

# Fourier-VLM: Compressing Vision Tokens in the Frequency Domain for Large Vision-Language Models

Huanyu Wang<sup>1</sup> Jushi Kai<sup>1</sup> Haoli Bai<sup>2</sup> Lu Hou<sup>2</sup> Bo Jiang<sup>1</sup> Ziwei He<sup>†3</sup> Zhouhan Lin<sup>†13</sup>

## Abstract

Vision-Language Models (VLMs) typically replace the predefined image placeholder token (`<image>`) in textual instructions with visual features from an image encoder, forming the input to a backbone Large Language Model (LLM). However, the large number of vision tokens significantly increases the context length, leading to high computational overhead and inference latency. While previous efforts mitigate this by selecting only important visual features or leveraging learnable queries to reduce token count, they often compromise performance or introduce substantial extra costs. In response, we propose **Fourier-VLM**, a simple yet efficient method that compresses visual representations in the frequency domain. Our approach is motivated by the observation that vision features output from the vision encoder exhibit concentrated energy in low-frequency components. Leveraging this, we apply a low-pass filter to the vision features using a two-dimensional Discrete Cosine Transform (DCT). Notably, the DCT is efficiently computed via the Fast Fourier Transform (FFT) operator with a time complexity of  $\mathcal{O}(n \log n)$ , minimizing the extra computational cost while introducing no additional parameters. Extensive experiments across various image-based benchmarks demonstrate that Fourier-VLM achieves competitive performance with strong generalizability across both LLaVA and Qwen-VL architectures. Crucially, it reduces inference FLOPs by up to 83.8% and boosts generation speed by 31.2% compared to LLaVA-v1.5, highlighting the superior efficiency and practicality.

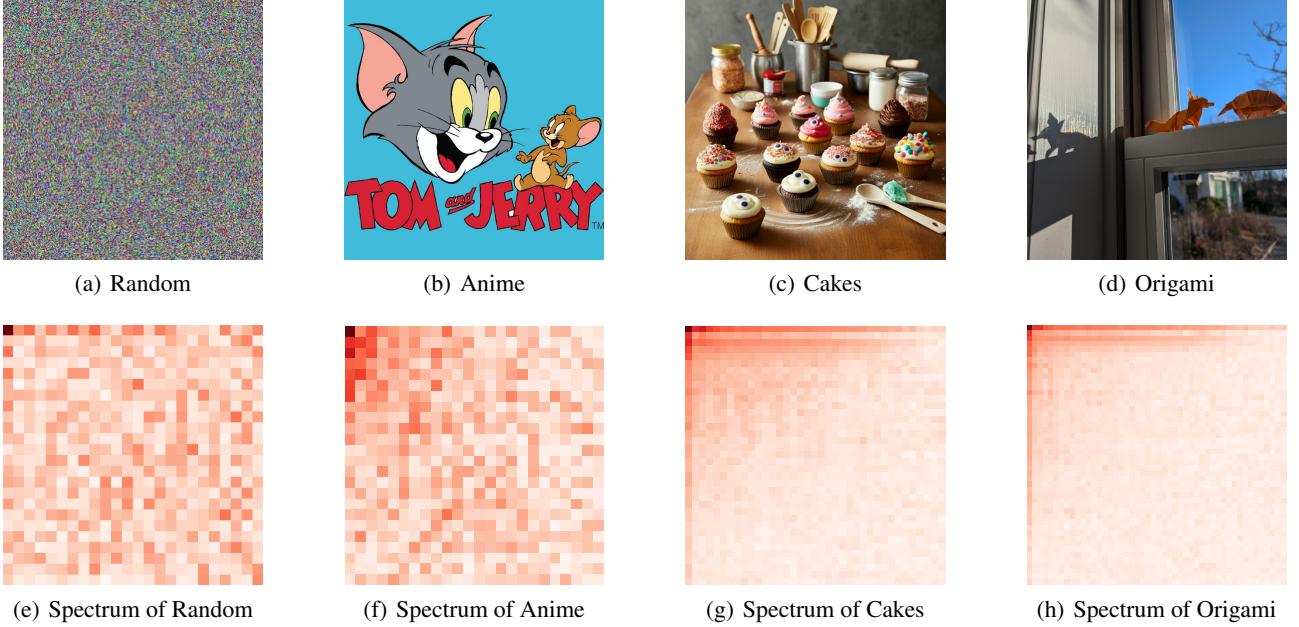
## 1. Introduction

Vision-Language Models (VLMs) extend the Large Language Models (LLMs) with visual understanding capabilities by attaching a vision encoder through a “glue layer”, such as a Multi-Layer Perceptron (MLP) or Q-Former (Li et al., 2023a). By leveraging the reasoning and processing capabilities of LLMs, VLMs demonstrate impressive performance in visual understanding tasks. However, VLMs require substantial computational resources due to the large number of vision tokens generated by the vision encoder. When VLMs replace the image placeholder with these vision tokens in the language instructions, the backbone LLMs are faced with an extremely long context, resulting in significant computational overhead and high inference latency. The challenge becomes even more pronounced when processing high-resolution images, multiple images, and videos.

Fortunately, it has been observed that visual representations in VLMs exhibit considerable redundancy, often surpassing that of natural language, which motivates the development of vision token compression techniques. Some approaches utilize learnable queries to extract visual features (e.g., QueCC (Li et al., 2024b), MQT-LLaVA (Hu et al., 2024)), while others select important vision tokens based on predefined rules (e.g., ATP-LLaVA (Ye et al., 2024)). Another line of work seeks to merge substantial vision tokens into a smaller set (e.g., LLaVA-PruMerge (Shang et al., 2024), VisToG (Huang et al., 2024)). However, these methods often introduce additional computational costs due to the complex compression modules while failing to maintain satisfactory performance, with limited generalizability across diverse VLM architectures.

To address these limitations, we aim to develop a more efficient and generalizable vision token compression strategy. One promising direction lies in the frequency domain. Specifically, we apply a two-dimensional Discrete Cosine Transform (DCT) to the image features output from the vision encoder. As shown in Figure 1, energy tends to concentrate in low-frequency components across all hidden dimensions. However, this pattern is barely noticeable in an RGB image with randomly generated pixel values, and varies in prominence across images of different types,

<sup>†</sup>Corresponding author <sup>1</sup>Shanghai Jiao Tong University, Shanghai, China <sup>2</sup>Noah’s Ark Lab, Huawei Technologies Ltd., Shanghai, China <sup>3</sup>Shanghai Innovation Institute, Shanghai, China. Correspondence to: Ziwei He <[ziweihe@outlook.com](mailto:ziweihe@outlook.com)>, Zhouhan Lin <[lin.zhouhan@gmail.com](mailto:lin.zhouhan@gmail.com)>.



