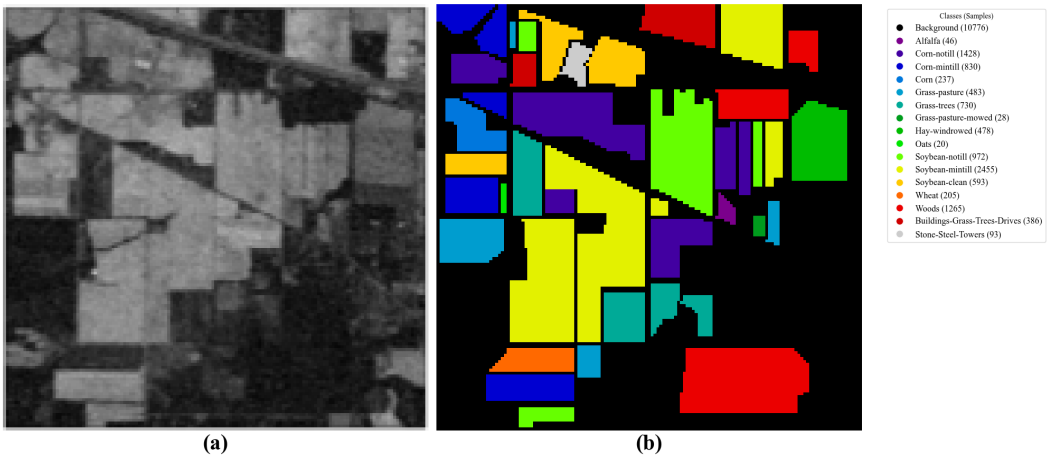
APPENDIX

A THE HYPERSPECTRAL IMAGE CLASSIFICATION EXPERIMENTS

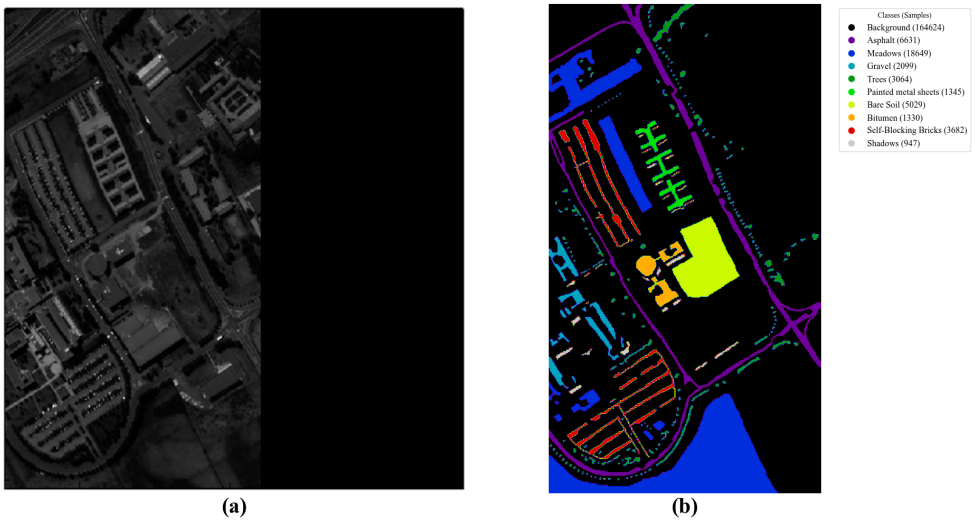
A.1 DATASETS

Due to the absence of a comprehensive hyperspectral dataset for object classification as extensive as ImageNet in the open-source domain, we opted for the next best alternative: three remote sensing datasets commonly used in hyperspectral image classification tasks—Indian Pines, Pavia University, and Salinas. Below is a detailed introduction to these three datasets.



(a) (b)

Figure 7: **Indian Pines dataset:** (a) Grayscale schematic diagram, (b) Ground truth labels (the pixel count for each sample type is displayed on the legend).



(a) (b)

Figure 8: **Pavia University dataset:** (a) Grayscale schematic diagram, (b) Ground truth labels (the pixel count for each sample type is displayed on the legend).

