

1 **A Proof: Maximizing Transmission Rate is Equivalent to Maximizing H_Q**

2 The rate of transmission is defined according to Shannon's information theory [8] as:

$$R = H_Q - H(y|x), \quad (1)$$

3 where $H(y|x)$ is the conditional entropy of y given x . This can be defined as:

$$H(y|x) = - \sum_{i,j} p(x_i, y_j) \log p(y_j|x_i) \quad (2)$$

$$= - \sum_{i,j} p(x_i) p(y_j|x_i) \log p(y_j|x_i) \quad (3)$$

$$= - \sum_i \left[p(x_i) \sum_j p(y_j|x_i) \log p(y_j|x_i) \right], \quad (4)$$

4 where $p(x_i, y_j)$ is the joint probability of x and y and $p(y_j|x_i)$ is the conditional probability - that is,
5 the probability of the output value being y_j given that we know the input is x_i .

6 Now, for a deterministic model where each input x_i maps to one and only one output y_j , we can have
7 $p(y_j|x_i)$ being 1 for some j and 0 for the others. As a result, the expression $p(y_j|x_i) \log p(y_j|x_i)$
8 is always 0, whether $p(y_j|x_i)$ is 0 or 1. Therefore, we have $H(y|x) = 0$. Consequently, the rate of
9 transmission R simplifies to $R = H_Q$. Hence, maximizing the rate of transmission is equivalent to
10 maximizing H_Q .

11 **B Proof: Cross-Entropy H_{pq} is Equivalent to H_q**

12 Under the condition Eq. (3) and Eq. (4) specified in the main text, we can rewrite the cross-entropy
13 H_{pq} as:

$$H_{pq} = - \sum_x p(x) \log q(x) \quad (5)$$

$$= - \sum_{j=1}^N \sum_{x \in G_j} p(x) \log q(x) \quad (6)$$

$$= - \sum_{j=1}^N \sum_{x \in G_j} p(x) \log \frac{Q(y_j)}{n_j} \quad (7)$$

$$= - \sum_{j=1}^N \log \frac{Q(y_j)}{n_j} \sum_{x \in G_j} p(x) \quad (8)$$

$$= - \sum_{j=1}^N Q(y_j) \log \frac{Q(y_j)}{n_j}. \quad (9)$$

14 Similarly, we can express H_q as:

$$H_q = - \sum_x q(x) \log q(x) \quad (10)$$

$$= - \sum_{j=1}^N \sum_{x \in G_j} q(x) \log q(x) \quad (11)$$

$$= - \sum_{j=1}^N \sum_{x \in G_j} q_j \log q_j \quad (12)$$

$$= - \sum_{j=1}^N Q(y_j) \log \frac{Q(y_j)}{n_j}. \quad (13)$$

15 Comparing the two results, we have

$$H_{pq} = H_q. \quad (14)$$

16 C Proof of Necessary and Sufficient Conditions for Orthogonality of Bases in 17 Two-Pixel Case

18 We want the outputs of the two MLPs to be independent, meaning we cannot predict the state of the
19 second MLP’s output given the first one:

$$Q(y_2|y_1) = Q(y_2). \quad (15)$$

20 Additionally, we want each MLP to partition the input space evenly. This is represented as:

$$Q(y_1) = \frac{1}{N_1}, Q(y_2) = \frac{1}{N_2}, \quad (16)$$

21 where N_1 and N_2 represent the number of output nodes of the two MLPs. Eq. (15) and Eq. (16) leads
22 to

$$Q(y_1, y_2) = \frac{1}{N_1 N_2}. \quad (17)$$

23 Conversely, from Eq. (17), we can derive Eq. (15) and Eq. (16), indicating they are equivalent.
24 Therefore, we conclude that the bases are orthogonal if and only if Eq. (17) is satisfied.

25 D Experimental Details and Supplementary Results for the Two-Pixel Case

26 To ensure that the multi-layer perceptron (MLP) models were versatile enough to approximate
27 complex functions [4], we employed relatively large MLPs with two hidden layers of 200 and 100
28 nodes, respectively. The Rectified Linear Unit (ReLU) is used as the activation function.

29 For training data, we randomly sampled 10 million pairs of horizontally neighboring pixels from the
30 COCO dataset [3]. The models were trained for 20 epochs using Adam optimizer with a learning rate
31 of 0.001. In the experiments involving two MLPs, we used MLPs of the same size and followed the
32 same training procedure as in the single MLP case. For the factor balancing the two terms in the loss
33 function in Eq. (11) and Eq. (13) in the main text, we use $k = 2/3$ in all experiments.

34 The partitions were visualized using the tricontour function from matplotlib, which generates contour
35 plots of unstructured triangular grids. The two pixel values were input as the x and y coordinates,
36 while the outputs of the MLPs were used as the z-values.

37 In addition to Fig. 1 in the main text, some interesting results are shown in Fig. 1.

38 E Experimental Details for the Image Patches Case

39 Two models were discussed in the main text of our study: a model trained on 5×5 color image
40 patches (the color model), and another trained on 4×4 grayscale image patches (the grayscale
41 model).