*Figure 1.* Heatmap visualization of the frequency spectra computed from vision encoder outputs of different images. Since only the magnitude is of interest, the absolute values across all hidden dimensions are averaged for each frequency component and then plotted on a logarithmic scale. (e) and (f) show the frequency spectra of the visual encoder outputs from LLaVA-v1.5, while (g) and (h) correspond to those from Qwen-2-VL.

indicating that the energy distribution inherently captures certain structural or semantic information. Moreover, this low-frequency energy concentration emerges consistently in both LLaVA-v1.5 and Qwen-2-VL vision encoders, hinting at its potential generality across different VLM architectures. The observed energy dominance implies semantic redundancy in high-frequency components of visual features, allowing for effective frequency truncation with minimal loss of semantic content.

Motivated by these observations, we introduce Fourier-VLM, a simple and effective method to compress vision tokens for VLMs. Our key contributions include:

- We observe an energy concentration in low-frequency components of visual representations, and design the Frequency Feature Compressor (FFC), a parameter-free and highly efficient module for vision token compression within the frequency domain.
- We apply our method to both LLaVA-v1.5 and Qwen-VL series, and evaluate across diverse image-based benchmarks, demonstrating its strong performance and generalizability.
- We empirically evaluate latency, FLOPs, and KV cache usage of our method, emphasizing its superior efficiency.

## 2. Related Work

### 2.1. Vision Token Compression in VLMs

Vision-Language Models (VLMs) have achieved remarkable progress in image and video understanding by integrating LLMs with visual encoders. Despite strong performance from models like LLaVA series (Liu et al., 2023; 2024a) and Qwen-VL series (Wang et al., 2024; Bai et al., 2025), the high volume of vision tokens remains a major bottleneck for efficient inference and practical deployment.

To address the challenge of long contexts in VLMs, previous research has primarily focused on reducing the number of vision tokens. One common approach involves selecting the most relevant vision tokens or merging less important ones. For example, ATP-LLaVA (Ye et al., 2024) computes an importance score for each vision token and dynamically determines a pruning threshold to remove redundant tokens within the backbone LLM, while LLaVA-PruMerge (Shang et al., 2024) clusters and averages vision tokens according to similarity between the class token and spatial tokens. Other methods leverage query transformers to extract visual features. MQT-LLaVA (Hu et al., 2024) employs a Matryoshka Query Transformer, whereas QueCC (Li et al., 2024b) introduces a query-based convolutional cross-attention module that enables text embeddings to query vision tokens. Furthermore, recent research has also explored transferring visual information to language tokens, such as

LLaVA-mini (Zhang et al., 2025), which employs a pre-fusion module that allows text tokens to integrate relevant visual information in advance. However, these approaches have yet to find an optimal balance between the additional computational cost of compression and performance degradation. This challenge motivates us to further investigate cost-efficient token compression methods.

## 2.2. Frequency-based Compression

Frequency domain techniques have long played an important role in signal processing and data compression. In computer vision, frequency transformations are foundational to standard image compression algorithms, such as JPEG. Previous study (Xu et al., 2020) has also shown that convolutional neural networks (CNNs) exhibit a strong sensitivity to low-frequency channels, revealing an inherent frequency bias in visual feature extraction. Beyond the vision domain, frequency-based approaches have also been widely explored in natural language processing. For instance, FNet (Lee-Thorp et al., 2022) replaces the self-attention mechanism in Transformer-like encoders with frequency transformation, achieving comparable performance with significantly reduced computational costs. Fourier-Transformer (He et al., 2023) applies frequency-domain truncation to downsample hidden states in Transformer models for improved efficiency. More recently, FreqKV (Kai et al., 2025) introduces a frequency-based key-value compression technique that effectively extends the context window of LLMs. When it comes to VLMs, DocPedia (Feng et al., 2024) leverages DCT coefficients extracted directly from RGB images to perform vision encoding, enabling higher input resolutions for document understanding tasks. However, the potential of applying frequency-domain techniques to vision-token-level representations remains largely unexplored.

## 3. Preliminaries

### 3.1. VLM architecture

Vision-Language Models (VLMs) generally follow a modular design, composed of three principal components: a vision encoder to extract visual features; a projector that maps visual embeddings into the same space as language tokens; and a Large Language Model (LLM) for multimodal reasoning and text generation.

Specifically, LLaVA-v1.5 (Liu et al., 2024a) employs CLIP ViT-L/336px as its vision encoder and adopts a two-layer MLP as the projector. Its overall input format is structured as follows:

(*system prompt*)  
 USER: <image> (*user instruction*)  
 ASSISTANT:

The image placeholder token <image> is placed immediately after the system prompt, followed by the user instruction. After tokenization, the <image> token is replaced with image features extracted by the vision encoder. This token sequence, which interleaves text tokens with vision tokens, forms the input context for the backbone LLM.

The Qwen-VL series (Wang et al., 2024; Bai et al., 2025) adopts a similar high-level architecture but introduces significant enhancements. It is built upon the Qwen language models and features a re-engineered Vision Transformer for improved performance and efficiency, along with an MLP-based vision-language merger that achieves 75% visual token compression. Unlike LLaVA-v1.5, Qwen-VL series supports arbitrary-resolution inputs, resulting in a variable number of visual tokens.

### 3.2. Discrete Cosine Transform

The Discrete Cosine Transform (DCT) is a linear invertible function  $\Psi: \mathbb{R}^n \rightarrow \mathbb{R}^n$  that converts a sequence of discrete real numbers from the spatial domain to the frequency domain. It exhibits strong energy compaction properties, as most signal information (e.g., audio, image) tends to concentrate in the low-frequency components after transformation. Among the various variants of DCT, we conduct the most widely used type-II DCT.

Formally, for a real-valued sequence of length  $N$ , denoted as  $\langle x_i \rangle = \{x_0, x_1, \dots, x_{N-1}\}$ , the DCT transformation is given by:

$$f_m = \alpha_m \sum_{i=0}^{N-1} x_i \cdot \phi_N(m, i), \quad (1)$$

where  $m \in \{0, 1, \dots, N-1\}$ . The basis function  $\phi(\cdot, \cdot)$  and the normalization factor  $\alpha_m$  are defined as:

$$\begin{aligned} \phi_N(x, y) &= \cos \left[ \frac{\pi}{N} x \left( y + \frac{1}{2} \right) \right] \\ \alpha_m &= \begin{cases} \sqrt{\frac{1}{N}} & \text{if } m = 0, \\ \sqrt{\frac{2}{N}} & \text{otherwise.} \end{cases} \end{aligned} \quad (2)$$

The above transformation expresses  $\langle x_i \rangle$  as a sum of orthogonal cosine functions at different frequencies, where the coefficients represent the contribution of each frequency component.

