Unleashing the Potential of Large Language Models through Spectral Modulation

Peng Sun[1] 2023212399@email.cufe.edu.cn
Yao Zhu[2*] ee_zhuy@zju.edu.cn
Yunjian Zhang[2] sdtczyj@gmail.com
Xiu Yan[3] yanx18@tsinghua.org.cn
Zizhe Wang[2] wangzz@act.buaa.edu.cn
Xiangyang Ji[2] xyji@tsinghua.edu.cn

[1] School of Information, Central University of Finance and Economics
[2] Tsinghua University, [3] Meituan Group

Abstract

Large Language Models (LLMs) have demonstrated impressive capabilities across various domains, garnering significant attention from both academia and industry. However, enhancing the performance of LLMs typically requires scaling up model sizes or fine-tuning with additional datasets, which results in substantial computational costs. This paper poses an intriguing question: Can we improve the performance of LLMs without additional training? Drawing inspiration from signal processing principles, which suggest that noise often resides in high-frequency components while low-frequency components carry the essence of signals, we propose uncovering untapped potential in LLMs from a frequency perspective. We hypothesize that the high-frequency components in the weight matrices of LLMs' linear layers may conceal noise that interferes with predictive accuracy. Therefore, we propose conducting spectral modulation in the parameter space of LLMs, which can seamlessly integrate with various models in a plug-and-play manner. Extensive experiments have demonstrated the superiority of our approach, with spectral modulation yielding an average performance improvement of up to 10.12%.

1 Introduction

The emergence of Large Language Models (LLMs) (Touvron et al., 2023; Achiam et al., 2023; Hu et al., 2023; Huang et al., 2023) has revolutionized text generation quality across various generative tasks, becoming a central and widely discussed topic in the field of artificial intelligence. This advancement can be attributed to the exponential growth in both training data and model parameters, exemplified by models like GPT-3 (Brown et al., 2020) with its staggering 175 billion parameters. However, achieving better performance often requires scaling up model size or increasing training data (Kim et al., 2023; Jin et al., 2023), which raises the question of whether it is possible to further enhance the capabilities of pre-trained LLMs without the need for full-model fine-tuning or parameter-efficient fine-tuning methods (Ding et al., 2023; Jin et al., 2023).

In the field of signal processing (Kong et al., 2018; Sheng et al., 2022), signals are usually analyzed within the frequency domain. Here, noise, which is typically interwoven with the signal, is found mostly in high-frequency components, whereas low-frequency components constitute the essence of the signal. To improve signal quality, researchers leverage spectral modulation as a denoising technique. Turning to large language models (LLMs), could the process of training their vast number of parameters inadvertently introduce performance-constraining noise into the parameter space, akin to the inevitable introduction of noise during signal acquisition in signal processing? Drawing inspiration from this analogy, this paper proposes to explore the potential of LLMs from a novel frequency-domain perspective.

The transformer-based large language models are primarily composed of self-attention and feed-forward modules, which are mainly constructed by linear layers. In this paper, we treat the weight matrices of the linear layers in the model as two-dimensional matrices, and utilize two-dimensional Fourier transform to obtain their spectra. We hypothesize that the low-frequency components within weight matrices contain the critical information pivotal to determining model performance, whereas high-frequency components may conceal noise detrimental to model performance. Consequently, modulating these high-frequency components may be beneficial in unlocking the untapped potential of Large Language Models (LLMs).

Motivated by this hypothesis, this paper introduces a spectral modulation method applied to weight matrices, specifically aimed at preserving

*Corresponding author: Yao Zhu.

low-frequency components while reducing high-frequency ones, thereby enhancing model performance. This method involves two hyperparameters for each weight matrix: a decay factor for reducing high-frequency components and a protection threshold for preserving low-frequency components. The visual impact of spectral modulation on the weight matrices is provided in Appendix H. Recognizing that different weight matrices in LLMs may require different optimal hyperparameters for effective joint spectral modulation, we sample a small validation dataset and employ Bayesian optimization to determine the hyperparameters and report their performance on a held-out test set. The optimization objective is to maximize the model's prediction accuracy on the validation dataset.

Experimentally, we conduct a comprehensive evaluation of three types of LLMs across ten datasets and achieve superior performance compared to the vanilla model. We analyze the outputs of the model from a lexical composition perspective and find that the answers generated by our spectrally modulated model exhibit higher informativeness. Meanwhile, we discussed the impact of spectral modulation on model interpretability and found that spectral modulation also improves the model in this aspect. Furthermore, we investigate how the spectral modulation method generalizes to tasks beyond linguistic ones.

In summary, the proposed method is surprisingly simple yet effective, and, to the best of our knowledge, it represents the first attempt to discuss constraints in LLMs' performance from a frequency perspective and to perform spectral modulation on the weight matrices of LLMs. Extensive experiments on three existing open-source large language models across 10 datasets demonstrate the effectiveness of our proposed method, which leads to average performance improvements of up to 10.12% compared to the vanilla models.

2 Related Work

Frequency domain analysis occupies a critical position in the field of signal processing. This technique facilitates the elucidation of the frequency composition of signals, the identification of their intrinsic characteristics, and the analysis of noise components within these signals. Taking image signal processing as an example, frequency domain analysis has many typical applications, including image compression (Pennebaker and Mitchell, 1992; Wong and Noyes, 1994; Wen et al., 2022) and image denoising (Wang et al., 2006; Yue et al., 2014; Hasinoff et al., 2016; Li et al., 2022; Erkan et al., 2020; Sheng et al., 2022), among others. Due to page length limitations, a more detailed review of related literature on the application of frequency domain analysis is provided in Appendix F.

In recent years, there has been a growing emergence of endeavors integrating frequency analysis into deep learning. Xu et al. (2020) introduce a learning-based frequency selection method aimed at identifying trivial frequency components that could be eliminated without sacrificing accuracy, which can enhance the precision of image classification. Qin et al. (2021) extend the concept of channel attention to the frequency domain, proposing the multi-spectral channel attention framework. This framework exhibited promising performance across tasks such as image classification, object detection, and instance segmentation. Patro and Agneeswaran (2024) propose a Scattering Vision Transformer to more effectively capture fine-grained information within the input. This model first uses Fourier transform operations to implement a frequency-domain layer for feature extraction in the shallow layers of the network, and subsequently employs multi-head self-attention modules in the deeper layers for feature modeling. Chen et al. (2024) propose the Frequency Domain Kernelization method, which uses the Discrete Cosine Transform (DCT) to map the Transformer's Query and Key to the frequency domain. This approach effectively transforms the complexity of attention computation to linear time, thereby reducing training costs and improving inference speed.

Our approach diverges from previous studies as we investigate the effects of frequency domain processing on the parameter space rather than on the input space or the feature space. Specifically, our method achieves enhanced performance by processing model parameters in the frequency domain without requiring additional retraining phases.

3 Method

3.1 Preliminaries

Since we suggest processing the weight matrices within the LLMs, we first provide a brief description of the typical Transformer architecture in this subsection.

Transformer-based LLMs are typically comprised of L layers of Transformer blocks. The

l^{th} transformer block first employs a self-attention module to mix information across time steps and then uses a feed-forward module to process the information at each time step, thereby mapping a vector sequence $(\boldsymbol{h}_1^{(l-1)}, \cdots, \boldsymbol{h}_T^{(l-1)})$ of length T to $(\boldsymbol{h}_1^{(l)}, \cdots, \boldsymbol{h}_T^{(l)})$, where the dimensionality of the vector is d.

A single-head self-attention module first maps each vector $\boldsymbol{h}_i$ to a query vector $\boldsymbol{q}_i = \boldsymbol{h}_i \boldsymbol{W}_q$, a key vector $\boldsymbol{k}_i = \boldsymbol{h}_i \boldsymbol{W}_k$ and a value vector $\boldsymbol{v}_i = \boldsymbol{h}_i \boldsymbol{W}_v$, where $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{d \times d}$ are layer-specific weight matrices. Then the self-attention module mechanism an output using the scaled dot-production attention followed by a linear transformation $\boldsymbol{W}_o$ as:

$$\boldsymbol{u}_i = \Big(\sum_{j=1}^{T} \text{softmax}(\frac{\boldsymbol{q}_i \boldsymbol{k}_j^{\mathrm{T}}}{\sqrt{d}}) \boldsymbol{v}_j\Big) \boldsymbol{W}_o \ . \tag{1}$$

A k-head self-attention is comprised of multiple single-head self-attentions, which computes a set of attention vectors by using different linear transformations for key, query, and value, and then concatenates these attention vectors. Hence the dimensions of weight matrices in multi-head attention is related to the input vertor size and the number of attention heads: $\boldsymbol{W}_q \in \mathbb{R}^{d \times dk}$, $\boldsymbol{W}_k \in \mathbb{R}^{d \times dk}$, $\boldsymbol{W}_v \in \mathbb{R}^{d \times dk}$ and $\boldsymbol{W}_o \in \mathbb{R}^{dk \times d}$.

The feed-forward module is typically comprised of a 2-layer multi-layer perception (MLP) with a ReLU or GELU activation function (Hendrycks and Gimpel, 2016). We denote the weight matrices of the first and second linear layers of this MLP by $\boldsymbol{W}_{in}$ and $\boldsymbol{W}_{out}$ respectively.

In summary, the core architecture of LLMs entails the following weight matrices $\mathcal{W} = \{\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v, \boldsymbol{W}_o, \boldsymbol{W}_{in}, \boldsymbol{W}_{out}\}$ for each layer. In our work, we primarily focus on the matrices within $\mathcal{W}$, and enhance model performance by spectral modulation. It is worth noting that Transformer blocks within different LLMs often exhibit small differences. These preliminaries aim to outline the terminology essential for our approach rather than present an full survey of these nuances.

3.2 Spectral Modulation

In the introduction section, we outlined the motivation behind this work. In this section, we formally describe how spectral modulation can be applied to LLMs to enhance model performance. A single-step spectral modulation is defined by four factors $(\tau, \ell, \alpha, \rho)$, including hyperparameters describing the specific matrix to be processed: the weight matrix type τ and the layer number ℓ, and hyperparameters required for spectral modulation: the protection threshold α and the reduction factor ρ.