Indian Pines (Fig. 7): The AVIRIS sensor captured the Indian Pines scene over a test site in Northwestern Indiana, which includes a 145x145 pixel array with 224 spectral reflectance bands spanning the 0.4–2.5 micrometer wavelength range. This dataset is a subset of a larger scene, predominantly composed of two-thirds agricultural land and one-third forest or other perennial natural vegetation. The area features significant infrastructure, including major highways, a railway, low-density housing, other built structures, and minor roads. Captured in June, the image shows early-stage crops like

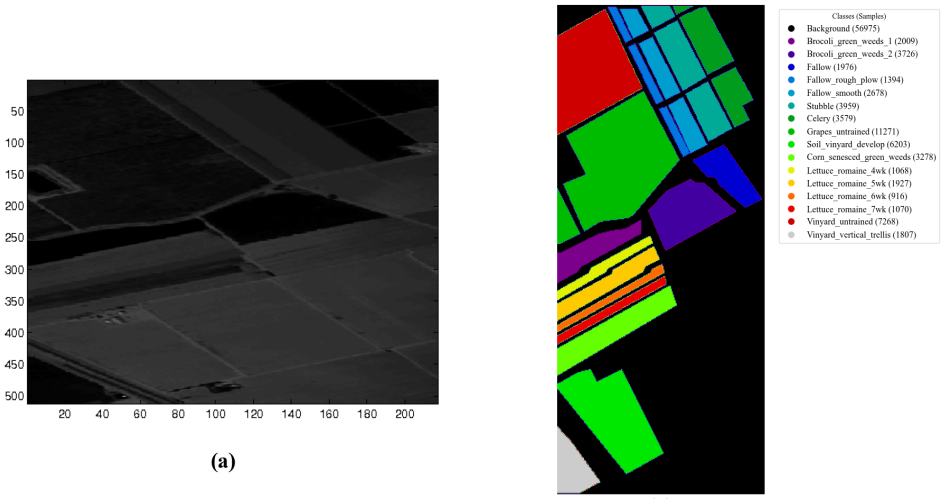


Figure 9: **Salinas dataset:** (a) Grayscale schematic diagram, (b) Ground truth labels (the pixel count for each sample type is displayed on the legend).

corn and soybeans with less than 5% coverage. The provided ground truth is divided into sixteen classes, which are not entirely distinct. To avoid water absorption features, the number of spectral bands has been reduced to 200 by excluding bands [104-108], [150-163], and 220.

Pavia University (Fig. 8): This scene was captured by the ROSIS sensor during a flight campaign over Pavia in Northern Italy. It comprises 103 spectral bands. The Pavia University image is 610 by 610 pixels, though some samples in both images are void of information and must be discarded prior to analysis. The geometric resolution of the image is 1.3 meters. The image ground truths distinguish nine different classes. The figures show the discarded samples as wide black strips.

Salinas (Fig. 9): The Salinas scene was collected using the 224-band AVIRIS sensor over Salinas Valley, California, noted for its high spatial resolution with 3.7-meter pixels. The captured area includes 512 lines by 217 samples. Similar to the Indian Pines scene, 20 water absorption bands were omitted, specifically bands: [108-112], [154-167], and 224. This imagery was only available in the form of at-sensor radiance data. The scene features a variety of elements including vegetables, bare soils, and vineyard fields. The Salinas ground truth encompasses 16 classes.

A.2 Task description and experimental setup

We conducted pixel classification tasks on these three datasets, classifying each pixel based on its own spectral data as well as that of its surrounding pixels. Although there is no universally accepted standard for dividing training and validation data, we followed the conventional approach of a large validation set and a small training set (6:4), ensuring no overlap between training and validation data. The dataset partitioned for this task comprises samples, each a 9x9 sub-image consisting of the target pixel for classification and its surrounding pixels.

To tailor to this task, we adjusted the configuration of LUM-ViT: the number of input channels was matched to the spectral channels of the hyperspectral datasets, the patch size was set to 9, and images input into the network were first enlarged to 27x27, resulting in 9 patches. Additionally, we reduced the embedding dimension to 192, while the rest of the settings remained largely consistent with those used for ImageNet.

We conducted training for 100 epochs from scratch on all three datasets, including a 5-epoch warmup, with a batch size of 64 and a peak learning rate of 0.01, utilizing cosine annealing. We observed some degree of overfitting and made fine adjustments accordingly. A large weight decay (0.001) setting could mitigate this issue.