42 **Grayscale Model:** Our training data consisted of roughly 100,000 images from the unlabeled section
43 of the COCO 2017 image dataset, converted to grayscale. In each training batch, we randomly
44 selected 1,000 images and extracted 1,000 random image patches from each image, leading to a total
45 of 1 million image patches per batch. From this batch, a mini-batch of 500 patches was randomly
46 chosen to calculate the loss function. To ensure numerical stability, a small value $\epsilon = 10^{-38}$ was
47 added when computing the distances between samples. We used the Adam optimizer with a learning
48 rate of $1e-3$, and the model was trained for a single epoch. The sparsity regularization parameter was
49 set to $\alpha = 0.05$.

50 **Color Model:** Training data for the color model were image patches extracted from 1.2 million
51 images in the training portion of ImageNet. Each training batch comprised 500 randomly chosen
52 images, with 100 random image patches extracted from each image. From each batch, a mini-
53 batch of 500 patches was sequentially selected (i.e., patches within each batch were not shuffled) to
54 compute the loss function. To augment the dataset, we randomly flipped images horizontally with

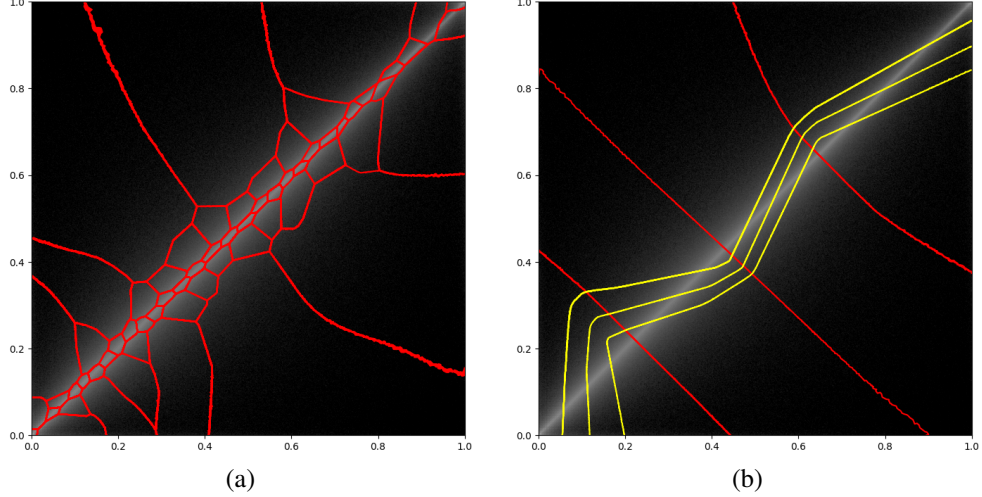


Figure 1: Additional results complementing Fig. 1 in the main text. (a) One basis with 64 independent states. The 2D structure of the even partition is complex, reflecting the high expressiveness of the MLP in modeling the probability space. (b) Two orthogonal bases, each with 4 independent states. The second MLP has learned to employ a zigzag partitioning structure, effectively dividing the total intensity into more sections. This suggests that the model may be compensating for the perceived insufficiency of the first MLP in partitioning the total intensity with only 4 states.

a probability of 0.5. We adopted the node-wise repel form of the loss function to ensure numerical stability (comparable results were achieved with the method used in the grayscale model). The model was trained for 10 epochs using the AdamW optimizer, with a learning rate of $2e-4$ for the first 5 epochs, and $1e-4$ for the remaining epochs. The sparsity regularization parameter was set to $\alpha = \frac{6}{96} = 0.0625$.

F Statistical Analysis of Even Code Representation

In this section, we conduct further statistical analyses on the even code representation using the grayscale model described in Appendix E. Fig. 2 (a) illustrates the proportion of samples activating each unique binary representation. At the most granular level, the distribution is highly uneven, reflecting the image similarity statistics learned by the model, as anticipated. Fig. 2 (b) shows the proportion of samples activating a certain number of output nodes, with the majority activating around 14 out of the 64 total output nodes.

To confirm that samples also have a relatively even distribution at middle scales, We select 30 random occupied positions in the binary representation space and count the total number of samples within varying distances to each. If the distribution is relatively even, the 30 curves should be close to each other (within the same order of magnitude) and exhibit similar shapes. The result is shown in Fig. 2 (c). Most curves are close even at the smallest scale, and they become closer as the scale increases. In Fig. 2 (d) we plot the occupancy rate of positions in the binary representation space at different distances to 30 random occupied positions. The occupancy rate is calculated as the total number of samples at a distance d to a random occupied position, divided by the number of all possible sites in the binary space with a distance d to the same position, which is the combination number $\binom{64}{d}$. A similar decaying trend is observed for all 30 random sites, except at a small scale or at a large d , where the boundary of the subspace is reached.