Fig. 1 illustrates an example of our single-step spectral modulation. In this example, we have τ = W_{in} and $\ell = L$, indicating that we modify the weight matrix $W_{in} \in \mathbb{R}^{M \times N}$ of the Feed Forward module in the L^{th} Transformer block. For clarity of description, here we denote the dimension of W_{in} as $M \times N$. We regard the weight matrix $W_{in}(m, n)$ as a two-dimensional discrete signal sampled in the spatial domain. Then, its two-dimensional discrete Fourier transform can be defined as:

$$F(u,v) = \tfrac{1}{\sqrt{MN}} \textstyle\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} W_{in}(m,n) e^{-\mathrm{j}2\pi(\frac{mu}{M} + \frac{nv}{N})}, \tag{2}$$

where u and v are discrete frequency variables, $u = 0, 1, \cdots, M-1$; $v = 0, 1, \cdots, N-1$. We then centralize the spectrum of the weight matrix, with the smallest frequency at the center and increasing frequencies further from the center. We apply a Fourier mask $\mathcal{M}_{\tau,\ell}$ in the shape of $M \times N$ to the centralized spectrum:

$$F^*(u,v) = \text{Centering}(F(u,v)) \odot \mathcal{M}_{\tau,\ell} \ , \tag{3}$$

where $\odot$ denotes element-wise multiplication and the Fourier mask $\mathcal{M}_{\tau,\ell}$ consists of a protection threshold α and a reduction factor ρ, serving to diminish high-frequency components in the spectrum while preserving low-frequency components:

$$\mathcal{M}_{\tau,\ell}(x,y) = \begin{cases} 1 & \text{if } |x| \leq \alpha \text{ and } |y| \leq \alpha \ , \\ \rho & \text{otherwise} \ . \end{cases} \tag{4}$$

Here we denote the center point in the Fourier mask $\mathcal{M}_{\tau,\ell}$ as $(0, 0)$ for clarity.

At last, we perform inverse centering on the spectrum F^* to obtain F', and utilize the inverse Fourier transform to convert the weight matrices back from the frequency domain to the spatial domain:

$$W_{in}^*(m,n) = \tfrac{1}{\sqrt{MN}} \textstyle\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F'(u,v) \, e^{\mathrm{j}2\pi(\frac{mu}{M} + \frac{nv}{N})} \tag{5}$$

So far, we have introduced the process of single-step spectral modulation. This process applies spectral modulation on a single weight matrix, thus involving only two hyperparameters. By exhaustively searching the specified parameter combinations through grid search, we can simply select the

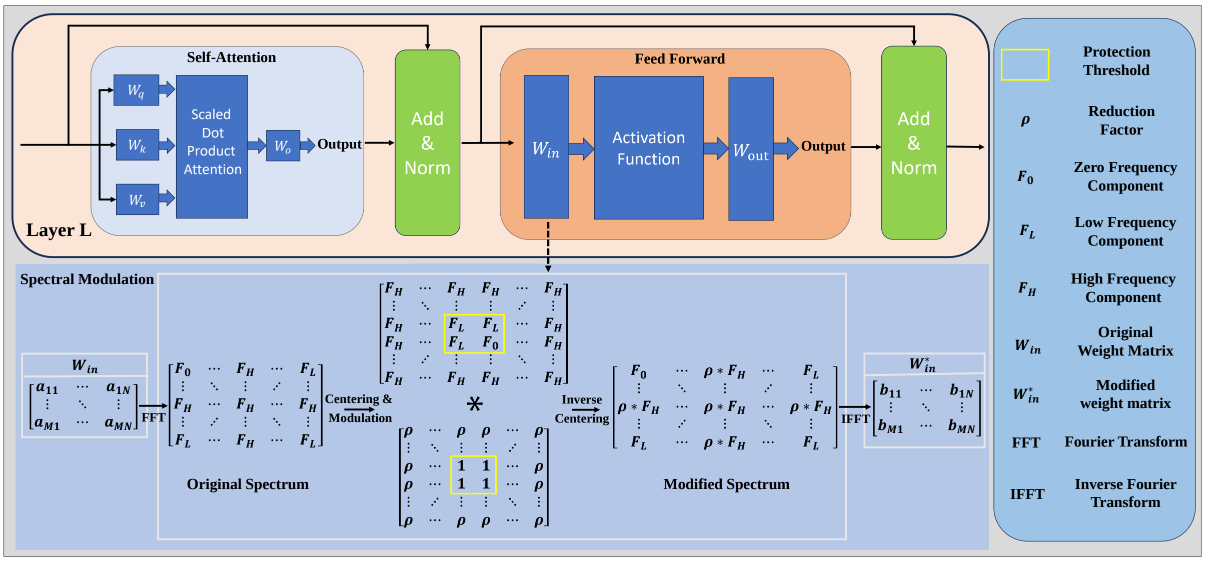

Figure 1: We enhance model performance by applying spectral modulation to specific weight matrices within the Transformer block, using the W_{in} matrix in the Feed Forward module as an example.

best-performing parameters α and ρ on the validation set to adjust the model and achieve improved performance on the test set.

Furthermore, how can we perform joint spectral modulation on multiple layers of the model to fully exploit the advantages of spectral modulation methods? Here, two challenges arise. Firstly, potential interactions between modulation operations on different matrices may exist, complicating the direct combination of optimal hyperparameters selected for single-step modulation. Secondly, joint modulation entails considering a large number of parameters, including various layer numbers and matrices within each layer, which leads to high computational costs for direct search. Hence, we propose using Bayesian optimization (Snoek et al., 2012) to search for the optimal parameters for joint spectral modulation. Bayesian optimization is an iterative method used for global optimization, aimed at identifying the global optimal solution of an objective function, particularly effective in cases where the objective function has a high computational cost or is non-differentiable. The method is based on Bayesian inference principles, approximating the posterior distribution of the objective function by constructing a Gaussian process model in the search space. Bayesian optimization can discover the global optimal parameter combinations of the objective function in a few iterations. In our scenario, the objective of Bayesian optimization is to maximize validation accuracy.

Our proposed spectral modulation can be effortlessly implemented with just a few lines of code. Additionally, this approach imposes no additional computational burden during inference, rendering it a highly practical solution that seamlessly integrates into existing LLMs to boost their performance. It is worth noting that, the focus of this work is to propose a promising approach to explore the potential of LLMs from a frequency perspective. While we believe there are many other spectral modulation methods and hyperparameter selection techniques, we leave this exploration to future work.

4 Experiments

In this section, we first introduce the experimental setup and then investigate the impact of hyperparameters on model performance with single-step spectral modulation. Next, we further enhance the performance of LLMs through multi-step spectral modulation. We then discuss the impact of spectral modulation on lexical composition to gain a deeper understanding of our approach. Following this, we compare spectral modulation with the existing method and explore its generalization performance on visual tasks. Additionally, we discuss the effect of spectral modulation on model's interpretability in Appendix I.

4.1 Experimental Setup

To evaluate the effect of spectral modulation on model performance, we assess the performance of language models before and after applying spectral

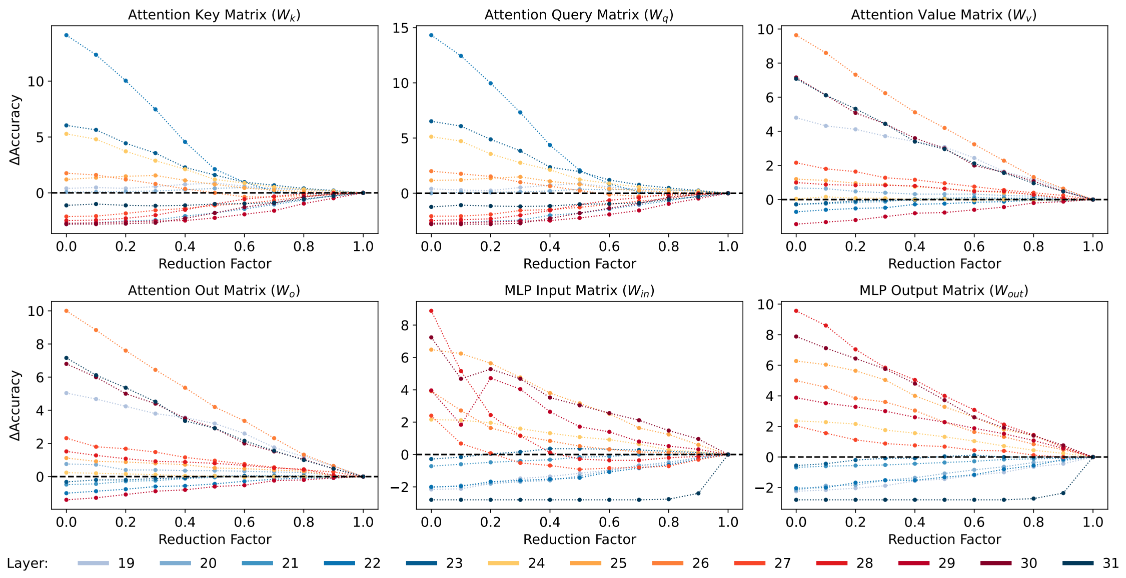

Figure 2: The effect of applying single-step spectral modulation to different weight matrices on model's performance on the FEVER dataset. The vertical axis shows the performance gain of the model with spectral modulation compared to the vanilla model, while the horizontal axis indicates the reduction factor applied to specific layers.

modulation on ten tasks. These tasks include CounterFact (Elazar et al., 2021), FEVER (Thorne et al., 2018), Bios Gender (De-Arteaga et al., 2019), Bios Profession (De-Arteaga et al., 2019), Truthful QA (Lin et al., 2021), BigBench Epistemic Reasoning (Srivastava et al., 2022), BigBench Wikidata QA (Srivastava et al., 2022), Stanford Sentiment Treebank 2 (SST-2) (Socher et al., 2013; Wang et al., 2018), Race (Lai et al., 2017), and Question Natural Language Inference (Wang et al., 2018). These tasks broadly test the language model's reasoning abilities, language understanding, and factuality. The hyperparameters involved in spectral modulation are selected based on a small validation set, and results are reported on the test set. For datasets that only provide a test set, we split them into separate validation and test sets. The models used for these tasks include GPT-J-6B (Wang and Komatsuzaki, 2021), Vicuna-7B-V1.5 (Zheng et al., 2024), and the newly released Llama3-8B [1]. We use the HuggingFace implementation for all three LLMs and run experiments on NVIDIA GeForce RTX 4090 GPUs with 24GB of memory. Due to page length limitations, specific details about the datasets and evaluation metrics are provided in Appendix.A and Appendix.B. Each computational experiment in this paper was conducted three times, and average results were reported.

[1]https://github.com/meta-llama/llama3

4.2 The Effect of Single-Step Spectral Modulation

In this section, we investigate the impact of hyperparameters on model performance when applying spectral modulation to specific parameter matrices.

Fig.2 illustrates the variation in model performance on the FEVER dataset when spectral modulation with different reduction factors is applied to different weight matrices of the model. More experimental results can be found in Appendix C. In this experiment, the protection threshold is set at a value of 256. From Fig. 2, two observations can be gleaned. First, the influence of the reduction factor varies across different layers and types of weight matrices. For instance, applying spectral modulation to the W_{out} matrix in layer 30 yields positive performance gains, which increase as the reduction factor decreases. Conversely, spectral modulation always has an adverse effect on the W_k matrix of the same layer. Second, on the FEVER dataset, applying spectral modulation to the feed-forward components (i.e., W_{in} and W_{out}) within the seven layers immediately preceding the last layer shows potential for enhancing model performance. While spectral modulation on other weight matrices may also improve model performance, to balance computational cost and performance, subsequent results in this paper primarily focus on joint spectral modulation through Bayesian optimization on the feed-forward modules of these seven layers, unless otherwise specified.