B BASE METHODS FOR COMPARISON

In the experimental section, we compared LUM-ViT with DU-ViT, Random-mask-ViT, Mag-mask-ViT and CS-method on the ImageNet-1k classification task to demonstrate the superior performance and design of LUM-ViT. This section will introduce the experimental setup for these four methods.

B.1 DU-VIT

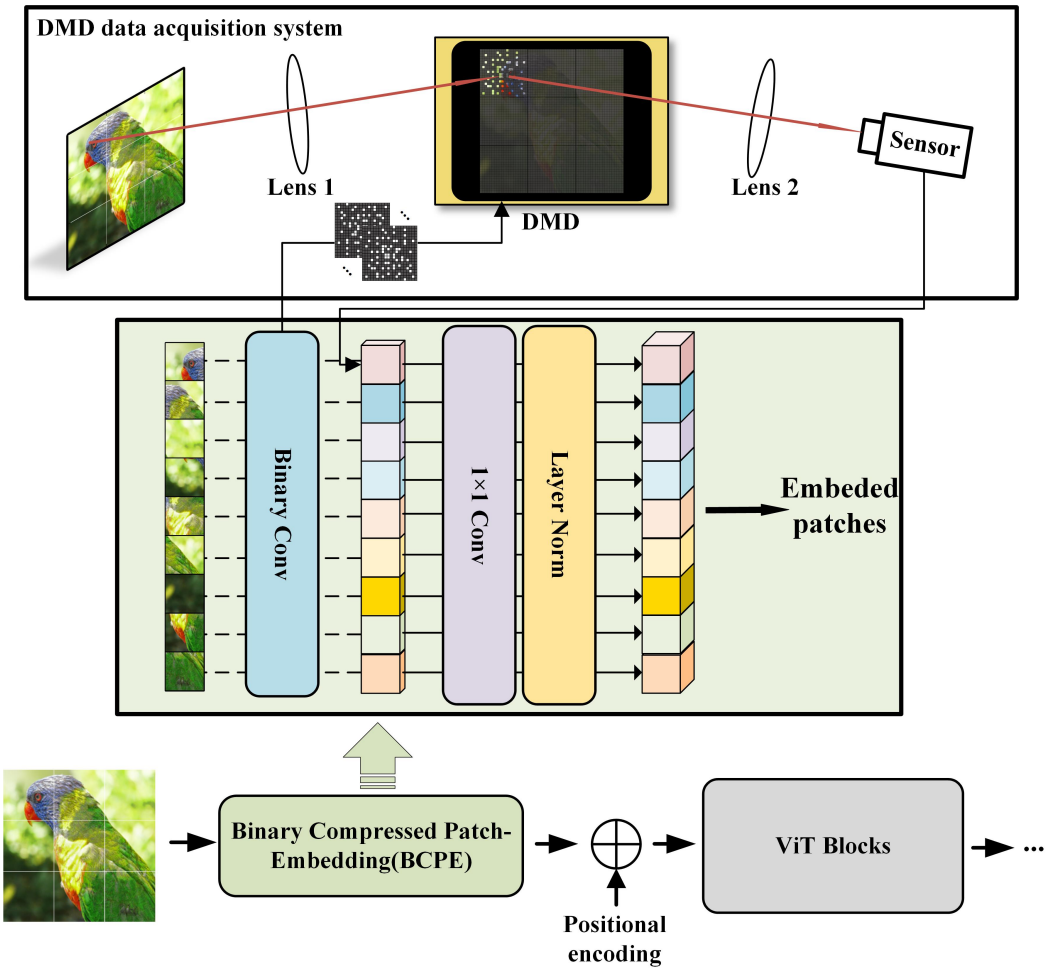


Figure 10: **DU-ViT.** We integrated the design of under-sampling and binary convolution within the patch-embedding layer, termed the Binary Compressed Patch-Embedding (BCPE) layer. The dashed lines in the diagram represent data flows exclusive to the training phase.