Conversely, given the frequency representation  $\langle f_m \rangle = \{f_0, f_1, \dots, f_{N-1}\}$ , the original sequence can be recovered by the inverse Discrete Cosine Transform (iDCT):

$$x_i = \sum_{k=0}^{N-1} \alpha_k \cdot f_k \cdot \phi_N(k, i) \quad (3)$$

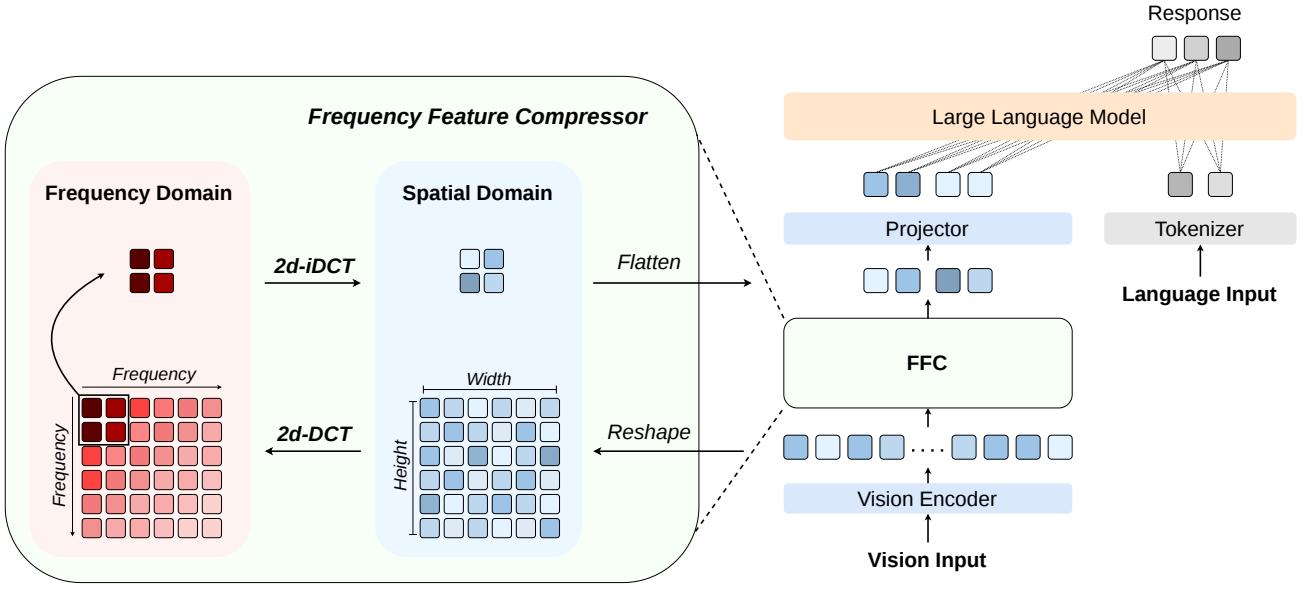


Figure 2. Illustration of the Fourier-VLM framework. After passing through the vision encoder, visual features are reshaped into a grid and transformed into the frequency domain. Darker colors indicate larger frequency magnitudes, while lighter colors represent smaller magnitudes. Only the low-frequency components are retained and subsequently converted back to the spatial domain, serving as the compressed visual features.

## 4. Fourier-VLM

In this section, we introduce Fourier-VLM, a highly efficient approach for visual representation compression. Building upon the typical VLM architecture, we retain the vision encoder, projector, and backbone LLM, while introducing a parameter-free component *Frequency Feature Compressor* (*FFC*) after the vision encoder to substantially reduce the number of vision tokens. The architecture and training details are provided below.

### 4.1. General Architecture

The architecture of Fourier-VLM is illustrated in Figure 2. The original vision input is first preprocessed (e.g., resizing or cropping for images, frame extraction for videos) before entering the vision encoder. The 3-channel resized image  $\mathbf{X}^v \in \mathbb{R}^{r \times r \times 3}$ , where  $r$  represents the input resolution, is initially passed through a CNN to extract low-level features, resulting in grid image features  $\hat{\mathbf{H}}^v \in \mathbb{R}^{N \times N \times h_v}$  where  $N^2$  is the number of patches and  $h_v$  is the output dimension of the CNN. These grid features are then flattened and passed through a pretrained ViT, which encodes the visual information into higher-level representations:

$$\mathbf{H}^v = \text{Vision-Encoder}(\mathbf{X}^v), \quad (4)$$

where  $\mathbf{H}^v \in \mathbb{R}^{N^2 \times h_v}$  and  $h_v$  is the output dimension of the vision encoder.

Subsequently, the visual features are processed by our FFC

module, which reduces the number of vision tokens from  $N^2$  to  $C^2$ :

$$\mathbf{H}_c^v = \text{FFC-Module}(\mathbf{H}^v), \quad (5)$$

where  $\mathbf{H}_c^v \in \mathbb{R}^{C^2 \times h_v}$ . The detailed structure is described in Section 4.2.

Finally, the compressed visual features are passed through the projector to align with the text embedding space, and processed alongside the text instructions by the backbone LLM, following the typical VLM architecture.

### 4.2. Frequency Feature Compressor

Inside the FFC module, the original visual features are first reshaped back to the grid size:

$$\mathbf{G}^v = \text{Reshape}(\mathbf{H}^v), \quad (6)$$

where  $\mathbf{G}^v \in \mathbb{R}^{N \times N \times h_v}$ . Then, a two-dimensional Discrete Cosine Transform (2d-DCT) is applied along the two spatial dimensions of size  $N$  to obtain the frequency representation  $\hat{\mathbf{F}}^v$ . Formally:

$$\hat{\mathbf{F}}_{m,n,:}^v = \alpha_m \alpha_n \sum_{p=0}^{N-1} \sum_{q=0}^{N-1} \mathbf{G}_{p,q,:}^v \cdot \phi_N(m, p) \cdot \phi_N(n, q), \quad (7)$$

where  $\hat{\mathbf{F}}^v \in \mathbb{R}^{N \times N \times h_v}$  and  $\phi(\cdot, \cdot)$ ,  $\alpha_m$  are specified in Equation (2). As depicted in Figure 1, the amplitude concentrates in the low-frequency components, which motivates

*Table 1.* Performance on 8 image-based benchmarks. “#I” denotes the number of vision tokens per image. All baseline results are reported from original papers, except MQT-LLaVA on VQA<sup>T</sup>, which was not originally reported and is newly evaluated by us. Note that PruMerge is the only baseline available in the 13B model setting.