Dataset	Model Name					
	GPT-J-6B	GPT-J-6B*	Vicuna-7B-V1.5	Vicuna-7B-V1.5*	Llama3-8B	Llama3-8B*
CounterFact	14.16	**27.59**	34.91	**37.53**	40.47	**46.78**
FEVER	50.28	**59.63**	53.33	**71.68**	65.90	**70.26**
Bios Gender	70.75	**79.02**	99.21	**99.26**	95.40	**98.12**
Bios Profession	75.32	**82.12**	81.97	**83.68**	91.96	**92.06**
TruthfulQA	54.87	**55.95**	44.56	**56.12**	44.92	**55.93**
BigBench-Epistemic Reasoning	37.13	**46.38**	50.25	**62.88**	40.06	**62.50**
BigBench-WikidataQA	51.94	**66.44**	59.91	**63.06**	62.31	**64.59**
Stanford Sentiment Treebank 2	54.75	**54.75**	62.53	**70.06**	70.84	**75.00**
Race	44.66	**73.50**	57.75	**76.59**	52.91	**72.03**
Question Natural Language Inference	50.81	**60.50**	64.19	**67.63**	51.88	**59.47**
Average	50.47	**60.59**	60.86	**68.85**	61.67	**69.67**

Table 1: Effect of joint spectral modulation on ten Question Answering datasets. We determine the best hyperparameters for each model and task using Bayesian optimization on a validation set and report their performance on a held-out test set. An asterisk (*) indicates enhanced LLMs using spectral modulation, while LLMs without an asterisk represent the vanilla models. The best results are highlighted in bold.

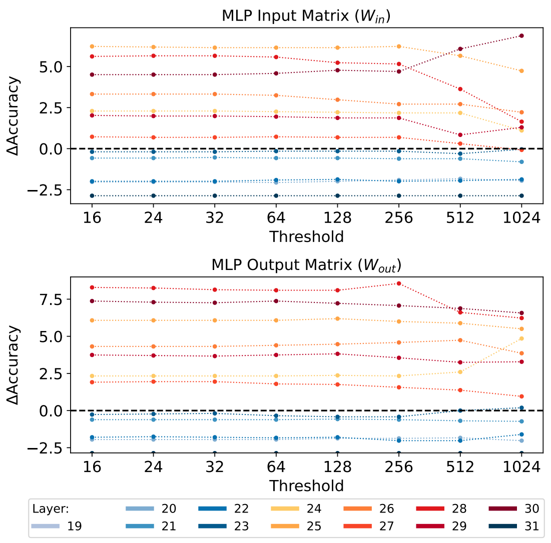

Figure 3: The effect of applying single-step spectral modulation to different weight matrices on the model's performance on the FEVER dataset. The horizontal axis indicates the protection threshold. Here, the reduction factor is set to 0.1.

Next, we examine the variations in model performance on the FEVER dataset when applying spectral modulation with different protection thresholds to various weight matrices in the model's feed-forward module, as illustrated in Fig. 3. The reduction factor was fixed at 0.1. A larger protection threshold signifies that a greater number of frequency components in the parameter matrix are preserved from modification. In most layers where positive performance gains are observed, such as the 26th, 27th, and 28th layers, decreasing the protection threshold from 1024 to 64 yields increased performance improvements, which subsequently stabilize. However, when applying spectral modulation to the W_{in} weight matrices of the 30th layer, performance gains are observed to be greater with a protection threshold of 1024 compared to 64. Although individually optimizing the protection threshold for each weight matrix could fully harness the potential of spectral modulation, we note the substantial performance improvements already realized with protection threshold set to 64. Consequently, with the objective of simplifying the parameter search space for Bayesian optimization in the context of joint spectral modulation, we adopt a protection threshold of 64 for subsequent experiments, unless otherwise specified.

4.3 The Superiority of Joint Spectral Modulation

Based on the experimental observations and parameter settings from the previous subsection, we employed Bayesian optimization to search the optimal parameters for joint spectral modulation and validated the effectiveness of our method across ten test datasets. With the assistance of spectral modulation, the widely used Vicuna-7B-V1.5 model achieves a 2.62% performance gain on the CounterFact dataset, a 18.35% performance gain on the FEVER dataset, and an average performance gain of 7.99% across the ten datasets. Assisted by spectral modulation, the recently released popular open-source large model Llama3-8B also achieves notable performance gains. Llama3-8B achieved a 6.31% performance gain on the CounterFact dataset, a 4.36% performance gain on the FEVER dataset, and an average performance gain of 8.00% across the ten datasets. In summary, joint spectral modulation consistently delivers performance gains across various datasets and models, requiring no additional training. We further evaluated the performance of joint spectral modulation using a conservative strategy rather than Bayesian optimization in Appendix D.

4.4 Impact on Lexical Composition

To further understand the changes brought about by spectral modulation, we analyze the outputs of the vanilla model, the spectrally modulated model, and their corresponding high-frequency counterparts in terms of lexical composition. The difference between the high-frequency model and the spectrally modulated model lies in the fact that the former retains the high-frequency components in the weight matrix while filtering out the low-frequency components, whereas the latter does the opposite.

Tab. 2 presents the lexical composition of answers for 200 samples in the Counterfact dataset, including nouns, verbs, prepositions, conjunctions, determiners, adjectives, adverbs, pronouns, numerals, and punctuation. The number of generated tokens is limited to a maximum of 10. The answers generated by our model are particularly notable for their use of nouns and verbs, which contain the most information. Additionally, these answers exhibit a higher frequency of prepositions and conjunctions, enhancing sentence fluency. In contrast, answers from the high-frequency model feature the highest usage of adverbs and numerals, focusing mainly on supplementing existing information. The vanilla model's answers have the highest usage of determiners and pronouns, which contain the least amount of information. In summary, the answers generated by our model, which reduce high-frequency components, have a higher information content. This likely contributes to improved generation accuracy (See Appendix B). In contrast, the vanilla model and the high-frequency model produce answers with less information content, potentially leading to inferior performance.

Tab. 3 provides two specific examples. The answer for the first question generated by the spectrally modulated model provides a specific geographic location. It uses the most nouns and the fewest determiners, making the information specific and detailed. The answers from the vanilla model and the high-frequency model provide shared building information but lack specific geographic location details, making the information somewhat vague and abstract. The phenomenon observed in the second example is similar, with our model providing more specific responses compared to the vanilla model, which tends to give vaguer answers. Appendix E provides additional discussion of the lexical composition of model-generated answers on other datasets.

Component		Ours	High	Vanilla
NOUN	Sum	**561**	514	493
	Average	**2.81**	2.57	2.47
VERB	Sum	**219**	184	148
	Average	**1.10**	0.92	0.74
PREP	Sum	**186**	182	170
	Average	**0.93**	0.91	0.85
CONJ	Sum	**108**	91	67
	Average	**0.54**	0.46	0.34
DET	Sum	176	199	**253**
	Average	0.88	1.00	**1.27**
ADJ	Sum	**125**	124	79
	Average	**0.63**	0.62	0.40
ADV	Sum	34	**42**	28
	Average	0.17	**0.21**	0.14
PRON	Sum	42	46	**54**
	Average	0.21	0.23	**0.27**
NUM	Sum	28	**43**	42
	Average	0.14	**0.22**	0.21
PUNCT	Sum	133	125	**140**
	Average	0.67	0.63	**0.70**

Table 2: The lexical composition analyze on the answers for the CounterFact dataset. “Vanilla", “Ours", “High" refer to answers derived from the vanilla Vicuna-7B, our spectrally modulated model, and the high-frequency model, respectively. “Sum" is the total occurrences of the corresponding component across all samples, while “Average" denotes the average occurrences of that component per sample. The maximum numbers are in bold.

4.5 Comparison with the Layer-Selective Rank Reduction Method

Sharma et al. (2023) found that carefully selected rank reductions can enhance Transformer performance and proposed the LAyer-SElective Rank reduction (LASER) method. How does our method differ from LASER? Motivationally, our approach is inspired by common denoising techniques in the field of signal processing, whereas LASER is motivated by the popular parameter-efficient fine-tuning method in the LLM domain, Low-Rank Adaptation (LoRA) (Hu et al., 2021). In terms of methodology, our method applies spectral modulation in the model's parameter space, while LASER performs low-rank approximation. In terms of effectiveness, our method outperforms LASER on multiple datasets, as illustrated in Fig.4.Our method and LASER enhance the performance of LLMs in a post-hoc manner from two different perspectives, with our approach achieving superior results. We believe there are still other methods that can im-

Question	There are more than 60 establishments across the avenue. The headquarters of Overkill Software is in	The language of Fully Booked is
Label	**Stockholm**	**English**
Ours	**Stockholm**, Sweden. The company was founded in	a mix of Filipino and **English**. nobody is
High	the same building. The building is located	a mix of formal and informal language. nobody
Vanilla	the same building as the studio. The	a language of the heart. nobody can read it

Table 3: Specific examples in the Counterfact dataset. 'Question' refers to the model input, and 'Label' refers to the correct answer. "Vanilla," "Ours," and "High" refer to answers derived from the vanilla Vicuna-7B-V1.5, our spectrally modulated model, and the high-frequency model, respectively. Correct answers are in bold.

prove the performance of LLMs, awaiting further exploration.

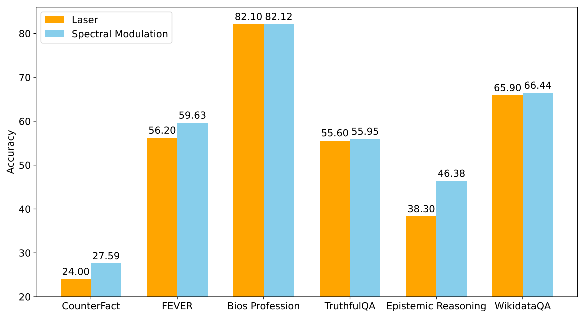

Figure 4: Comparison of our method and LASER across different datasets with GPT-J-6B.

4.6 The Effect of Spectral Modulation in Vision Tasks

To understand whether the findings of this study can generalize beyond language tasks, we conducted an exploration of the effect of spectral modulation on computer vision tasks. In our experiments, we selected the ViT-B/32 model pre-trained on ImageNet (Wightman, 2019), as well as the ViT-B/32 and ResNet-50 models obtained using Contrastive Language-Image Pretraining (CLIP) (Radford et al., 2021), and tested their performance on ImageNet (Deng et al., 2009), ImageNet-V2 (Recht et al., 2019), and ImageNet-Sketch (Wang et al., 2019) datasets. As shown in Tab. 4, models with spectral modulation consistently achieved superior performance compared to the original models. Specifically, for the ResNet-50 model pretrained with CLIP, spectral modulation resulted in performance gains of 0.26% and 0.16% on the ImageNet and ImageNet-V2 datasets, respectively. Although these performance gains are smaller compared to those achieved in the textual domain, it is noteworthy that these improvements were achieved solely through spectral modulation without any additional training or fine-tuning of the pre-trained models, thus incurring almost no extra cost. In Appendix G, we visualize some images that were misclassified by the original model but correctly classified by the spectrally modulated model.