In early experiments, we considered a straightforward method to reduce the number of convolutional kernels in the patch-embedding layer to achieve under-sampling. We experimented with this method but observed poor performance. We speculate that the reduction in convolutional kernels led to a significant decrease in the embedded dimension of the data input to the backbone, and this reduction in data dimensionality severely limited the model's expressive capability. Building on this, we added 1×1 convolutional layer after the reduced-kernel convolution to restore the embedded dimension to its pre-under-sampling size. We refer to this network as DU-ViT, that is, Directly Under-sampling ViT. This network achieved reasonably good results in our experiments and can serve as a baseline model for under-sampling.

The structure of DU-ViT is shown in Fig. 10. To accommodate DMD operations, this method also employs binary convolutional layers. In our experiments, we primarily observed that adding a

Layer Normalization after the 1×1 convolution further enhances performance, hence this setup was retained. During the training of DU-ViT, we maintained most of the training settings consistent with LUM-ViT to facilitate comparison, including the three-stage training strategy.

We noted that the performance difference between DU-ViT and LUM-ViT remains relatively stable when the under-sampling rate is above 10%. However, under extreme under-sampling conditions, such as 2%, there is a significant gap in performance between DU-ViT and LUM-ViT.

B.2 RANDOM-MASK-VIT AND MAG-MASK-VIT

We believe that LUM-ViT's ability to maintain performance at low under-sampling rates is not solely due to the adoption of a masking approach for under-sampling, but also attributed to the success of the learnable mask strategy. To confirm the superiority of the learnable mask approach, we conducted experiments with other masking strategies, summarized into the following two schemes:

Random-mask-ViT: This approach utilizes a random mask strategy. Specifically, we randomly generated a mask and applied it to the output of the patch-embedding layer in the pre-trained model, followed by finetuning with the mask fixed in place.

Mag-mask-ViT: Drawing on common strategies from neural network pruning, we implemented a mask based on data magnitude. Specifically, we calculated the average magnitude at each position of the patch-embedding layer's output from the pre-trained model using the training dataset. We then selected the positions with the highest magnitudes to retain according to the required under-sampling ratio, masking the rest. With the mask fixed, we proceeded to finetune the model.

Under the condition that most training settings remained unchanged, we trained and tested the Random-mask-ViT and Mag-mask-ViT models. We observed that the performance of these two schemes was inferior to DU-ViT, and even less so to LUM-ViT. Similar to DU-ViT, we also noted a more rapid decline in performance at extreme under-sampling rates (2%). Based on the comparison of LUM-ViT with these two models, we conclude that the learnable mask strategy with trainable parameters is superior, adapting well to the target datasets and downstream tasks.

B.3 CS-METHOD

Compressed Sensing (CS) is an innovative signal acquisition and reconstruction theory that allows for the capture of signals at a rate significantly below the Nyquist sampling criterion and accurately reconstructs sparse signals or signals that can be sparsely represented from these few samples.

The principle of CS indicates that for a sparse signal, or a signal that can be sparsely represented, it is possible to capture most of the signal information through low-dimensional measurements and to accurately reconstruct the original signal from these measurements. This process can be described by the following mathematical model:

$$\mathbf{y} = \Phi \mathbf{x}, \tag{11}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the original signal, $\Phi \in \mathbb{R}^{m \times n}$ is the measurement matrix, and $\mathbf{y} \in \mathbb{R}^m$ are the measurements with $m \ll n$, meaning that the dimension of measurements is much less than the dimension of the signal.

The sparsity of a signal refers to the number of non-zero elements in a transform domain (like Fourier, wavelet, *etc.*) being few. If the signal itself is not sparse, it can be transformed using a transformation matrix Ψ:

$$\mathbf{s} = \Psi \mathbf{x}, \tag{12}$$

where $\mathbf{s}$ is the sparse representation in the transform domain.