Model	#I	VQA <sup>v2</sup>	GQA	SciQA	VQA <sup>T</sup>	POPE	MMB	LLaVA <sup>W</sup>	MMMU	Avg.
LLaVA-v1.5-7B	576	78.5	62.0	66.8	58.2	85.9	64.3	65.4	35.3	64.6
<i>Compression ratio: 55.6%</i>										
MQT-LLaVA (Hu et al., 2024)	256	76.8	61.6	67.6	53.2	84.4	64.3	<b>64.6</b>	<b>34.8</b>	63.4
<b>Fourier-LLaVA</b>	256	<b>78.6</b>	<b>62.7</b>	<b>69.9</b>	<b>56.0</b>	<b>85.3</b>	<b>66.4</b>	64.4	33.1	<b>64.6</b>
<i>Compression ratio: 75.0%</i>										
ATP-LLaVA (Ye et al., 2024)	144	76.4	59.5	<b>69.1</b>	-	84.2	<b>66.0</b>	-	-	-
MQT-LLaVA (Hu et al., 2024)	144	76.4	61.4	67.5	52.6	83.9	64.4	61.4	34.4	62.8
Prumerge (Shang et al., 2024)	144	76.8	-	68.3	<b>57.1</b>	84.0	64.9	-	-	-
<b>Fourier-LLaVA</b>	144	<b>77.7</b>	<b>61.7</b>	69.0	54.7	<b>85.0</b>	65.6	<b>67.8</b>	<b>35.3</b>	<b>64.6</b>
<i>Compression ratio: 88.9%</i>										
ATP-LLaVA (Ye et al., 2024)	88	73.3	56.8	67.2	-	82.6	<b>64.7</b>	-	-	-
MQT-LLaVA (Hu et al., 2024)	64	75.3	60.0	67.0	51.7	83.6	63.5	59.4	<b>34.4</b>	61.9
<b>Fourier-LLaVA</b>	64	<b>76.3</b>	<b>60.4</b>	<b>69.3</b>	<b>52.6</b>	<b>85.3</b>	<b>64.7</b>	<b>63.1</b>	<b>34.4</b>	<b>63.3</b>
<i>Compression ratio: 93.75%</i>										
MQT-LLaVA (Hu et al., 2024)	36	73.7	58.8	66.8	50.4	81.9	63.4	59.6	<b>34.4</b>	61.1
<b>Fourier-LLaVA</b>	36	<b>74.9</b>	<b>59.5</b>	<b>69.0</b>	51.0	<b>84.0</b>	<b>64.3</b>	<b>61.1</b>	32.8	<b>62.1</b>
PruMerge (Shang et al., 2024)	32	72.0	-	68.5	<b>56.0</b>	76.3	60.9	-	-	-
LLaVA-v1.5-13B	576	80.0	63.3	71.6	61.3	85.9	67.7	72.5	36.4	67.3
<i>Compression ratio: 75.0%</i>										
PruMerge (Shang et al., 2024)	144	77.8	-	71.0	<b>58.6</b>	84.4	65.7	-	-	-
<b>Fourier-LLaVA</b>	144	<b>78.7</b>	62.7	<b>71.1</b>	57.4	<b>85.4</b>	<b>66.2</b>	69.5	35.8	65.9

the pruning of high-frequency (less important) components:

$$\mathbf{F}^v = \hat{\mathbf{F}}^v [0:C, 0:C, :], \quad (8)$$

where  $\mathbf{F}^v \in \mathbb{R}^{C \times C \times h_v}$  and  $C^2$  is the number of preserved vision tokens.

Finally, we apply a two-dimensional inverse Discrete Cosine Transform (2d-iDCT) to reconstruct the spatial features from the frequency-domain tokens:

$$\mathbf{T}_{i,j,:}^v = \sum_{p=0}^{C-1} \sum_{q=0}^{C-1} \alpha_p \alpha_q \cdot \mathbf{F}_{p,q,:}^v \cdot \phi_C(p, i) \cdot \phi_C(q, j), \quad (9)$$

where  $\mathbf{T}^v \in \mathbb{R}^{C \times C \times h_v}$ , which is then flattened to obtain the compressed image features:

$$\mathbf{H}_c^v = \text{Flatten}(\mathbf{T}^v), \quad (10)$$

where  $\mathbf{H}_c^v \in \mathbb{R}^{C^2 \times h_v}$ .

Indeed, DCT can be efficiently implemented via FFT-based routines. Details on efficiency analysis and implementation are provided in Section 6 and Appendix B, respectively.

### 4.3. Training

Since the frequency-domain truncation inevitably alters the original feature distribution, additional training is necessary for the model to adapt to new visual representations. See Appendix A for detailed training settings.

For Fourier-LLaVA, built upon LLaVA-v1.5 series, we follow the same two-stage training procedure as the base model. Note that our FFC module is always applied but contains no trainable parameters.

**Feature Alignment:** In this stage, the vision encoder and the backbone LLM are frozen, and only the 2-layer MLP projector is trainable. The model learns to align the visual features with the language space using the 558k subset of the LAION-CC-SBU dataset. It takes around 2 hours for 1 epoch pretraining on 4×RTX 6000 Ada (48G) GPUs.

**Visual Instruction Tuning:** In this stage, only the vision encoder remains frozen, while both the 2-layer MLP projector and the backbone LLM are trainable. We use the 665k mixed instruction tuning data to teach the model to follow multimodal instructions. We adopt LoRA for efficient train-

Table 2. Performance of Qwen-VL series. “#I” denotes the average number of vision tokens per image. We report the performance of all models using lmms-eval, with the number of visual tokens standardized to 256 - 2304 for fair comparison.

Model Series		#I	MME	VQA <sup>T</sup>	POPE	RWQA	MMB	MMStar	Avg.
Qwen-2-VL-2B	Vanilla	553	1894.4	<b>78.2</b>	86.1	61.4	72.3	<b>44.2</b>	<b>66.8</b>
	<b>Fourier-Qwen-2</b>	236	<b>1917.1</b>	75.4	<b>86.5</b>	<b>61.6</b>	<b>72.6</b>	43.3	65.7
	△ vs. vanilla	-57.3%	+1.2%	-3.6%	+0.5%	+0.3%	+0.4%	-2.0%	-1.6%
Qwen-2.5-VL-3B	Vanilla	553	<b>2138.3</b>	<b>77.6</b>	87.2	59.6	<b>77.8</b>	<b>56.2</b>	71.1
	<b>Fourier-Qwen-2.5</b>	236	2110.8	76.1	<b>87.8</b>	<b>64.3</b>	76.8	55.3	<b>72.6</b>
	△ vs. vanilla	-57.3%	-1.3%	-1.9%	+0.7%	+7.9%	-1.3%	-1.6%	+2.1%

ing, setting the rank  $r = 128$  and scaling factor  $\alpha = 256$ . The training process takes approximately 25 hours for 2 epochs on 4×A100 (40G) GPUs with a batch size of 256.

For Fourier-Qwen, built upon the Qwen-VL series, we only perform continued finetuning on 600k single-image conversation samples from LLaVA-NeXT (Liu et al., 2024b), inserting our FFC module after the original MLP-based merger. Leveraging the dynamic input resolution capability of Qwen-VL, we set the number of visual tokens to range from 256 to 2304 during training to ensure efficient GPU memory usage.

## 5. Experiments

### 5.1. Settings

Following the evaluation setup of LLaVA-v1.5 series, we adopt eight image-based benchmarks in our main experiments: VQA<sup>v2</sup> (Goyal et al., 2017), GQA (Hudson & Manning, 2019), SciQA (Lu et al., 2022), VQA<sup>T</sup> (Singh et al., 2019), POPE (Li et al., 2023b), MMB (Liu et al., 2024c), LLaVA<sup>W</sup> (Liu et al., 2023), and MMMU (Yue et al., 2024). In addition, for evaluating Qwen-VL series, we include three more benchmarks: MME (Fu et al., 2024), RealWorldQA (xAI, 2024), and MMStar (Chen et al., 2024). These benchmarks cover a wide range of tasks, including image-based question answering, domain-specific knowledge grounding, and text recognition.

