of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a Nyströmformer on the text classification task during training. We plot the mean over five trials and the standard deviation.

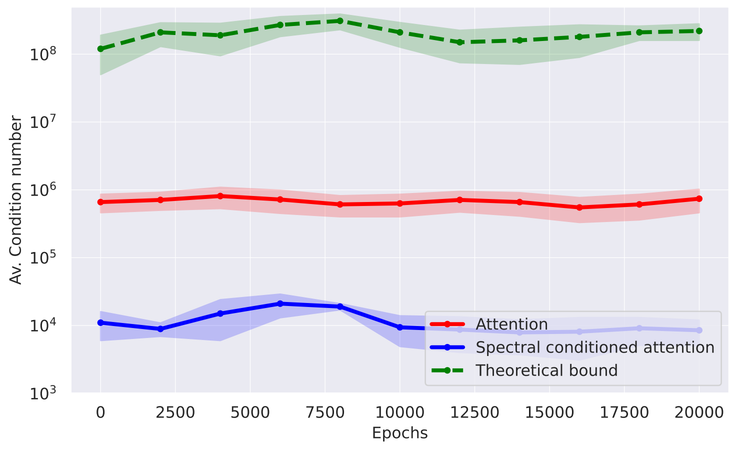

Figure 16: Average condition number of the self-attention Jacobian of a Nyströmformer on the text classification task over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10). We ran five trials with five different random seeds with plots showing the mean and standard deviations.

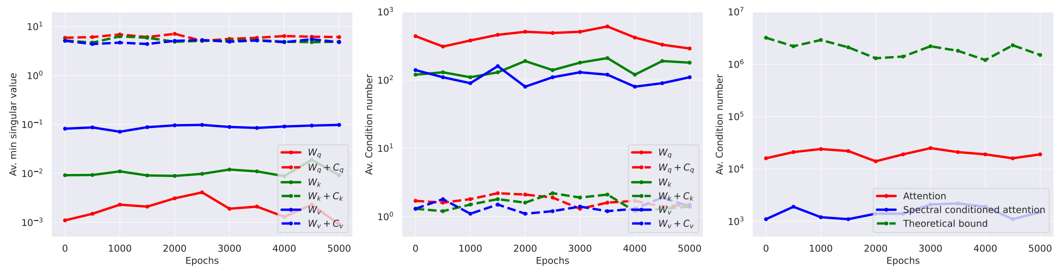

Figure 17: Analysis for Nyströmformer on listops task. **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) and their spectrally conditioned counterparts ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) throughout training. **Middle:** Condition numbers of W_Q, W_K, and W_V, and their spectrally conditioned forms during training. **Right:** Average condition number of the attention Jacobian over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10).

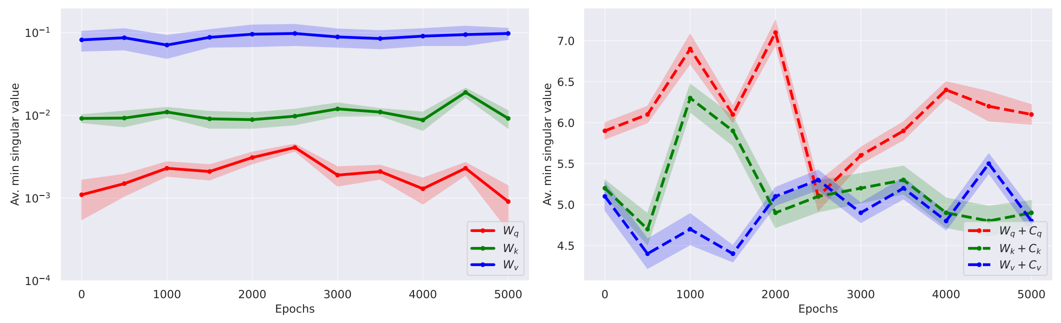

Figure 18: **Left:** Average minimum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a Nyströmformer on the listops task during training. We plot the mean over five trials and the standard deviation. **Right:** Average minimum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a Nyströmformer on the listops task during training. We plot the mean over five trials and the standard deviation.