For signal reconstruction, one of the common algorithms is Orthogonal Matching Pursuit (OMP). OMP is an iterative greedy algorithm that seeks to find the set of sparse coefficients that best matches the measurement vector $\mathbf{y}$. The steps of the OMP algorithm are as follows:

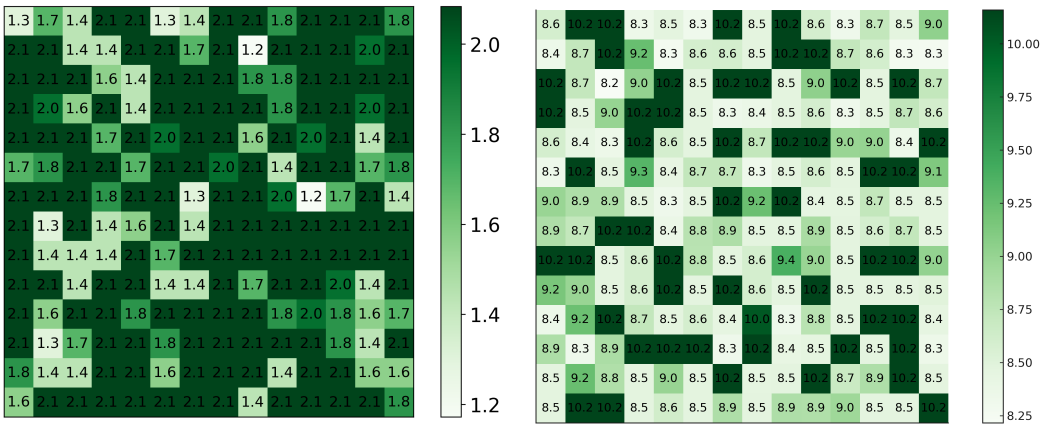
1. Initialize the residual $\mathbf{r}_0 = \mathbf{y}$, support set $S_0 = \emptyset$, and iteration count $k = 0$.
2. Find the column ϕ_j that correlates most significantly with the residual $\mathbf{r}_k$ and update the support set $S_{k+1} = S_k \cup \{j\}$.

3. Update the signal estimate by solving the least squares problem: $\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \|\mathbf{y} - \Phi_{S_{k+1}}\mathbf{x}\|_2$.
4. Update the residual to $\mathbf{r}_{k+1} = \mathbf{y} - \Phi_{S_{k+1}}\mathbf{x}_{k+1}$.
5. If a stopping criterion is met (such as the residual is small enough or a predetermined number of iterations is reached), stop the iteration.
6. Otherwise, increment k and go back to step 2.

Based on the aforementioned methods, we conducted tests on the ImageNet-1k dataset. Specifically, we undersampled the original images at different under-sampling rates using a Gaussian random mask (this process is adapted for DMD computation), then reconstructed the images using the OMP algorithm, and finally completed the classification task through ViT.

Because the CS-method yields poor reconstruction of the original image at low under-sampling rates ($< 20\%$), the subsequent ViT model struggles with the classification task, leading to very poor final results. The contrast between these results and those of LUM-ViT aligns with our logic: the CS-method cannot effectively utilize the prior information of the object being detected, making it unsuitable for low under-sampling rates, whereas deep learning methods can achieve better results by effectively using prior information, maintaining performance even at extremely low under-sampling rates.

C VISULIZATION OF THE LEARNABLE MASK AND ANALYSE



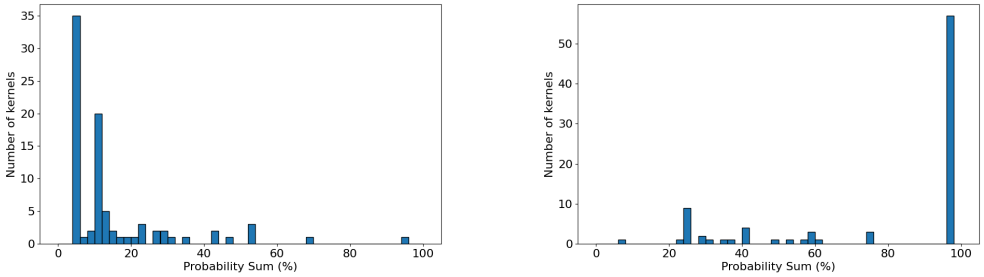
(a)Heatmap for LUM-ViT-2% patches. (b)Heatmap for LUM-ViT-10% patches.

Figure 11: **The passing-throgh probability heatmap of patches at different positions in LUM-ViT.** In figure (b), we truncated the top 5% of the data to allow the image to reveal meaningful information and to prevent the distortion caused by extreme outliers.

In LUM-ViT, the mask is applied to each data point output by the patch-embedding layer, with each data point corresponding to the output of a specific convolutional kernel on a specific patch. The mask filters both kernels and patches: if all positions of patches affected by a kernel are masked, then the kernel is eliminated; similarly, if all kernel positions in a patch are masked, the patch is eliminated.