Model	ImageNet	ImageNet-V2	ImageNet-S
CLIP-RN50	59.82	52.89	35.44
CLIP-RN50*	**60.08**	**53.05**	**35.59**
CLIP-ViT-B/32	60.22	52.40	46.45
CLIP-ViT-B/32*	**60.38**	**52.63**	**46.74**
ViT-B/32	78.75	66.41	29.65
ViT-B/32*	**78.82**	**66.54**	**29.89**

Table 4: The effect of the proposed spectral modulation in vision tasks. The asterisk (*) indicates enhanced deep model using spectral modulation, while models without an asterisk represent vanilla models.

5 Conclusions

This paper revisits the factors limiting the performance of large language models (LLMs) from a frequency perspective, hypothesizing that high-frequency components in the weight matrices of LLMs may carry noise detrimental to model performance. To address this, we propose a straightforward, retraining-free spectral modulation method that enhances model performance by reducing high-frequency components while preserving low-frequency components in the weight matrices. Extensive experiments conducted on ten different datasets and three distinct LLMs demonstrate the effectiveness of our method. Moreover, we analyzed the lexical composition in the generated outputs from the spectrally modulated models and their high-frequency counterparts, which suggests that spectral modulation contributes to more specific and complete outputs. Additionally, experiments on the classic ImageNet classification task demonstrate the effectiveness of the method's generalization to non-linguistic tasks. We hope that our findings can inspire further research into the internal mechanisms of LLMs from the perspective of spectral analysis, ultimately leading to the development of more reliable and effective LLMs.

Limitations

This paper provides novel insights into the performance obstacles of large language models from a frequency domain perspective and introduces a method to enhance performance through spectral modulation of weight matrices, requiring no additional training.

A limitation of our proposed approach is the inevitable exclusion of high-frequency information, whose role in the model remains uncertain. Although we conducted extensive experiments on multiple benchmark datasets to demonstrate the effectiveness and broad applicability of our proposed method, we must acknowledge that there may be specific tasks where high-frequency information should not be diminished. Decoupling components detrimental to model performance from the frequency domain of the language model's parameter space without inadvertently losing beneficial components is a question worthy of further exploration, which we leave for future work.

Another limitation might be the relatively simple setting of hyperparameters in the proposed joint spectral modulation method. For instance, the experimental results in Fig. 2 indicate that modulating the spectra of parameter matrices like W_k and W_q could also enhance model performance. However, to avoid compromising the performance and computational efficiency of Bayesian optimization with too many hyperparameters, we applied joint spectral modulation only to the feed-forward modules in the last seven layers of the model. This approach may not fully exploit the advantages of joint spectral modulation. Nevertheless, the focus of this paper is to analyze performance limitations from a frequency domain perspective and to present a method that effectively improves model performance. Further exploration of better joint spectral modulation methods is left for future work.

Ethical Considerations

We believe that our method does not inherently pose potential negative societal impacts, as its primary objective is to enhance the performance of language models. However, concerns about the potential misuse of language models leading to negative societal impacts do exist. This issue is not unique to our method; it is a shared concern applicable to all language models, meriting continued attention and research by the academic community in the times ahead.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 1877–1901.

Hanting Chen, Zhicheng Liu, Xutao Wang, Yuchuan Tian, and Yunhe Wang. 2024. Dijiang: Efficient large language models through compact kernelization. *arXiv preprint arXiv:2403.19928*.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Uğur Erkan, Serdar Enginoğlu, Dang NH Thanh, and Le Minh Hieu. 2020. Adaptive frequency median filter for the salt and pepper denoising problem. *IET Image Processing*, 14(7):1291–1302.

Samuel W Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. 2016. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Transactions on Graphics (ToG)*, 35(6):1–12.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units. *arXiv preprint arXiv:1606.08415*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. 2023. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068.

Feihu Jin, Jiajun Zhang, and Chengqing Zong. 2023. Parameter-efficient tuning for large language model without calculating its gradients. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 321–330.

Sungdong Kim, Sanghwan Bae, Jamin Shin, Soyoung Kang, Donghyun Kwak, Kang Yoo, and Minjoon Seo. 2023. Aligning large language models through synthetic feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13677–13700.

Qinglin Kong, Qian Song, Yan Hai, Rui Gong, Jietao Liu, and Xiaopeng Shao. 2018. Denoising signals for photoacoustic imaging in frequency domain based on empirical mode decomposition. *Optik*, 160:402–414.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*.

Dasong Li, Yi Zhang, Ka Lung Law, Xiaogang Wang, Hongwei Qin, and Hongsheng Li. 2022. Efficient burst raw denoising with variance stabilization and multi-frequency denoising network. *International Journal of Computer Vision*, 130(8):2060–2080.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *Preprint*, arXiv:2109.07958.

Badri Patro and Vijay Agneeswaran. 2024. Scattering vision transformer: Spectral mixing matters. *Advances in Neural Information Processing Systems*, 36.

William B Pennebaker and Joan L Mitchell. 1992. *JPEG: Still image data compression standard*. Springer Science & Business Media.

Zequn Qin, Pengyi Zhang, Fei Wu, and Xi Li. 2021. Fcanet: Frequency channel attention networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 783–792.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2019. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pages 5389–5400. PMLR.

Soumya Sanyal and Xiang Ren. 2021. Discretized integrated gradients for explaining language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10285–10299.

Pratyusha Sharma, Jordan T Ash, and Dipendra Misra. 2023. The truth is in there: Improving reasoning in language models with layer-selective rank reduction. *arXiv preprint arXiv:2312.13558*.

Zehua Sheng, Xiongwei Liu, Si-Yuan Cao, Hui-Liang Shen, and Huaqi Zhang. 2022. Frequency-domain deep guided image denoising. *IEEE Transactions on Multimedia*.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *NAACL-HLT*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. 2019. Learning robust global representations by penalizing local predictive power. *Advances in Neural Information Processing Systems*, 32.

Jin Wang, Yanwen Guo, Yiting Ying, Yanli Liu, and Qunsheng Peng. 2006. Fast non-local algorithm for image denoising. In *2006 International Conference on Image Processing*, pages 1429–1432. IEEE.

Heping Wen, Linchao Ma, Linhao Liu, Yiming Huang, Zefeng Chen, Rui Li, Zhen Liu, Wenxing Lin, Jiahao Wu, Yunqi Li, et al. 2022. High-quality restoration image encryption using dct frequency-domain compression coding and chaos. *Scientific Reports*, 12(1):16523.

Ross Wightman. 2019. Pytorch image models. https://github.com/rwightman/pytorch-image-models.

Ping Wah Wong and Steven Noyes. 1994. Space-frequency localized image compression. *IEEE transactions on image processing*, 3(3):302–307.

Kai Xu, Minghai Qin, Fei Sun, Yuhao Wang, Yen-Kuang Chen, and Fengbo Ren. 2020. Learning in the frequency domain. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1740–1749.

Huanjing Yue, Xiaoyan Sun, Jingyu Yang, and Feng Wu. 2014. Cid: Combined image denoising in spatial and frequency domains using web images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2933–2940.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Yao Zhu, Yuefeng Chen, Xiaodan Li, Rong Zhang, Xiang Tian, Bolun Zheng, and Yaowu Chen. 2023. Information-containing adversarial perturbation for combating facial manipulation systems. *IEEE Transactions on Information Forensics and Security*, 18:2046–2059.

A Dataset Processing Details

For each dataset in Table 1, we use 20% of the processed dataset for hyperparameter search, while the remaining 80% is used as the test dataset to evaluate the effectiveness of the hyperparameters. We describe the specific dataset processing as follows:

CounterFact (Elazar et al., 2021). The dataset consists of approximately 20,000 examples, each with 3 paraphrases, resulting in a total of 65,757 examples. For the GPT-J-6B model, the label is always one token long. However, for the Llama3-8B model and Vicuna-7B-V1.5 model, the label can consist of multiple tokens. Our experimental framework employs the question-answer format, where the input prompted question is designed based on the original dataset.

FEVER (Thorne et al., 2018). We merge the dev and test split of the original FEVER dataset by initially filtering out samples labeled as 'NOT ENOUGH INFO', retaining only those categorized under 'SUPPORTS' and 'REFUTES'. Subsequently, we eliminate entries featuring identical claims with differing labels. This process yields a dataset comprising 13,086 samples, with 6,510 deriving from the original dev set and 6,576 deriving from the original test set. The experimental framework employs a question-answer format where the input correlates each claim to a prompted question, formatted as <*"Is the following claim true or false: " + question.strip() + " The claim is"*> where the question is derived from the claim by adjusting its punctuation. If the question does not end with a period or question mark, we add a period to it. Concurrently, in the model's output, the labels 'SUPPORTS' and 'REFUTES' are systematically mapped to true (1) and false (0), respectively, enabling a direct comparison with the predicted outcomes.

Bios Gender (De-Arteaga et al., 2019). We adopt the dev split of the original Bias in Bios dataset to create a dataset comprising 39,642 samples. The task of this dataset is to determine the gender of the people in the textual biographies (denoted as bios) part of the samples. There are only two possible labels: male and female. The experimental framework employs a question-answer format where the input correlates each bios to a prompted question, formatted as <*"Consider the following text: " + bios + " Is the person in this text male or female? The person is"*>. We do not process the bios part.

Bios Profession (De-Arteaga et al., 2019). We adopt the dev split of the original Bias in Bios dataset. Our objective is to predict the occupations of the individuals in the samples. Here we limit the predictions to occupations that can be represented as a single token. For the Llama3-8B model, the relevant professions include accountant, attorney, comedian, composer, dentist, filmmaker, nurse, model, teacher, dj, and rapper. For the GPT-J-6B model and Vicuna-7B-V1.5 model,the professions are journalist, poet, composer, model, teacher, architect, painter, and professor. Based on the criteria, we obtained datasets containing 11,383 and 19,223 samples, respectively. Our experimental framework is in the form of question-answer, where the prompted question transforms the bios part of the sample into <*"Consider the following text: " + bios.strip() + " What is the profession of the person in this text? The profession of this person is"*>. It is crucial to ensure that if the bios part of the sample ends with a period or a question mark, we use it as is for the question. If not, we append a period to ensure grammatical correctness before forming the question.