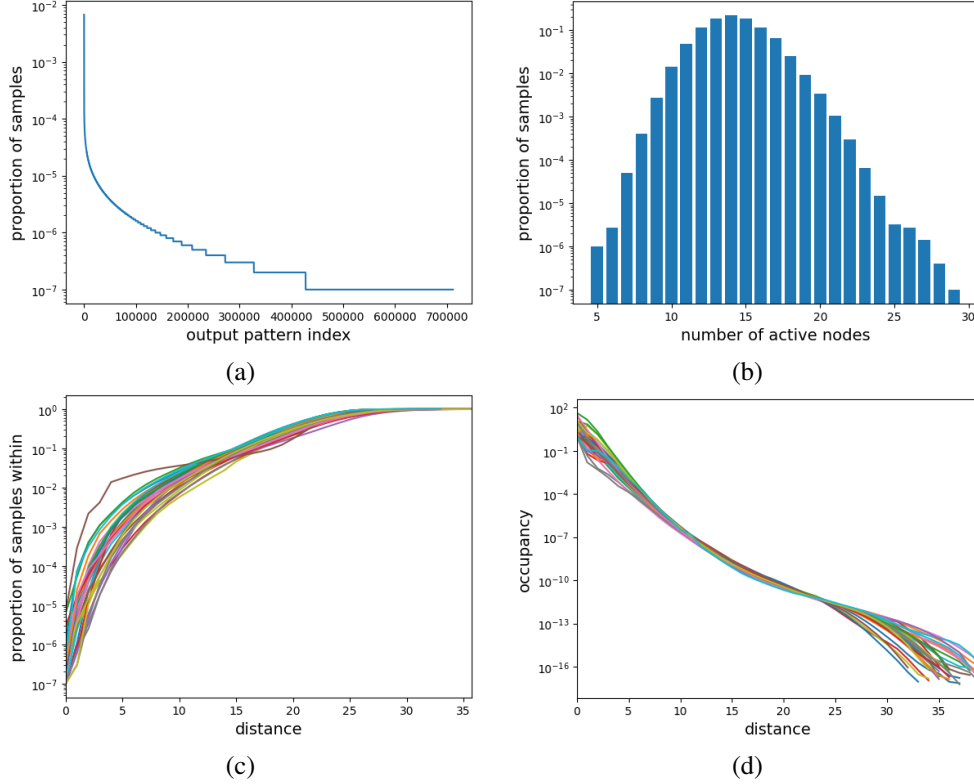


Figure 2: Statistical analysis of the learned representation using the loss function Eq. (15) defined in the main text. (a) Proportion of image patch samples activating each unique representation pattern. (b) Proportion of image patch samples with representation having different numbers of active nodes. (c) Proportion of image patch samples within varying distances to 30 random positions in the binary representation space. (d) Occupancy rate at different distances to 30 random positions in the binary representation space.

78 G Decoding Even Code

79 To intuitively understand how well the binary representation preserves information, we can decode
 80 the binary representations and compare them to the original image patches. The encoder model and
 81 its training is the same with the grayscale model described in Appendix E except here we use color
 82 images and $\alpha = 0.03$.

83 To train the decoder, 10 million random image patches are fed into the encoder to generate corre-
 84 sponding representations. Since the mapping is many-to-one, we average all image patches with the
 85 same binary representation as the training target and use the binary vector as the feature. This results
 86 in approximately half a million feature-target pairs for training the decoder. We use an MLP with the
 87 same size as the encoder to construct the decoder, which is trained with a batch size of 128 for 100
 88 epochs, using the Adam optimizer and a learning rate of 0.001.

89 Once the decoder is trained, we first use the encoder to encode every non-overlapping image patch of
 90 an image. Then, the decoder is used to convert the binary representations back into image patches,
 91 which are then tilted together to get the decoded image. Fig. 3 shows an example of original and
 92 decoded images. While the even code method does not explicitly optimize for image reconstruction
 93 performance in the loss function, unlike many previous image patch modeling methods [1, 2, 6, 7, 5],
 94 it is noteworthy that the binary representation preserves critical information such as luminance, color,
 95 and boundaries effectively. This retention of key details ensures the accurate comprehension of the
 96 image in subsequent processing stages.



Figure 3: Demonstration of decoding the binary representation. (a) The original image. (b) The image reconstructed from decoded image patches, which are titled together. Despite the lack of explicit optimization for image reconstruction, key details such as luminance, color, and boundaries are preserved effectively.

References

- [1] Daniel Lee and Seung Sebastian. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788, oct 1999.
- [2] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Ng. Efficient sparse coding algorithms. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.
- [3] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.
- [4] Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the Number of Linear Regions of Deep Neural Networks. *Advances in Neural Information Processing Systems*, 4(January):2924–2932, feb 2014.
- [5] Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, oct 2015.
- [6] Simon Osindero and Geoffrey Hinton. Modeling image patches with a directed hierarchy of Markov random fields. In *Advances in Neural Information Processing Systems 20 (NIPS’07)*, pages 1121–1128. 2008.
- [7] Marc Aurelio Ranzato and Geoffrey E Hinton. Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images. In *AISTATS ’10*, volume 9, pages 621–628, 2010.
- [8] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, 1948.