To explore the selective effect of the learnable mask on different kernel-patch data points, we visualized two models: LUM-ViT-10% and LUM-ViT-2%.

Fig. 11 presents the visualization results for different patch positions, offering an intuitive realization of the macroscopic emphasis of all kernels on different image locations. Overall, we did not observe a pronounced spatial preference, which might be due to the fact that all patches could contain important information across the entire dataset. Comparing the two models, the heatmap of LUM-ViT-10% appears more random, and we observed four extreme outliers with values exceeding 60%, which, if not truncated, would significantly impact the visualization effect.



(a)Histogram for LUM-ViT-2% kernels. (b)Histogram for LUM-ViT-10% kernels.

Figure 12: **Histogram of the number of patches retained by each convolutional kernel in LUM-ViT.** The horizontal axis represents the sum of probabilities for all patch positions in a single kernel to pass through the mask. Due to the selection and elimination of most convolutional kernels in low under-sampling scenarios, the number of data points at position 0 in the histogram is overwhelmingly large. To prevent this from impacting the visual effect, we have excluded the data points within the [0,5] range.

Fig. 12 visualizes the selection probability of different kernels. We calculated the sum of probabilities for all patch positions passing through the mask for each kernel and tallied the number of kernels for different total probability sums. Many kernels are directly discarded at low under-sampling rates, leading to an overwhelming number of points near 0. To avoid impacting the visualization, we removed data within the [0,5] interval. We observed that in LUM-ViT-10%, there is a tendency to select kernels without selecting patches, meaning the model tends to keep or discard all patches under one kernel. This phenomenon disappears in LUM-ViT-2%, where the model tends to allocate different patch positions to different kernels.

Synthesizing the visual results from Fig.11 and Fig.12, we infer that the model tends to select kernels under relatively low under-sampling pressure (*e.g.*, 10%), satisfying under-sampling requirements by retaining only the most useful kernels. When the under-sampling pressure is relatively high (*e.g.*, 2%), selecting kernels alone does not meet the under-sampling rate requirements, thus forcing the model to make selections among different patch positions under different kernels, leading to a collaborative division of labor across different kernels for different patch positions.

D Details on the DMD signal acquisition system

D.1 Detailed overview of DMD

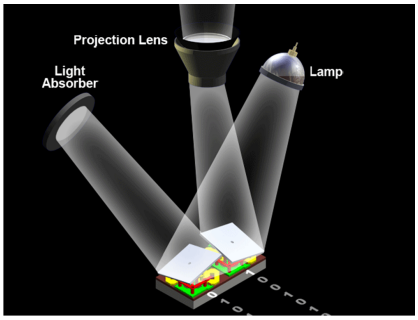
Digital Micromirror Devices (DMDs) are sophisticated optical semiconductor devices that form the core of Digital Light Processing (DLP) technology. A typical DMD chip hosts an array of up to millions of micromirrors, each with dimensions in the order of a few micrometers.

The core functionality of a DMD lies in its ability to modulate light spatially. Each micromirror can be individually tilted by electrostatic forces, typically between $+12$ to -12 degrees, corresponding to on and off states, respectively. This tilt controls whether light is directed towards the projection lens (on) or absorbed by a light trap (off), thereby controlling the brightness of each pixel in the projected image.

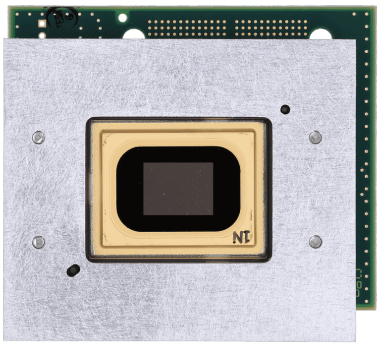
The speed at which these mirrors switch affects the image's refresh rate, with current DMDs capable of switching at speeds of thousands of times per second.

DMDs are integral to various applications due to their high spatial resolution and rapid response time. Common uses include high-definition projectors for cinema and business presentations, where their ability to precisely control light ensures high-quality image projection.