Truthful QA. (Lin et al., 2021) We adopt the validation split of the Truthful QA dataset to create a dataset comprising 5,882 samples. The dataset consists of multiple-choice questions, with each question having multiple answer options, including both correct and incorrect ones. We treat each question along with each of its answers individually, forming separate samples. For instance, a question with four answers is translated into four samples. Our experimental framework employs the question-answer format. We combine each question with its answers to create multiple prompted questions. These are structured as <*question + " " + answer + " Is this statement true or false? This statement is"*>. It is essential to ensure that each answer ends with a period or a question mark; if any answer does not, we add a period to ensure consistency. Each prompted question corresponds to the answer true or false. After processing the dataset, we obtained the corresponding question-answer pairs.

BigBench Epistemic Reasoning (Srivastava et al., 2022). We consolidate the validation and train split of the Big Bench epistemic reasoning dataset to create a dataset consisting of 2000 samples. The experimental framework adopts a question-choice format, where the prompted question directly utilizes the original question form in the sample, and the choice is taken to be the one with the highest model-generated probability among the label options *["entailment", "non-entailment"]*.

BigBench Wikidata QA (Srivastava et al., 2022). We consolidate the validation and train split of the Big Bench Wikidata QA dataset, filtering out examples where the number of target labels exceeds one. Our experimental framework utilizes a question-answer format, with the input questions designed based on the original dataset. The labels are always composed of multiple tokens.

Stanford Sentiment Treebank 2 (SST-2) (Socher et al., 2013; Wang et al., 2018). We merge the train and dev split of the Stanford Sentiment Treebank 2 (SST-2) dataset from the GLUE benchmark. This dataset includes sentences that have been annotated by humans. These sentences are extracted from movie reviews and sorted into two categories based on emotional valence: positive emotions (marked as 1) and negative emotions (marked as 0). Our experimental framework adopts a question-choice format. The structure of the input prompted question is as follows: <*"Consider the following comment: " + question.strip() + " Is the sentiment expressed in this comment positive or negative? The result is"*>. Consistently formatting the question based on the ending of the comment is crucial. If a comment in a sample concludes with a period, semicolon, question mark, or exclamation mark, the question should replicate the exact sentence as it appears. If the comment lacks one of these punctuation marks, a period should be added before formulating the question.

Race (Lai et al., 2017) We merge the train, validation, and test split of the RACE dataset into a single pool. This dataset is designed for reading comprehension tasks. Each sample comprises a passage, a corresponding question, and four answer choices, where only one option is correct. To augment the data, each sample is expanded by combining the text and question with each of the four answer choices separately, thereby transforming each original sample into four distinct samples. Our experimental framework utilizes a question-answer format. The input prompted question is structured as follows: <*"For the passage " + "passage.strip()" + ", the question is " + "question.strip()" + "is the answer " + "option.strip()" + " to this question right or wrong? The result is"*>. Here, passage is the text, question is the corresponding question, and option is one of the four options for the question.

Question Natural Language Inference (QNLI) (Wang et al., 2018). We merge the train and dev split of the Question Natural Language Inference (QNLI) dataset from the GLUE benchmark. The experimental framework adopts a question-choice format, where the prompted question directly utilizes the original question form in the sample, and the choice is taken to be the one with the highest model-generated probability among the label options *["entailment", "non-entailment"]*.

B Details for Evaluation Metrics

For different datasets, we employ distinct methods to calculate accuracy. These methods are categorized into two types:

Classification Accuracy: For datasets with a limited set of possible labels, where each label is single token long, we decode the token with the highest predicted probability as the predicted label. If it matches the ground-truth label, we consider the prediction as correct.

Generation Accuracy: For datasets with open-ended labels, where each label may consist of multiple tokens, we preprocess both the generated text and the label by converting them to lowercase and stripping whitespaces. The prediction is correct if the ground-truth label is contained within the generated text.

CounterFact(Elazar et al., 2021). We evaluate the correctness of our predictions based on generation accuracy. For the GPT-J-6B model, we utilize the first token generated as the output since all labels are single token long. For the Llama3-8B model and Vicuna-7B-V1.5 model, we set the maximum generation length to 10 tokens, which are then used as the generated text.

FEVER(Thorne et al., 2018). We evaluate the correctness of our predictions based on classification accuracy. We select the label from the set *[true, false]* that has the highest probability in the model's prediction results as the predicted label.

Bios Gender(De-Arteaga et al., 2019). We evaluate the correctness of our predictions based on classification accuracy. We select the label from the set *[male, female]* that has the highest probability in the model's prediction results as the predicted label.

Bios Profession(De-Arteaga et al., 2019). We evaluate the correctness of our predictions based on classification accuracy. For the Llama3-8B model, we select the label from the set *[accountant,*

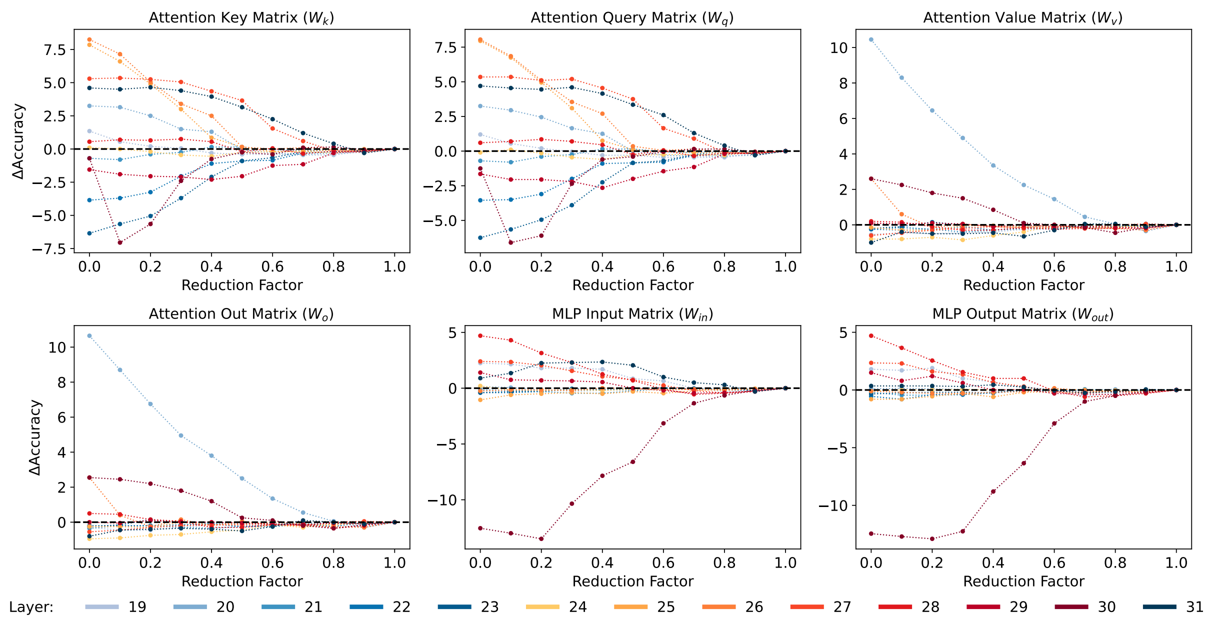

Figure 5: The effect of applying spectral modulation to different layers on the performance of Vicuna-7B-V1.5 on the BigBench-Epistemic Reasoning dataset. The vertical axis shows the performance gain of the model with spectral modulation compared to the vanilla model, while the horizontal axis indicates the reduction factor applied to specific layers. In this experiment, the protection threshold is set to 256.

attorney, comedian, composer, dentist, filmmaker, nurse, model, teacher, dj, rapper] that has the highest probability in the model's prediction results as the predicted label. For the GPT-J-6B model and Vicuna-7B-V1.5 model, we select the label from the set *[journalist, poet, composer, model, teacher, architect, painter, professor]* that has the highest probability in the model's prediction results as the predicted label.

Truthful QA.(Lin et al., 2021) We evaluate the correctness of our predictions based on classification accuracy. We select the label from the set *[true, false]* that has the highest probability in the model's prediction results as the predicted label.

BigBench Epistemic Reasoning(Srivastava et al., 2022). We evaluate the correctness of our predictions based on classification accuracy. We select the label from the set *[entailment, non-entailment]* that has the highest probability in the model's prediction results as the predicted label.

BigBench Wikidata QA(Srivastava et al., 2022). As the set of labels are open-ended, we compute the generation accuracy similar to CounterFact. For the three LLMs we generate 10 tokens uniformly.

Stanford Sentiment Treebank 2 (SST-2)(Socher et al., 2013; Wang et al., 2018). We evaluate the correctness of our predictions based on classification accuracy. We select the label from the set *[positive, negative]* that has the highest probability in the model's prediction results as the predicted label.

Race(Lai et al., 2017) We evaluate the correctness of our predictions based on classification accuracy. We select the label from the set *[right, wrong]* that has the highest probability in the model's prediction results as the predicted label.

Question Natural Language Inference (QNLI)(Wang et al., 2018). We evaluate the correctness of our predictions based on classification accuracy. We select the label from the set *[entailment, non-entailment]* that has the highest probability in the model's prediction results as the predicted label.

C The Effect of the Reduction Factor on Epistemic Reasoning Dataset

The main text discusses the performance gains resulting from applying spectral modulation with different reduction factors to various weight matrices of the model on the FEVER dataset. Here, we investigate this phenomena on the BigBench-Epistemic Reasoning dataset, with results presented in Fig.5. We can derive two findings from this. First, the impact of spectral modulation on different weight matrices may vary across different tasks. For example, applying spectral modulation to the W_{out} matrix in layer 30 leads to positive performance gains on the FEVER dataset but results in negative effects on the BigBench-Epistemic Reasoning dataset. Second, appropriate spectral modulation can enhance the model's performance on the BigBench-Epistemic Reasoning dataset, indicating that spectral modulation is a promising method for improving model performance. To simplify the search space for Bayesian optimization, this paper applies joint spectral modulation only to the seven layers preceding the last layer of the model.

D Joint Spectral Modulation with a Conservative Strategy

We conducted an experiment on the Vicuna-7B-V1.5 model to evaluate the performance of spectral modulation, applying joint spectral modulation across ten datasets with α set to 64 and ρ set to 0.95. Such an operation means that we uniformly modulated the high-frequency components conservatively. The results are shown in the Tab. 6, and this conservative modulation method achieves promising average performance.

E Lexical Composition Analysis on FEVER

In the FEVER dataset, we randomly selected 1,000 samples and analyzed the top three tokens generated by the vanilla Vicuna-7B-V1.5, our spectrally modulated model, and the high-frequency counterpart. Tab. 5 presents the frequency of the ten most frequently occurring tokens among these top three tokens. The answer set for the FEVER dataset is [true, false]. It should be noted that 'true' and 'false' in the answers do not indicate the correctness or incorrectness of the prediction. For example, if the predicted token is 'true' but the ground-truth label is 'false,' this prediction is incorrect.