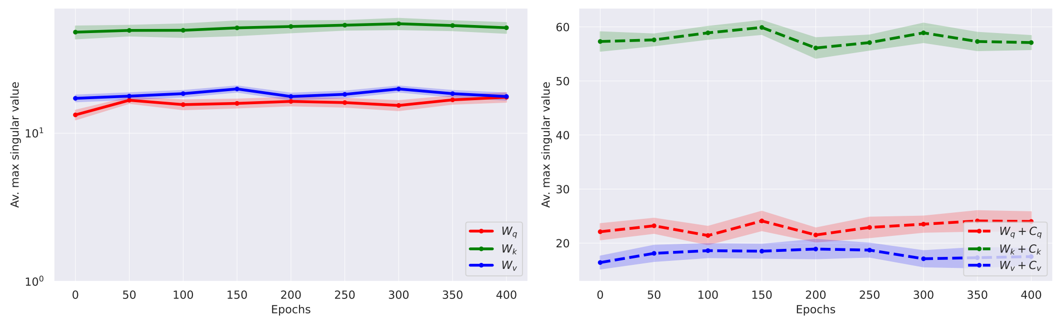

Figure 19: **Left:** Average maximum singular value of the query, key, and value projection matrices (W_Q, W_K, W_V) for a Nyströmformer on the listops task during training. We plot the mean over five trials and the standard deviation. **Right:** Average maximum singular value of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a Nyströmformer on the listops task during training. We plot the mean over five trials and the standard deviation.

Memory and Overheads: The analysis carried out on FLOPS and memory in Section A.2.1 apply in this case as once again the correction matrices C_Q, C_K and C_V are fixed throughout training and only added for forward passes.

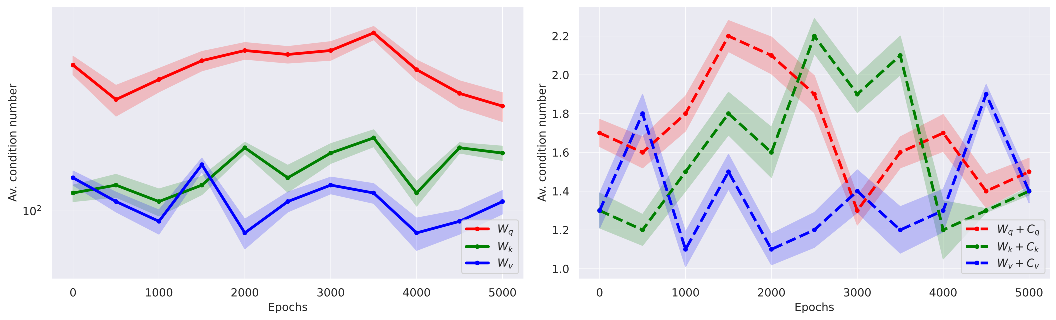

Figure 20: **Left:** Average condition number of the query, key, and value projection matrices (W_Q, W_K, W_V) for a Nyströmformer on the listops task during training. We plot the mean over five trials and the standard deviation. **Right:** Average condition number of the corrected query, key, and value projection matrices ($W_Q + C_Q$, $W_K + C_K$, $W_V + C_V$) for a Nyströmformer on the listops task during training. We plot the mean over five trials and the standard deviation.

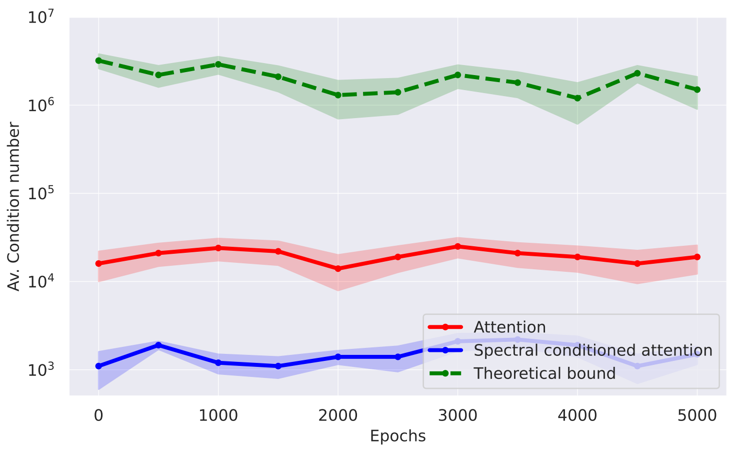

Figure 21: Average condition number of the self-attention Jacobian of a Nyströmformer on the listops task over the course of training, before and after spectral conditioning, along with the theoretical bound from Equation (10). We ran five trials with five different random seeds with plots showing the mean and standard deviations.