For most evaluations, We utilize *lmms-eval* (Zhang et al., 2024), a highly efficient evaluation framework, meticulously crafted for consistent and efficient evaluation of Large Multimodal Models. However, lmms-eval fails to reproduce some benchmarks due to differences in dataset splits. Therefore, for VQA<sup>T</sup>, MMB and LLaVA<sup>W</sup>, we follow the official evaluation scripts provided by LLaVA-v1.5, while other benchmarks are evaluated using lmms-eval. For LLaVA<sup>W</sup>, we use *gpt-4-0613* to evaluate performance with temperature set to 0.2.

### 5.2. Main Results

Table 1 presents the results of Fourier-LLaVA on various image-based benchmarks, evaluated with different numbers of preserved vision tokens:  $C^2 = 256, 144, 64, 36$ . Notably, even with a reduced number of vision tokens, Fourier-LLaVA achieves competitive results. With 256 vision tokens ( $44\% \times 576$ ), Fourier-LLaVA outperforms the vanilla LLaVA-v1.5-7B on 4 benchmarks. Moreover, with just 144 vision tokens ( $25\% \times 576$ ), it still surpasses the base model in terms of average score. Even at the lowest setting of 36 vision tokens ( $6.25\% \times 576$ ), Fourier-LLaVA only exhibits a 3.87% drop in average, while continuing to outperform the base model on SciQA and MMB.

For other token compression approaches, we compare the performance under the same number of vision tokens. Note that all the methods mentioned are implemented based on LLaVA-v1.5 series, with CLIP ViT-L/336px as the vision encoder and Vicuna-v1.5 as the backbone LLM, and all require a two-stage training process. Generally, Fourier-LLaVA outperforms MQT-LLaVA, LLaVA-PruMerge and ATP-LLaVA across nearly all benchmarks at the same vision token count, achieving a 2.2% increase in average scores.

Furthermore, when scaled to a 13B backbone, Fourier-LLaVA continues to outperform PruMerge under the same vision token budget. Besides, with only 25% of the original vision tokens, our approach only incurs a 2.1% drop in the average score compared to the base model.

We further implement our approach on Qwen-VL series (Wang et al., 2024; Bai et al., 2025). As shown in Table 2, our method still maintains competitive performance when applied to both Qwen-2-VL-2B and Qwen-2.5-VL-3B. Despite an additional 57.3% reduction in vision tokens via our Frequency Feature Compressor, Fourier-Qwen-2 incurs only a 1.6% drop in average performance. Remarkably, Fourier-Qwen-2.5 even outperforms the baseline by 2.1%, and achieves substantial gains on real-world question answering tasks, with a 7.9% improvement on RealWorldQA. These results highlight the strong generalization capability

Table 3. Performance on MVBench. “#V” denotes the average number of visual tokens per video. We report the performance of baselines and our models using lmms-eval. For fair comparison, when evaluating Qwen-VL series, the number of visual tokens per frame is set to 256 - 384, and the maximum number of frames is 16.

Model	Size	#V	Action	Object	Position	Scene	Count	Attribute	Pose	Character	Cognition	Avg.
VideoChatGPT	7B	356	32.1	40.7	21.5	31.0	28.0	44.0	29.0	33.0	30.3	32.7
Video-LLaMA	7B	40	34.4	42.2	22.5	43.0	28.3	39.0	32.5	40.0	29.3	34.1
Video-LLaVA	7B	4096	48.0	46.5	27.8	84.5	35.5	45.8	34.0	42.5	34.2	43.1
VideoChat	7B	4096	38.0	41.2	26.3	48.5	27.8	44.3	26.5	41.0	27.7	35.5
VideoChat2	7B	4096	61.3	57.3	23.0	88.5	40.5	51.3	49.0	36.5	47.0	51.1
LLaVA-v1.5	7B	2304	<b>52.8</b>	43.7	<b>31.5</b>	<b>83.5</b>	37.0	<b>45.3</b>	<b>44.5</b>	<b>50.0</b>	<b>37.3</b>	<b>45.6</b>
<b>Fourier-LLaVA</b>	7B	288	48.1	<b>46.3</b>	31.0	81.0	<b>40.5</b>	41.3	36.0	49.0	36.3	44.0
$\Delta$ vs. LLaVA-v1.5		-87.5%	-4.7	+2.6	-0.5	-2.5	+3.5	-4.0	-8.5	-1.0	-1.0	-1.6
Qwen-2-VL	2B	3193	<b>68.5</b>	<b>65.7</b>	<b>44.8</b>	<b>90.5</b>	<b>60.8</b>	<b>65.3</b>	<b>53.5</b>	<b>59.0</b>	47.8	<b>61.4</b>
<b>Fourier-Qwen-2</b>	2B	1328	67.7	63.3	40.0	88.5	55.5	64.8	51.0	57.0	<b>50.2</b>	59.8
$\Delta$ vs. Qwen-2-VL		-58.4%	-0.8	-2.4	-4.8	-2.0	-5.3	-0.5	-2.5	-2.0	+2.4	-1.6
Qwen-2.5-VL	3B	3193	<b>68.3</b>	<b>67.7</b>	<b>48.8</b>	<b>91.0</b>	<b>63.0</b>	<b>75.0</b>	<b>51.5</b>	<b>75.5</b>	56.0	<b>65.2</b>
<b>Fourier-Qwen-2.5</b>	3B	1328	67.2	65.7	48.5	90.5	54.8	70.5	47.5	66.0	<b>57.5</b>	62.9
$\Delta$ vs. Qwen-2.5-VL		-58.4%	-1.1	-2.0	-0.3	-0.5	-8.2	-4.5	-4.0	-9.5	+1.5	-2.3

of our approach across different vision encoders, backbone LLMs, and input resolutions.

## 6. Analyses

### 6.1. Time Complexity

A direct computation of the Discrete Cosine Transform (DCT) is inefficient, requiring  $\mathcal{O}(N^2)$  operations for an N-point sequence. However, by leveraging the Fast Fourier Transform (FFT) operator, we can accelerate the DCT computation to  $\mathcal{O}(N \log N)$ , same for iDCT.

For a real-valued sequence of length N, denoted as  $\langle x_i \rangle = \{x_0, x_1, \dots, x_{N-1}\}$ , we first rearrange it by separating the even- and odd-indexed elements and reversing the order of the odd-indexed elements. If N is even, the reordered sequence  $\langle y_k \rangle$  is defined as:

$$y_k = \begin{cases} x_{2k} & k = 0, 1, \dots, \frac{N-2}{2} \\ x_{2N-1-2k} & k = \frac{N}{2}, \dots, N-1 \end{cases} \quad (11)$$

That is,  $\langle y_k \rangle = \{x_0, x_2, \dots, x_{N-2}, x_{N-1}, x_{N-3}, \dots, x_1\}$ . A similar rearrangement applies when N is odd.