We find that in the context of the prompted questions, the conjunction 'that', the pronoun 'neither', the adverbs 'not', the determiner 'a', and the punctuation marks '::' and '...:' appear significantly more frequently in answers generated by the high-frequency model than in answers generated by our spectrally modulated model. These elements convey relatively little information and serve more as supplements and embellishments. If we use both the high-frequency and low-frequency components of the model for prediction, the output generated by the high-frequency components contains less information and may interfere with the accurate predictions from the low-frequency components. This explains why the performance of the vanilla model is inferior to that of the spectrally modulated model, which predominantly relies on low-frequency components.

Top-10	Ours		High		Vanilla	
1	false	970	not	998	true	1000
2	not	876	::	759	false	996
3	true	700	actually	294	not	735
4	::	209	a	287	that	219
5	that	208	that	259	::	27
6	based	4	neither	200	likely	18
7	a	2	false	70	based	2
8	subject	2	...:	60	a	1
9	likely	2	in	15	set	1
10	directed	1	likely	12	partially	1

Table 5: Top ten word frequency statistics for the top three tokens with predicted probabilities. We consider 1000 samples in the FEVER dataset. The numbers on the left indicate the rankings. "Vanilla", "Ours", "High" refer to answers derived from the vanilla Vicuna-7B-V1.5, our spectrally modulated model, and the high-frequency model, respectively.

F Related Literature to Applications of Frequency Domain Analysis

Frequency domain analysis occupies a critical position in the field of signal processing. This technique facilitates the elucidation of the frequency composition of signals, the identification of their intrinsic characteristics, and the analysis of noise components within these signals. Taking image signal processing as an example, frequency domain analysis has many typical applications, including image compression (Pennebaker and Mitchell, 1992; Wong and Noyes, 1994; Wen et al., 2022) and image denoising (Wang et al., 2006; Yue et al., 2014; Hasinoff et al., 2016; Li et al., 2022; Erkan et al., 2020; Sheng et al., 2022), among others.

As for the image compression methods, the JPEG compression standard (Pennebaker and Mitchell, 1992) performs discrete cosine transformation (DCT) and quantization operations on patches of the image. Wong and Noyes (1994) propose a space-frequency partition scheme to fully leverage the exceptional localization properties of wavelets in both spatial and frequency domains, resulting in enhanced PSNR compared to traditional subband and wavelet-based methods. Wen et al. (2022) propose a method for high-quality image encryption and restoration based on DCT frequency-domain compression coding and chaos, aiming to enhance the information effectiveness during digital image transmission.

As for the denoising methods, Wang et al. (2006) introduce an approximate measure about the similarity of neighborhood windows to perform non-local denoising, and use an efficient Summed Square Image scheme and Fast Fourier Transform (FFT) to accelerate the calculation of this measure. Yue et al. (2014) put forward a method that perform denoising strategies in both the frequency and spatial domains, subsequently fusing the denoised results in the frequency domain for a final results, which significantly improves the visual quality of images. Hasinoff et al. (2016) and Li et al. (2022) propose to utilize Wiener filtering in the frequency domain to mitigate noise in low-light images and enhance imaging quality. Erkan et al. (2020) introduce an adaptive frequency median filter, a filter that prioritizes the uniqueness of gray values, effectively eliminating salt-and-pepper noise. Sheng et al. (2022) perform the decomposition of frequencies in their denoising model, encouraging separate treatment of low-frequency and high-frequency content. This approach ensures spatial smoothness while preserving high-frequency structures, thereby enabling the recovery of clear images with prominent structures from noisy inputs.

In recent years, there has been a growing emergence of applications integrating frequency analysis into deep learning. Xu et al. (2020) introduce a learning-based frequency selection method aimed at identifying trivial frequency components that could be eliminated without sacrificing accuracy, which can enhance the precision of image classification. Qin et al. (2021) extend the concept of channel attention to the frequency domain, proposing the multi-spectral channel attention framework. This framework exhibited promising performance across tasks such as image classification, object detection, and instance segmentation. Zhu et al. (2023) introduce random dropout in the frequency domain as an approach of data augmentation, thereby enhancing the robustness of trained models. Patro and Agneeswaran (2024) propose a Scattering Vision Transformer to more effectively capture fine-grained information within the input. The model first uses Fourier transform operations to implement a frequency-domain layer for feature extraction in the shallow layers of the network, and subsequently employs multi-head self-attention modules in the deeper layers for feature modeling. Chen et al. (2024) propose the Frequency Domain Kernelization method, which uses the Discrete Cosine Transform (DCT) to map the Transformer's Query and Key to the frequency domain. This approach effectively transforms the complexity of attention computation to linear time, thereby reducing training costs and improving inference speed.

G Visual Examples Correctly Classified by Our Method

In the main text, we numerically demonstrate that spectral modulation can effectively improve the model's performance in visual tasks by correctly classifying more samples. In this section, we visualize some images that were misclassified by the original model but correctly classified by the spectrally modulated model, as shown in Fig. 6. We selected the CLIP version of ResNet-50 (Radford et al., 2021) for our experiments. Notably, this version of ResNet-50 includes a Transformer-style attention pooling layer at the end. Therefore, we applied spectral modulation to this layer, setting the reduction factor to 0.95 and the protection threshold to 25. The images are presented with their correct labels and the incorrect predictions made by the original model in the format (label | Prediction). These results illustrate the potential of the spectral modulation method to generalize to non-linguistic tasks. Further exploration of how to fully leverage its effectiveness across different tasks is left for future work.

H Visualization of Weight Matrices and Spectrally Modulated Weight Matrices

In the main text, we discussed how to perform spectral modulation on weight matrices and experimentally validated its superior performance. In this section, we visually present the impact of spectral modulation on the model's weight matrices, with the results shown in Fig. 7. For this experiment, the reduction factor is set to 0.1. The spectra of W_{in} and W_{out} are uniformly distributed from high to low frequencies, with similar intensity across different frequency bands. After spectral modulation, the high-frequency information is significantly reduced.

I Discussion on Model Interpretability

In this subsection, we analyze the interpretability of the model based on Discretized Integrated Gradients (DIG) (Sanyal and Ren, 2021), a typical attribution-based explanation method. DIG effectively identifies the significance of each word in the input sample during the model's processing for a specific answer. For both the vanilla model and the spectrally modulated model, we normalize the saliency of each word with respect to the correct and incorrect answers, respectively, and then visualize the differences between these normalized values to observe the impact of each word on the model's prediction results.

In Fig.8 (a), we observe that our spectrally modulated model pays more attention to key information 'Sean Penn' and 'stage actor'. In contrast, although the vanilla model also focuses on these parts, it excessively pays attention to less informative words like 'is,' 'ever,' and 'a.' These words should not have a significant influence on the model's decisions.

In Fig.8 (b), the key information includes 'Sancho Panza', 'character', and 'Don Quixote'. It is observed that although the vanilla model pays attention to these pieces of information, it disproportionately focuses on the non-critical word 'in'. In contrast, the spectrally modulated model allocates most of its attention to the key information.

In Fig.8 (c), the key information consists of 'capital' and 'Mogadishu'. The original model overly focuses on 'capital', resulting in insufficient attention to other key information. Although the spectrally modulated model slightly underemphasizes 'capital', it distributes its attention more evenly across the other key information compared to the vanilla model.

This phenomenon indicates that the spectrally modulated model not only demonstrates better performance but also possesses good interpretability.

J Bayesian Optimization Recommended Hyperparameters

Our proposed joint spectral modulation refines the weight matrices in the feed-forward modules of the seven layers preceding the final layer in LLMs. The optimal hyperparameters are determined using Bayesian optimization, with the total iteration time varying across different datasets and models. For example, on the FEVER dataset using the Vicuna-7B-V1.5 model, the accuracy is calculated as classification accuracy, with labels being a single token in length. The sample size during the search was 2617, and it took 23 GPU hours for 150 search steps on the RTX 4090 GPU. The search time mainly depends on the inference model, the number of tokens generated, the sample size, and the number of search steps. If the accuracy for the FEVER dataset were calculated as generation accuracy with "max_new_tokens" set to 10, the search time would be approximately 10 times longer than that required for classification accuracy. In fact, the optimal results frequently appeared within the first 50 steps. To reduce computational cost, 50 steps can still yield satisfactory results. In our experiments, we set the number of iterations to 150, and the recommended hyperparameters are listed in Tab. 7, 8, and 9. These hyperparameters can be used to reproduce our experimental results. Naturally, additional iterations and improved selection schemes for the weight matrices could yield hyperparameters that achieve even better performance, but that is not the focus of this paper.

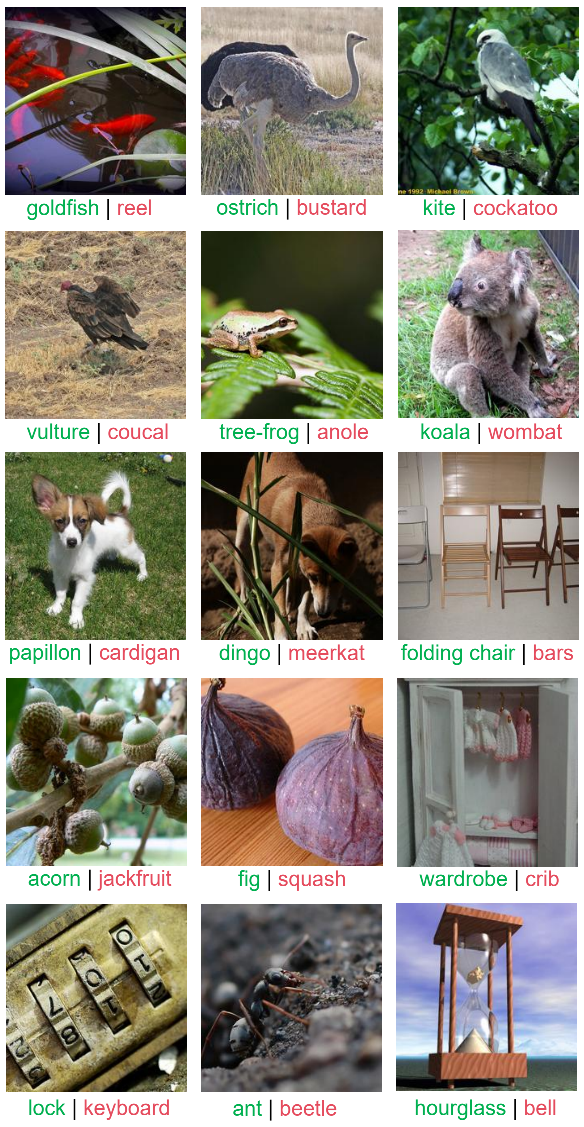

Figure 6: Examples correctly classified by the spectrally modulated CLIP model with a ResNet-50 backbone, but misclassified by the original CLIP model.