In the field of hyperspectral data processing, DMDs enable the selective reflection of different wavelengths, facilitating the capture of spectral information for each pixel, which is critical for applications in remote sensing and material analysis. Furthermore, the potential of DMDs in optical neural networks is being explored, where they can modulate light in complex patterns to mimic neural connections, potentially leading to advancements in photonic computing and artificial intelligence.



(a)The reflective function of a single micromirror on a DMD chip.



(b)DMD evaluation board.

Figure 13: **DMD chip and its micromirror function.**

D.2 DMD SIGNAL ACQUISITION SYSTEM

In LUM-ViT, the DMD serves as a key component for optical modulation before signal acquisition, termed the DMD signal acquisition system. Its schematic is shown in Fig. 1 in the main text, and its physical representation in Fig. 6, which are not displayed here again.

The DMD signal acquisition system comprises two sets of focusing lenses, a DMD chip, and a single-point light signal detector. During operation, light from the target object is imaged onto the DMD's micromirror surface by Lens Group 1. The DMD modulates the image projected onto it by setting its micromirrors to different reflection states. The modulated image is then reflected towards the optical detector, where Lens Group 2 focuses the light onto the detector, completing signal acquisition. It's worth noting that while the spectrometer here can capture comprehensive spectral information at an extremely high spectral resolution, it only measures the light intensity at a single point at a time, rather than capturing a multi-pixel image in one shot like conventional photographic equipment.

During its operation, the DMD can control the state of its micromirror array at each timestep, modulating the optical signal in various ways. Since the DMD performs spatial optical modulation—simultaneously modulating all spectral channels within its working range with the same mask—it is well-suited for modulating hyperspectral signals.

The DMD is the critical control component of the DMD signal acquisition system, with signal modulation achieved through different masks. In single-pixel cameras based on CS theory, the DMD typically uses random masks following a Gaussian distribution. In LUM-ViT, neural network weights obtained through training are used as masks for the DMD, integrating the DMD signal acquisition system into the neural network computation.

E TRAINING SETTINGS OF THE THREE STAGE OF THE TRAINING PHASE

During the Training Phase, we divided the training process into three stages, using distinct training parameters to train the mask, the kernel-level binary layer, and overall finetuning, respectively. In initial experiments, we attempted to combine these stages, but due to the instability introduced by the learnable mask and binarization, training them together led to difficulty in convergence or suboptimal performance. Therefore, we consider this three-stage training approach necessary.

Regarding the first stage of training, conventional training parameters suffice for mask training. The primary consideration is the initial value of the mask's trainable parameters. We experimented with an all-ones mask (i.e., a non-selective mask) as the initial value, but results were inferior to those with a randomly initialized mask. During training, the undersampling rate quickly drops to the target value within a few epochs and then fluctuates slightly throughout the remaining training process.

The second stage is the most unstable part of the entire training process. Due to the significant difference between binary and full precision, and because LUM-ViT requires binarization of the entire first layer (which affects all subsequent layers' computations), many settings from full-precision

network training are no longer applicable. Specifically, we froze the mask weights from the first stage to prevent changes in the mask from affecting the training of the binary layer. We abandoned most data augmentation strategies and adopted a more sensitive and stable learning rate momentum parameter setting, commonly used in training binary networks.

The third stage is the final finetuning. Since the second stage's training settings are primarily tailored to the binary layer and do not suit the full-precision layers, we need to finetune these layers last with parameters suitable for full-precision training. Specifically, in this stage, we froze the weights of the binary layer and the mask, finetuning only the remaining network layers with conventional parameter settings.

Table 4: Training settings.

config	value		
	Stage 1	**Stage 2**	**Stage 3**
optimizer	AdamW	AdamW	AdamW
optimizer momentum(β_1, β_2)	0.9, 0.999	0.6, 0.9999	0.9, 0.999
base learning rate	5e-4	1e-4	1e-4
weight decay	0.03	5e-5	0.03
batch size	4096	4096	4096
learning rate schedule	cosine decay	cosine decay	cosine decay
warmup epochs	5	1	5
frozen layers	none	mask	bi-conv-kernels, mask
label smoothing	0.1	0.1	0.1
random erase	0.25	0	0.25
mixup	0.8	0	0.8
cutmix	1.0	0	1.0
drop path	0.1	0.1	0.1