Next, we compute the FFT of  $\langle y_n \rangle$ , yielding the sequence  $\langle z_m \rangle$ :

$$\langle z_m \rangle = FFT(\langle y_n \rangle), \quad (12)$$

where  $\langle z_m \rangle$  is a complex sequence of length N. The DCT coefficients can then be computed as:

$$f_k = \Re(z_k) \cdot \cos\left(\frac{k\pi}{2N}\right) - \Im(z_k) \cdot \sin\left(\frac{k\pi}{2N}\right), \quad (13)$$

where  $\Re(\cdot)$  and  $\Im(\cdot)$  denote the real and imaginary parts of a complex number, respectively.

For the two-dimensional DCT (2d-DCT) applied to a matrix  $\mathbf{X} \in \mathbb{R}^{N \times N}$ , the transformation is equivalent to performing a one-dimensional DCT along each row, followed by another one-dimensional DCT along each column. Consequently, applying 2d-DCT to an  $N \times N$  matrix results in a total time complexity of  $\mathcal{O}(N^2 \log N)$ .

Therefore, the FFC module in Fourier-VLM, which first applies a 2d-DCT on the grid image features of size  $N \times N \times h_v$  and then an 2d-iDCT on the compressed image features of size  $C \times C \times h_v$ , has an overall time complexity of:

$$\mathcal{O}(N^2 \log N + C^2 \log C) \quad (14)$$

Table 4 lists the time complexity of standard modules for processing  $(B, N^2, h_v)$  inputs. Under the typical condition  $h_v \gg M > N$ , our FFC module achieves the lowest theoretical time complexity, offering significantly higher computational efficiency than attention-based (e.g., ATP-LLaVA) and query-based (e.g., MQT-LLaVA) compression approaches.

### 6.2. Floating Point Operations and Latency

In practice, we evaluate the FLOPs (Floating Point Operations) of Fourier-LLaVA under varying numbers of vision tokens using *calflops* on an RTX 4090 (24G) GPU. Additionally, we measure the Time to First Token (TTFT) and KV cache usage on an A100 (40G) GPU.

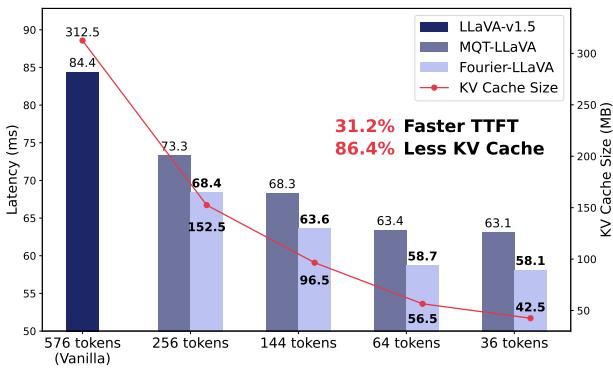
*Table 4.* Time complexity of common modules processing visual features of shape  $(B, N^2, h_v)$ . Here,  $M$  denotes the number of learnable queries. Note that MQT-LLaVA adopts a Query Transformer for visual token compression.

Module	Time Complexity
MLP	$\mathcal{O}(B \cdot h_v^2 \cdot N^2)$
Self-Attention	$\mathcal{O}(B \cdot h_v \cdot N^4)$
Query Transformer	$\mathcal{O}(B \cdot h_v \cdot N^2 \cdot M)$
<b>FFC (Ours)</b>	$\mathcal{O}(B \cdot h_v \cdot N^2 \cdot \log N)$

As illustrated in Table 5 and Figure 3, Fourier-LLaVA achieves a substantial reduction in computational cost compared to LLaVA-v1.5-7B, reducing FLOPs by 83.8%. Furthermore, our method lowers KV cache usage by 86.4% and improves inference speed by 31.2% in TTFT, outperforming other token compression baselines such as MQT-LLaVA under equivalent visual token counts. These improvements highlight the efficiency of our parameter-free compression approach, distinguishing Fourier-VLM from prior works that rely on attention-based reductions or learnable parameters, which is crucial for enabling real-time deployment of VLMs on resource-constrained devices.

*Table 5.* Flops of Fourier-LLaVA

Model	#I	FLOPs (T)	↓ (%)
LLaVA-v1.5-7B	576	8.54	-
<b>Fourier-LLaVA</b>	256	4.30	49.6
	144	2.81	67.1
	64	1.75	79.5
	36	<b>1.38</b>	<b>83.8</b>



*Figure 3.* Latency and KV cache usage of Fourier-LLaVA.

### 6.3. Applicability to Video Tasks

Fourier-LLaVA and Fourier-Qwen series, though trained only on single-image conversations, generalize well to zero-

shot video tasks. We further evaluate on MVBench (Li et al., 2024a), a comprehensive multi-modal video understanding benchmark that encompasses 20 challenging video tasks.

As shown in Table 3, our method leverages significantly fewer vision tokens while outperforming various open-source VLMs. Specifically, Fourier-Qwen series achieves a 58.4% reduction in visual tokens with only a 3.1% drop in average performance, while Fourier-LLaVA retains 96.5% of the base model’s performance utilizing merely 12.5% of the video tokens, and still surpasses the base model on the Object and Count tasks. This zero-shot capability in video understanding highlights the efficiency and robustness of our method, and suggests promising directions for future work on scaling frequency-aware compression to video-language models.

## 7. Conclusion

In this paper, we propose Fourier-VLM, an efficient and effective approach for vision token compression of Vision-Language Models within the frequency domain. By integrating our Frequency Feature Compressor (FFC) into both LLaVA-v1.5 and Qwen-VL series, Fourier-VLM strikes an excellent balance between performance and efficiency. It retains over 96% average accuracy across 8 image-based benchmarks while using only 6.25% of the original vision tokens, additionally reducing FLOPs to 16.16% and speeding up inference by 31.2%. Our approach also generalizes well across model architectures and input resolutions, and demonstrates promising zero-shot capability on video understanding tasks. These results highlight a favorable trade-off between cost and performance, enabling more efficient deployment of VLMs in practical scenarios.

## References

- Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., Zhu, Y., Yang, M., Li, Z., Wan, J., Wang, P., Ding, W., Fu, Z., Xu, Y., Ye, J., Zhang, X., Xie, T., Cheng, Z., Zhang, H., Yang, Z., Xu, H., and Lin, J. Qwen2.5-vl technical report, 2025. URL <https://arxiv.org/abs/2502.13923>.
- Chen, L., Li, J., Dong, X., Zhang, P., Zang, Y., Chen, Z., Duan, H., Wang, J., Qiao, Y., Lin, D., and Zhao, F. Are we on the right way for evaluating large vision-language models?, 2024. URL <https://arxiv.org/abs/2403.20330>.
- Feng, H., Liu, Q., Liu, H., Tang, J., Zhou, W., Li, H., and Huang, C. Docpedia: Unleashing the power of large multimodal model in the frequency domain for versatile document understanding, 2024. URL <https://arxiv.org/abs/2311.11810>.