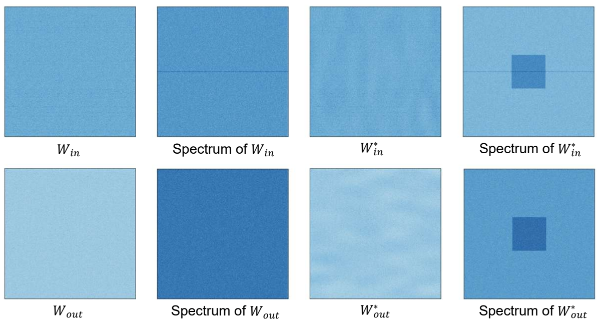

Figure 7: Taking the weight matrices in the feed-forward module of the penultimate layer of the Vicuna-7B-V1.5 model as an example, we visualize their spatial domain and spectral domain before and after spectral modulation, with darker colors indicating higher intensity. For visualization purposes, we have truncated a central portion of the weight matrix and the spectrum.

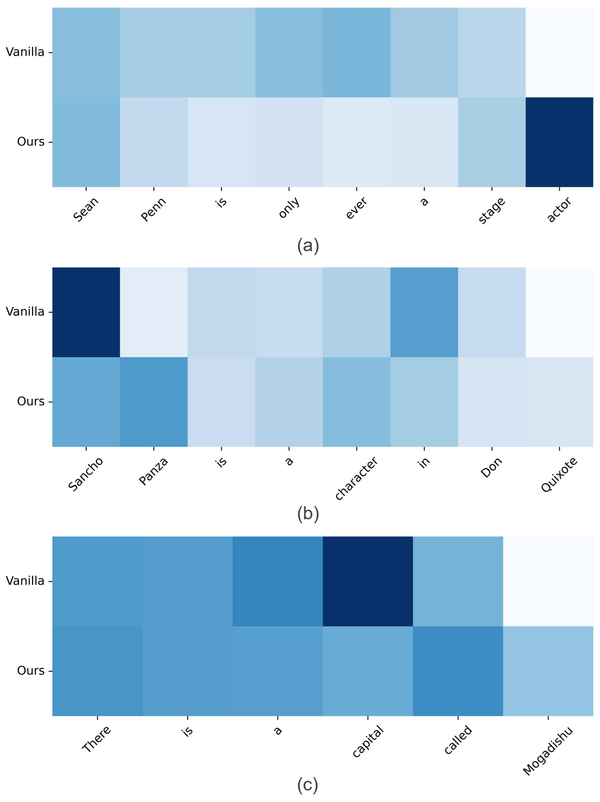

Figure 8: The attribution of each word in different input samples on the vanilla model and our spectrally modulated model on the FEVER dataset, with darker colors indicating greater impact. The model used is Vicuna-7B-V1.5.

Dataset	Model Name		
	Vicuna-7B-V1.5	Vicuna-7B-V1.5*(α=64,ρ=0.95)	Vicuna-7B-V1.5*(Bayesian optimisation)
CounterFact	34.91	**35.63**	37.53
FEVER	53.33	**56.74**	71.68
Bios Gender	99.21	**99.22**	99.26
Bios Profession	**81.97**	81.91	83.68
TruthfulQA	44.56	**44.84**	56.12
BigBench-Epistemic Reasoning	**50.25**	50.00	62.88
BigBench-WikidataQA	59.91	**61.03**	63.06
Stanford Sentiment Treebank 2	62.53	**66.91**	70.06
Race	57.75	**61.28**	76.59
Question Natural Language Inference	64.19	**64.94**	67.63
Average	60.86	**62.25**	68.85

Table 6: Effect of uniform joint spectral modulation on ten datasets. We tested the original model on the validation set, as well as the model modified by uniform joint spectral modulation and the model modulated with the best parameters obtained through Bayesian optimization. We report their performance and compare the performance of the vanilla model with that of the model modified by uniform joint spectral modulation, with better results highlighted in bold.

Dataset	Hyperparameters						
CounterFact	[W_{in}, 20, 0.78]	[W_{in}, 21, 0.70]	[W_{in}, 22, 0.85]	[W_{in}, 23, 0.26]	[W_{in}, 24, 0.25]	[W_{in}, 25, 0.04]	[W_{in}, 26, 0.35]
	[W_{out}, 20, 0.99]	[W_{out}, 21, 0.00]	[W_{out}, 22, 0.00]	[W_{out}, 23, 0.75]	[W_{out}, 24, 0.28]	[W_{out}, 25, 0.28]	[W_{out}, 26, 1.00]
FEVER	[W_{in}, 20, 0.06]	[W_{in}, 21, 0.94]	[W_{in}, 22, 0.72]	[W_{in}, 23, 0.37]	[W_{in}, 24, 0.50]	[W_{in}, 25, 0.95]	[W_{in}, 26, 0.88]
	[W_{out}, 20, 0.23]	[W_{out}, 21, 0.87]	[W_{out}, 22, 0.98]	[W_{out}, 23, 0.88]	[W_{out}, 24, 0.12]	[W_{out}, 25, 0.86]	[W_{out}, 26, 0.98]
Bios Gender	[W_{in}, 20, 0.92]	[W_{in}, 21, 0.52]	[W_{in}, 22, 1.00]	[W_{in}, 23, 0.87]	[W_{in}, 24, 0.86]	[W_{in}, 25, 0.36]	[W_{in}, 26, 0.75]
	[W_{out}, 20, 0.65]	[W_{out}, 21, 0.04]	[W_{out}, 22, 0.21]	[W_{out}, 23, 0.86]	[W_{out}, 24, 1.00]	[W_{out}, 25, 0.24]	[W_{out}, 26, 0.98]
Bios Profession	[W_{in}, 20, 0.27]	[W_{in}, 21, 0.11]	[W_{in}, 22, 0.35]	[W_{in}, 23, 0.51]	[W_{in}, 24, 0.29]	[W_{in}, 25, 0.43]	[W_{in}, 26, 0.81]
	[W_{out}, 20, 0.82]	[W_{out}, 21, 0.42]	[W_{out}, 22, 0.65]	[W_{out}, 23, 0.95]	[W_{out}, 24, 0.09]	[W_{out}, 25, 0.83]	[W_{out}, 26, 0.93]
TruthfulQA	[W_{in}, 20, 0.95]	[W_{in}, 21, 0.98]	[W_{in}, 22, 0.11]	[W_{in}, 23, 0.56]	[W_{in}, 24, 0.81]	[W_{in}, 25, 0.27]	[W_{in}, 26, 0.76]
	[W_{out}, 20, 0.86]	[W_{out}, 21, 0.92]	[W_{out}, 22, 0.04]	[W_{out}, 23, 0.36]	[W_{out}, 24, 0.16]	[W_{out}, 25, 0.83]	[W_{out}, 26, 0.78]
BigBench-Epistemic Reasoning	[W_{in}, 20, 0.98]	[W_{in}, 21, 0.24]	[W_{in}, 22, 0.60]	[W_{in}, 23, 0.76]	[W_{in}, 24, 0.94]	[W_{in}, 25, 1.00]	[W_{in}, 26, 0.44]
	[W_{out}, 20, 0.66]	[W_{out}, 21, 0.76]	[W_{out}, 22, 0.16]	[W_{out}, 23, 0.11]	[W_{out}, 24, 0.21]	[W_{out}, 25, 0.02]	[W_{out}, 26, 0.16]
BigBench-WikidataQA	[W_{in}, 20, 0.59]	[W_{in}, 21, 0.54]	[W_{in}, 22, 0.31]	[W_{in}, 23, 0.98]	[W_{in}, 24, 0.86]	[W_{in}, 25, 0.56]	[W_{in}, 26, 0.40]
	[W_{out}, 20, 0.83]	[W_{out}, 21, 0.32]	[W_{out}, 22, 0.17]	[W_{out}, 23, 0.82]	[W_{out}, 24, 0.88]	[W_{out}, 25, 0.93]	[W_{out}, 26, 1.00]
Stanford Sentiment Treebank 2	[W_{in}, 20, 0.03]	[W_{in}, 21, 0.30]	[W_{in}, 22, 0.32]	[W_{in}, 23, 0.51]	[W_{in}, 24, 0.95]	[W_{in}, 25, 0.63]	[W_{in}, 26, 0.95]
	[W_{out}, 20, 0.77]	[W_{out}, 21, 0.99]	[W_{out}, 22, 0.52]	[W_{out}, 23, 0.96]	[W_{out}, 24, 0.93]	[W_{out}, 25, 0.39]	[W_{out}, 26, 1.00]
Race	[W_{in}, 20, 0.29]	[W_{in}, 21, 0.79]	[W_{in}, 22, 0.31]	[W_{in}, 23, 0.50]	[W_{in}, 24, 0.11]	[W_{in}, 25, 0.10]	[W_{in}, 26, 0.88]
	[W_{out}, 20, 0.85]	[W_{out}, 21, 0.99]	[W_{out}, 22, 0.86]	[W_{out}, 23, 0.89]	[W_{out}, 24, 0.62]	[W_{out}, 25, 0.01]	[W_{out}, 26, 0.05]
Question Natural Language Inference	[W_{in}, 20, 0.59]	[W_{in}, 21, 0.04]	[W_{in}, 22, 0.60]	[W_{in}, 23, 0.13]	[W_{in}, 24, 0.88]	[W_{in}, 25, 0.88]	[W_{in}, 26, 0.26]
	[W_{out}, 20, 0.12]	[W_{out}, 21, 0.22]	[W_{out}, 22, 0.70]	[W_{out}, 23, 0.22]	[W_{out}, 24, 0.69]	[W_{out}, 25, 0.14]	[W_{out}, 26, 0.41]

Table 7: Recommended hyperparameters obtained using Bayesian search in the GPT-J-6B model. We illustrate the hyperparameters in the format [τ, ℓ, ρ], representing the weight matrix, the number of layers, and the reduction factor. The protection threshold is set to 64.

Dataset	Hyperparameters						
Counterfact	[W_{in}, 24, 0.89]	[W_{in}, 25, 0.42]	[W_{in}, 26, 0.71]	[W_{in}, 27, 0.42]	[W_{in}, 28, 0.58]	[W_{in}, 29, 0.59]	[W_{in}, 30, 0.44]
	[W_{out}, 24, 0.06]	[W_{out}, 25, 0.34]	[W_{out}, 26, 0.97]	[W_{out}, 27, 0.86]	[W_{out}, 28, 0.35]	[W_{out}, 29, 0.74]	[W_{out}, 30, 0.68]
FEVER	[W_{in}, 24, 0.94]	[W_{in}, 25, 0.68]	[W_{in}, 26, 1.00]	[W_{in}, 27, 0.91]	[W_{in}, 28, 0.69]	[W_{in}, 29, 0.19]	[W_{in}, 30, 0.89]
	[W_{out}, 24, 0.78]	[W_{out}, 25, 0.84]	[W_{out}, 26, 0.83]	[W_{out}, 27, 0.89]	[W_{out}, 28, 0.92]	[W_{out}, 29, 0.69]	[W_{out}, 30, 0.30]
Bios Gender	[W_{in}, 24, 0.98]	[W_{in}, 25, 0.78]	[W_{in}, 26, 0.18]	[W_{in}, 27, 0.56]	[W_{in}, 28, 0.69]	[W_{in}, 29, 0.32]	[W_{in}, 30, 0.71]
	[W_{out}, 24, 0.72]	[W_{out}, 25, 0.74]	[W_{out}, 26, 0.44]	[W_{out}, 27, 0.35]	[W_{out}, 28, 0.09]	[W_{out}, 29, 0.98]	[W_{out}, 30, 0.96]
Bios Profession	[W_{in}, 24, 0.74]	[W_{in}, 25, 0.81]	[W_{in}, 26, 0.91]	[W_{in}, 27, 0.75]	[W_{in}, 28, 0.81]	[W_{in}, 29, 0.11]	[W_{in}, 30, 0.06]
	[W_{out}, 24, 1.00]	[W_{out}, 25, 0.94]	[W_{out}, 26, 0.20]	[W_{out}, 27, 0.07]	[W_{out}, 28, 0.09]	[W_{out}, 29, 0.37]	[W_{out}, 30, 0.53]
TruthfulQA	[W_{in}, 24, 0.91]	[W_{in}, 25, 0.54]	[W_{in}, 26, 0.33]	[W_{in}, 27, 0.33]	[W_{in}, 28, 0.06]	[W_{in}, 29, 0.68]	[W_{in}, 30, 0.35]
	[W_{out}, 24, 0.48]	[W_{out}, 25, 0.71]	[W_{out}, 26, 0.21]	[W_{out}, 27, 0.07]	[W_{out}, 28, 0.52]	[W_{out}, 29, 0.76]	[W_{out}, 30, 0.68]
BigBench-Epistemic Reasoning	[W_{in}, 24, 0.80]	[W_{in}, 25, 0.16]	[W_{in}, 26, 0.64]	[W_{in}, 27, 0.68]	[W_{in}, 28, 0.33]	[W_{in}, 29, 0.31]	[W_{in}, 30, 0.96]
	[W_{out}, 24, 0.42]	[W_{out}, 25, 0.92]	[W_{out}, 26, 0.82]	[W_{out}, 27, 0.13]	[W_{out}, 28, 0.47]	[W_{out}, 29, 0.70]	[W_{out}, 30, 0.99]
BigBench-WikidataQA	[W_{in}, 24, 0.48]	[W_{in}, 25, 0.94]	[W_{in}, 26, 0.08]	[W_{in}, 27, 0.63]	[W_{in}, 28, 0.28]	[W_{in}, 29, 0.70]	[W_{in}, 30, 0.68]
	[W_{out}, 24, 0.93]	[W_{out}, 25, 0.45]	[W_{out}, 26, 0.28]	[W_{out}, 27, 0.86]	[W_{out}, 28, 0.14]	[W_{out}, 29, 0.79]	[W_{out}, 30, 0.94]
Stanford Sentiment Treebank 2	[W_{in}, 24, 0.96]	[W_{in}, 25, 0.97]	[W_{in}, 26, 0.53]	[W_{in}, 27, 0.43]	[W_{in}, 28, 0.15]	[W_{in}, 29, 0.22]	[W_{in}, 30, 0.08]
	[W_{out}, 24, 0.95]	[W_{out}, 25, 0.99]	[W_{out}, 26, 0.13]	[W_{out}, 27, 0.65]	[W_{out}, 28, 0.41]	[W_{out}, 29, 0.44]	[W_{out}, 30, 0.96]
Race	[W_{in}, 24, 0.47]	[W_{in}, 25, 0.57]	[W_{in}, 26, 0.94]	[W_{in}, 27, 0.29]	[W_{in}, 28, 0.95]	[W_{in}, 29, 0.68]	[W_{in}, 30, 0.51]
	[W_{out}, 24, 0.76]	[W_{out}, 25, 0.71]	[W_{out}, 26, 0.87]	[W_{out}, 27, 0.08]	[W_{out}, 28, 0.95]	[W_{out}, 29, 0.61]	[W_{out}, 30, 0.53]
Question Natural Language Inference	[W_{in}, 24, 0.22]	[W_{in}, 25, 0.56]	[W_{in}, 26, 0.30]	[W_{in}, 27, 0.50]	[W_{in}, 28, 0.95]	[W_{in}, 29, 0.59]	[W_{in}, 30, 0.67]
	[W_{out}, 24, 0.70]	[W_{out}, 25, 0.01]	[W_{out}, 26, 0.59]	[W_{out}, 27, 0.91]	[W_{out}, 28, 0.63]	[W_{out}, 29, 0.46]	[W_{out}, 30, 0.86]

Table 8: Recommended hyperparameters obtained using Bayesian search in the Vicuna-7B-V1.5 model. We illustrate the hyperparameters in the format [τ, ℓ, ρ], representing the weight matrix, the number of layers, and the reduction factor. The protection threshold is set to 64.

Dataset	Hyperparameters						
CounterFact	[W_{in}, 24, 0.51]	[W_{in}, 25, 0.52]	[W_{in}, 26, 0.40]	[W_{in}, 27, 0.44]	[W_{in}, 28, 0.45]	[W_{in}, 29, 0.76]	[W_{in}, 30, 0.53]
	[W_{out}, 24, 0.75]	[W_{out}, 25, 0.57]	[W_{out}, 26, 0.39]	[W_{out}, 27, 0.61]	[W_{out}, 28, 0.48]	[W_{out}, 29, 0.83]	[W_{out}, 30, 0.96]
FEVER	[W_{in}, 24, 0.76]	[W_{in}, 25, 0.42]	[W_{in}, 26, 0.44]	[W_{in}, 27, 0.95]	[W_{in}, 28, 0.43]	[W_{in}, 29, 0.95]	[W_{in}, 30, 0.56]
	[W_{out}, 24, 0.75]	[W_{out}, 25, 0.29]	[W_{out}, 26, 0.90]	[W_{out}, 27, 0.09]	[W_{out}, 28, 0.51]	[W_{out}, 29, 0.91]	[W_{out}, 30, 0.89]
Bios Gender	[W_{in}, 24, 0.76]	[W_{in}, 25, 0.93]	[W_{in}, 26, 0.16]	[W_{in}, 27, 0.57]	[W_{in}, 28, 0.42]	[W_{in}, 29, 0.57]	[W_{in}, 30, 0.65]
	[W_{out}, 24, 0.86]	[W_{out}, 25, 0.81]	[W_{out}, 26, 0.88]	[W_{out}, 27, 0.80]	[W_{out}, 28, 0.87]	[W_{out}, 29, 0.69]	[W_{out}, 30, 0.48]
Bios Profession	[W_{in}, 24, 0.91]	[W_{in}, 25, 0.36]	[W_{in}, 26, 0.84]	[W_{in}, 27, 0.65]	[W_{in}, 28, 0.64]	[W_{in}, 29, 0.54]	[W_{in}, 30, 0.99]
	[W_{out}, 24, 0.88]	[W_{out}, 25, 0.13]	[W_{out}, 26, 0.61]	[W_{out}, 27, 0.35]	[W_{out}, 28, 0.91]	[W_{out}, 29, 0.89]	[W_{out}, 30, 0.78]
TruthfulQA	[W_{in}, 24, 0.65]	[W_{in}, 25, 0.23]	[W_{in}, 26, 0.99]	[W_{in}, 27, 0.92]	[W_{in}, 28, 0.93]	[W_{in}, 29, 0.37]	[W_{in}, 30, 0.00]
	[W_{out}, 24, 0.90]	[W_{out}, 25, 0.24]	[W_{out}, 26, 0.61]	[W_{out}, 27, 0.59]	[W_{out}, 28, 0.72]	[W_{out}, 29, 0.61]	[W_{out}, 30, 0.98]
BigBench-Epistemic Reasoning	[W_{in}, 24, 0.67]	[W_{in}, 25, 0.06]	[W_{in}, 26, 0.39]	[W_{in}, 27, 0.40]	[W_{in}, 28, 0.41]	[W_{in}, 29, 0.21]	[W_{in}, 30, 0.84]
	[W_{out}, 24, 0.69]	[W_{out}, 25, 0.53]	[W_{out}, 26, 0.95]	[W_{out}, 27, 0.35]	[W_{out}, 28, 0.07]	[W_{out}, 29, 0.73]	[W_{out}, 30, 0.73]
BigBench-WikidataQA	[W_{in}, 24, 0.28]	[W_{in}, 25, 0.42]	[W_{in}, 26, 0.37]	[W_{in}, 27, 0.92]	[W_{in}, 28, 0.87]	[W_{in}, 29, 0.33]	[W_{in}, 30, 0.68]
	[W_{out}, 24, 0.94]	[W_{out}, 25, 0.42]	[W_{out}, 26, 0.81]	[W_{out}, 27, 0.72]	[W_{out}, 28, 0.72]	[W_{out}, 29, 0.89]	[W_{out}, 30, 0.94]
Stanford Sentiment Treebank 2	[W_{in}, 24, 0.76]	[W_{in}, 25, 0.79]	[W_{in}, 26, 0.61]	[W_{in}, 27, 0.96]	[W_{in}, 28, 0.15]	[W_{in}, 29, 0.68]	[W_{in}, 30, 0.40]
	[W_{out}, 24, 0.88]	[W_{out}, 25, 0.57]	[W_{out}, 26, 0.93]	[W_{out}, 27, 0.36]	[W_{out}, 28, 0.34]	[W_{out}, 29, 0.86]	[W_{out}, 30, 0.99]
Race	[W_{in}, 24, 0.38]	[W_{in}, 25, 0.15]	[W_{in}, 26, 0.44]	[W_{in}, 27, 0.00]	[W_{in}, 28, 0.70]	[W_{in}, 29, 0.96]	[W_{in}, 30, 0.60]
	[W_{out}, 24, 0.43]	[W_{out}, 25, 0.01]	[W_{out}, 26, 0.64]	[W_{out}, 27, 0.30]	[W_{out}, 28, 0.78]	[W_{out}, 29, 0.92]	[W_{out}, 30, 0.94]
Question Natural Language Inference	[W_{in}, 24, 0.08]	[W_{in}, 25, 0.38]	[W_{in}, 26, 0.40]	[W_{in}, 27, 0.99]	[W_{in}, 28, 0.00]	[W_{in}, 29, 0.01]	[W_{in}, 30, 0.75]
	[W_{out}, 24, 0.76]	[W_{out}, 25, 0.67]	[W_{out}, 26, 0.71]	[W_{out}, 27, 0.45]	[W_{out}, 28, 0.83]	[W_{out}, 29, 0.51]	[W_{out}, 30, 0.93]

Table 9: Recommended hyperparameters obtained using Bayesian search in the Llama3-8B model. We illustrate the hyperparameters in the format [τ, ℓ, ρ], representing the weight matrix, the number of layers, and the reduction factor. The protection threshold is set to 64.