- Fu, C., Chen, P., Shen, Y., Qin, Y., Zhang, M., Lin, X., Yang, J., Zheng, X., Li, K., Sun, X., Wu, Y., and Ji, R. Mme: A comprehensive evaluation benchmark for multimodal large language models, 2024. URL <https://arxiv.org/abs/2306.13394>.
- Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D. Making the v in vqa matter: Elevating the role of image understanding in visual question answering, 2017. URL <https://arxiv.org/abs/1612.00837>.
- He, Z., Yang, M., Feng, M., Yin, J., Wang, X., Leng, J., and Lin, Z. Fourier transformer: Fast long range modeling by removing sequence redundancy with fft operator. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8954–8966. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-acl.570. URL <http://dx.doi.org/10.18653/v1/2023.findings-acl.570>.
- Hu, W., Dou, Z.-Y., Li, L. H., Kamath, A., Peng, N., and Chang, K.-W. Matryoshka query transformer for large vision-language models, 2024. URL <https://arxiv.org/abs/2405.19315>.
- Huang, M., Huang, R., Shi, H., Chen, Y., Zheng, C., Sun, X., Jiang, X., Li, Z., and Cheng, H. Efficient multi-modal large language models via visual token grouping, 2024. URL <https://arxiv.org/abs/2411.17773>.
- Hudson, D. A. and Manning, C. D. Gqa: A new dataset for real-world visual reasoning and compositional question answering, 2019. URL <https://arxiv.org/abs/1902.09506>.
- Kai, J., Zeng, B., Wang, Y., Bai, H., He, Z., Jiang, B., and Lin, Z. Freqkv: Frequency domain key-value compression for efficient context window extension, 2025. URL <https://arxiv.org/abs/2505.00570>.
- Lee-Thorp, J., Ainslie, J., Eckstein, I., and Ontanon, S. Fnet: Mixing tokens with fourier transforms, 2022. URL <https://arxiv.org/abs/2105.03824>.
- Li, J., Li, D., Savarese, S., and Hoi, S. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023a. URL <https://arxiv.org/abs/2301.12597>.
- Li, K., Wang, Y., He, Y., Li, Y., Wang, Y., Liu, Y., Wang, Z., Xu, J., Chen, G., Luo, P., Wang, L., and Qiao, Y. Mvbench: A comprehensive multi-modal video understanding benchmark, 2024a. URL <https://arxiv.org/abs/2311.17005>.
- Li, K. Y., Goyal, S., Semedo, J. D., and Kolter, J. Z. Inference optimal vlms need only one visual token but larger models, 2024b. URL <https://arxiv.org/abs/2411.03312>.
- Li, Y., Du, Y., Zhou, K., Wang, J., Zhao, W. X., and Wen, J.-R. Evaluating object hallucination in large vision-language models, 2023b. URL <https://arxiv.org/abs/2305.10355>.
- Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning, 2023. URL <https://arxiv.org/abs/2304.08485>.
- Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved baselines with visual instruction tuning, 2024a. URL <https://arxiv.org/abs/2310.03744>.
- Liu, H., Li, C., Li, Y., Li, B., Zhang, Y., Shen, S., and Lee, Y. J. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024b. URL <https://llava-vl.github.io/blog/2024-01-30-llava-next/>.
- Liu, Y., Duan, H., Zhang, Y., Li, B., Zhang, S., Zhao, W., Yuan, Y., Wang, J., He, C., Liu, Z., Chen, K., and Lin, D. Mmbench: Is your multi-modal model an all-around player?, 2024c. URL <https://arxiv.org/abs/2307.06281>.
- Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., and Kalyan, A. Learn to explain: Multimodal reasoning via thought chains for science question answering, 2022. URL <https://arxiv.org/abs/2209.09513>.
- Shang, Y., Cai, M., Xu, B., Lee, Y. J., and Yan, Y. Llava-prumerge: Adaptive token reduction for efficient large multimodal models, 2024. URL <https://arxiv.org/abs/2403.15388>.
- Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., and Rohrbach, M. Towards vqa models that can read, 2019. URL <https://arxiv.org/abs/1904.08920>.
- Wang, P., Bai, S., Tan, S., Wang, S., Fan, Z., Bai, J., Chen, K., Liu, X., Wang, J., Ge, W., Fan, Y., Dang, K., Du, M., Ren, X., Men, R., Liu, D., Zhou, C., Zhou, J., and Lin, J. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution, 2024. URL <https://arxiv.org/abs/2409.12191>.
- xAI. Grok-1.5V and RealWorldQA Evaluation. <https://x.ai/news/grok-1.5v>, 2024. Accessed: 2025-07-29.

Xu, K., Qin, M., Sun, F., Wang, Y., Chen, Y.-K., and Ren, F. Learning in the frequency domain, 2020. URL <https://arxiv.org/abs/2002.12416>.

Ye, X., Gan, Y., Ge, Y., Zhang, X.-P., and Tang, Y. Atp-llava: Adaptive token pruning for large vision language models, 2024. URL <https://arxiv.org/abs/2412.00447>.

Yue, X., Ni, Y., Zhang, K., Zheng, T., Liu, R., Zhang, G., Stevens, S., Jiang, D., Ren, W., Sun, Y., Wei, C., Yu, B., Yuan, R., Sun, R., Yin, M., Zheng, B., Yang, Z., Liu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi, 2024. URL <https://arxiv.org/abs/2311.16502>.

Zhang, K., Li, B., Zhang, P., Pu, F., Cahyono, J. A., Hu, K., Liu, S., Zhang, Y., Yang, J., Li, C., and Liu, Z. Lmms-eval: Reality check on the evaluation of large multimodal models, 2024. URL <https://arxiv.org/abs/2407.12772>.

Zhang, S., Fang, Q., Yang, Z., and Feng, Y. Llava-mini: Efficient image and video large multimodal models with one vision token, 2025. URL <https://arxiv.org/abs/2501.03895>.

## A. Training Details

The training details for Fourier-LLaVA and Fourier-Qwen are provided in Table 6. Note that for Qwen-VL, the pixel input range is set to 200,704 - 1,806,336, corresponding to 256 - 2304 vision tokens, to maintain efficient GPU memory usage.

Table 6. Training details for Fourier-VLM.

Settings	Fourier-LLaVA		Fourier-Qwen
	Stage 1	Stage 2	
<i>Trainable modules</i>			
Vision Encoder			✓
Projector	✓	✓	✓
Language Model		✓	✓
Dataset	558k	665k	600k
Epochs	1	2	2
LoRA $r/\alpha$	-	128 / 256	-
Batch Size	256	256	128
Learning Rate	1e-3	2e-4	1e-6
MM LR	-	2e-5	1e-5
Vision LR	-	-	1e-6
Optimizer	AdamW		AdamW
Schedule	Cosine		Cosine
Warmup Ratio	0.03		0.03

## B. Algorithms

Algorithms 1 and 2 present the implementations of the type-II Discrete Cosine Transform (DCT) and its inverse (iDCT), which serve as core building blocks of our Frequency Feature Compressor (FFC) in Algorithm 3.

### Algorithm 1 Discrete Cosine Transform

```

1: Input: Tensor  $x$  of shape  $(\dots, N)$ 
2: Output: Tensor  $V$  of shape  $(\dots, N)$ 
3:  $X \leftarrow \text{Reshape}(x, (-1, N))$ 
4:  $v \leftarrow \text{Concat}(X[:, 0 :: 2], \text{Flip}(X[:, 1 :: 2]), \text{dim} = 1)$ 
5:  $V_c \leftarrow \text{FFT}(v)$ 
6:  $k \leftarrow -\frac{\pi}{2N} \cdot \text{arange}(N)$ 
7:  $W_r \leftarrow \cos(k), \quad W_i \leftarrow \sin(k)$ 
8:  $V \leftarrow \Re(V_c) \cdot W_r - \Im(V_c) \cdot W_i$ 
9: if Normalize is required then
10:    $V[:, 0] \leftarrow V[:, 0] / (2\sqrt{N})$ 
11:    $V[:, 1 :] \leftarrow V[:, 1 :] / (2\sqrt{N/2})$ 
12: end if
13:  $V \leftarrow 2 \cdot \text{Reshape}(V, \text{original shape of } x)$ 
14: return  $V$ 

```

**Algorithm 2** Inverse Discrete Cosine Transform

---

```

1: Input:  $V$  of shape  $(\dots, N)$ 
2: Output:  $x$  of shape  $(\dots, N)$ 
3:  $X \leftarrow \text{Reshape}(V, (-1, N))/2$ 
4: if Normalize is required then
5:    $X[:, 0] \leftarrow X[:, 0] \cdot 2\sqrt{N}$ 
6:    $X[:, 1:] \leftarrow X[:, 1:] \cdot 2\sqrt{N/2}$ 
7: end if
8:  $k \leftarrow \frac{\pi}{2N} \cdot \text{arange}(N)$ 
9:  $W_r \leftarrow \cos(k), W_i \leftarrow \sin(k)$ 
10:  $V_t^r \leftarrow X$ 
11:  $V_t^i \leftarrow \text{Concat}(0, -\text{Flip}(X)[:, :-1])$ 
12:  $V^r \leftarrow V_t^r \cdot W_r - V_t^i \cdot W_i$ 
13:  $V^i \leftarrow V_t^r \cdot W_i + V_t^i \cdot W_r$ 
14:  $Z \leftarrow \text{Complex}(V^r, V^i)$ 
15:  $z \leftarrow \text{Real}(\text{IFFT}(Z))$ 
16: Initialize  $x$  as zeros of shape  $z$ 
17:  $x[:, 0 :: 2] += z[:, : N - \lfloor N/2 \rfloor]$ 
18:  $x[:, 1 :: 2] += \text{Flip}(z)[:, : \lfloor N/2 \rfloor]$ 
19:  $x \leftarrow \text{Reshape}(x, \text{original shape of } V)$ 
20: return  $x$ 
```

---

**Algorithm 3** Frequency Feature Compressor

---

```

1: Input: Image feature  $I$  of shape  $(B, N^2, h)$ 
2: Output: Compressed image feature  $I^c$  of shape  $(B, C^2, h)$ 
3:  $X \leftarrow \text{Reshape}(\text{Transpose}(I, (1, 2)), (B, h, N, N))$ 
4:  $F_1 \leftarrow \text{DCT}(X)$ 
5:  $F_2 \leftarrow \text{DCT}(\text{Transpose}(F_1, (2, 3)))$ 
6:  $F \leftarrow \text{Transpose}(F_2, (2, 3))$ 
7:  $F^c \leftarrow F[:, :, : C, : C]$ 
8:  $R_1 \leftarrow \text{IDCT}(F^c)$ 
9:  $R_2 \leftarrow \text{IDCT}(\text{Transpose}(R_1, (2, 3)))$ 
10:  $Y \leftarrow \text{Transpose}(R_2, (2, 3))$ 
11:  $I^c \leftarrow \text{Transpose}(\text{Reshape}(Y, (B, h, C^2)), (1, 2))$ 
12: return  $I^c$ 
```

---

## C. Impact on Model Output

When compressing images, truncating the high-frequency components of an RGB image leads to blurring, as seen in the JPEG image compression standard. We further explore the effects of truncating high-frequency components in visual feature representations, aiming to understand its impact on model output and information retention.

Indeed, compressing visual features in the frequency domain introduces perceptual distortions in fine details while preserving the overall structure, similar to JPEG compression. Specifically, as shown in Figure 4, the original “heart-shaped sign” progressively loses its distinct shape and is interpreted as “a banner”, “a balloon”, or even “a box” as the compression ratio increases. Additionally, the perceived position of

the sign shifts from the left side to “the center of the image”, while the overall message, “Happy Mother’s Day”, remains correctly understood. At the highest compression ratio, the model begins to generate erroneous details, hallucinating “a teddy bear, a cupcake, and a rose.” However, these distortions and hallucinations can be effectively mitigated through finetuning, significantly enhancing generation quality and demonstrates the robustness of our method.

Vision input:	Text input: Describe this image in detail.
	<b>576 vision tokens:</b> The image features <u>a heart-shaped sign</u> with the words " <u>Happy Mother's Day</u> " written in a cursive style. The heart is filled with <u>pink hearts</u> , giving it a warm and loving appearance. The sign is placed on <u>a white background</u> , making the message stand out. The heart-shaped sign is a creative and thoughtful way to celebrate Mother's Day and express appreciation for the mothers in our lives.
	<b>256 vision tokens:</b> The image features a heart-shaped sign with the words " <u>Happy Mother's Day</u> " written on it. The heart is <u>pink and white</u> , and it is placed in <u>the center of the image</u> . The sign is likely meant to celebrate and honor mothers on this special day.
	<b>144 vision tokens:</b> The image features a <u>heart-shaped banner</u> with the words " <u>Happy Mother's Day</u> " written on it. The heart is prominently displayed in <u>the center of the banner</u> , making it the focal point of the design. The banner is likely meant to be a celebratory and warm tribute to mothers on this special day.
	<b>64 vision tokens:</b> The image features a <u>heart-shaped balloon floating in the air</u> , with the words " <u>Happy Mother's Day</u> " written on it. The balloon is positioned towards the <u>left side</u> of the image, and it appears to be the main focus of the scene.
	<b>36 vision tokens:</b> The image features a <u>heart-shaped box with a red background</u> , placed on <u>a table</u> . The box is filled with various items, including <u>a teddy bear, a cupcake, and a rose</u> . The teddy bear is positioned towards the <u>left side</u> of the box, while the cupcake is located towards the <u>right side</u> , and the rose is placed in the <u>center</u> of the box. The arrangement of these items creates a visually appealing and romantic display.

Figure 4. Model outputs with varying numbers of vision tokens. Here, our FFC module is directly applied to LLaVA-v1.5-7B without any additional training. Correct information is underlined, while incorrect text is highlighted in red.