BRAIN-INSPIRED L_p-CONVOLUTION BENEFITS LARGE KERNELS AND ALIGNS BETTER WITH VISUAL CORTEX

Jea Kwon[1*], **Sungjun Lim**[2], **Kyungwoo Song**[2†], **C. Justin Lee**[3†]
[1]Max Planck Institute, [2]Yonsei University, [3]Institute for Basic Science
jea.kwon@mpi-sp.org
{lsj9862, kyungwoo.song}@yonsei.ac.kr
cjl@ibs.re.kr

ABSTRACT

Convolutional Neural Networks (CNNs) have profoundly influenced the field of computer vision, drawing significant inspiration from the visual processing mechanisms inherent in the brain. Despite sharing fundamental structural and representational similarities with the biological visual system, differences in local connectivity patterns within CNNs open up an interesting area to explore. In this work, we explore whether integrating biologically observed connectivity patterns can enhance model performance and foster alignment with brain representations. We introduce a novel methodology, termed L_p-convolution, which employs the multivariate p-generalized normal distribution (MPND). We took advantage of MPND's conformational flexibility to carefully bridge disparities between artificial and biological connectivity patterns by designing an adaptable L_p-masks. L_p-masks finds the optimal conformation through task-dependent adaptation such as distortion, scale, and rotation. This allows L_p-convolution to perform well in tasks that require flexible input field shapes, including not only square-shape but also horizontal and vertical ones. Furthermore, we demonstrate that L_p-convolution with biological constraint which we call Gaussian structured sparsity significantly enhances the performance of historically successful CNNs with large kernels. Lastly, we present that neural representations of CNNs aligns better with the visual cortex when the conformation of L_p-masks is close to a Gaussian distribution, a biologicially closer condition.

1 INTRODUCTION

The rise of Vision Transformers (ViTs) has revolutionized the field of computer vision, outperforming traditional Convolutional Neural Networks (CNNs) on many tasks and establishing new performance standards (Dosovitskiy et al., 2020; Liu et al., 2021b; Touvron et al., 2021). This shift to ViTs, however, demands a significant increase in model parameters, larger datasets, and extended training periods (Maurício et al., 2023). Under constraints of time and resources, CNNs remain a more practical choice, especially as **CNNs often perform better on smaller datasets** (Liu et al., 2021a; Zhu et al., 2023). This advantage is partly due to CNNs' inherent design that mirrors biological visual systems like the primary visual cortex (V1) in the brain (Hubel & Wiesel, 1962; 1965; Fukushima, 1980), which serves as strong inductive biases, such as hierarchical structures, local feature learning, and parameter sharing, which contribute to their effective generalization capabilities (LeCun et al., 1989; Bartunov et al., 2018; Pogodin et al., 2021).

The findings of visual information processing in the brain have greatly influenced the development of CNNs, providing a reciprocal platform for understanding neural visual mechanisms (Hassabis et al., 2017; Zador et al., 2023; Yang & Wang, 2020; Lindsay, 2021). However, the significant architectural differences between the biological brain and modern computers present engineering challenges that make integrating biological insights into CNNs impractical and often inefficient for problem-solving (Marković et al., 2020). Thus, it is important to integrate brain-inspired concepts in a manner that is compatible with existing CNN architectures. In this study, we identify connectivity

[*]This work was conducted at the Institute for Basic Science. [†]Co-corresponding authors.

patterns as a promising area for innovation. While CNNs typically feature rectangular, dense, and uniformly distributed connections, the brain's visual area V1 exhibits circular, sparse, and normally distributed connections (Lerma-Usabiaga et al., 2021; Seeman et al., 2018; Hage et al., 2022; Rossi et al., 2020) (See Appendix A.3 for the actual connectivity pattern of V1 neurons). Exploring these **distinct connectivity patterns could introduce novel inductive bias** that significantly enhance the performance and efficiency of CNNs, potentially transforming their design and application.

Traditionally successful CNNs have relied on stacking small 3x3 kernels beyond the initial layer (He et al., 2016; Simonyan & Zisserman, 2014), as simply increasing the kernel size does not necessarily enhance performance, which we call **large kernel problem**, despite its increased trainable parameters (Peng et al., 2017) (See Table 1 Base vs Large). However, modern CNNs that achieve performance comparable to ViTs have been adopting significantly larger kernel sizes, ranging from 7x7 to 50x50 (Liu et al., 2022; Ding et al., 2022; Liu et al., 2023). This shift indicates new possibilities and directions in CNN design.

In this paper, we ask interesting question that could mutually captivate both neuroscience and machine learning communities: **Would introducing biological connectivity pattern as a novel inductive bias into a CNN resolve large kernel problem and aligns better with brain's representations?** To answer this question, we introduce L_p-convolution, a novel approach that leverages the multivariate p-generalized normal distribution (MPND) to address the disparities between biological and artificial connectivity patterns (Fig. 1). Through channel-wise trainable L_p-masks in convolutional layers (Fig. 2), we explore their conformational adaptability (Fig. 4 and 5), resulting in enhanced performance in large kernel CNNs and improved alignment with biological representations (Fig. 6 and Table 1). Code, datasets, and pre-trained models are available at `https://github.com/jeakwon/lpconv/`.

2 Bridging disparity between biological and artificial connectivity patterns

Standard CNN architectures are typically designed with rectangular, dense, and uniformly distributed connections (LeCun et al., 1998; Krizhevsky et al., 2012; Simonyan & Zisserman, 2014; He et al., 2015), in contrast to the circular, sparse, and normally distributed connections commonly observed in biological neuron (Lerma-Usabiaga et al., 2021; Seeman et al., 2018; Hage et al., 2022). Early studies in biological modeling using CNNs have shown that task-specific adaptations can lead to sparse weight patterns (Maheswaranathan et al., 2018; Tanaka et al., 2019; Lindsey et al., 2019; Yan et al., 2020; Zheng et al., 2021). These insights demonstrate the adaptability of CNNs and highlight the potential for bridging artificial and biological connectivity patterns. To address this, we analyzed the functional connectivity patterns of both biological and artificial systems, by introducing the MPND (Goodman & Kotz, 1973). MPND is the key of our paper to bridge the conformational difference between biological and artificial connectivity patterns, or local receptive fields (RFs). Note, While the term 'receptive field' can sometimes include sensory-level inputs and multi-layer interactions in a broader context, in this work, we specifically define the **RF as a local connectivity pattern** between neurons in immediately adjacent layers (See detail in Appendix A.20).

Multivariate p-generalized normal distribution Let $\mathbf{s}$ represent the d-dimensional random vectors indicating specific points within RF. s_0 is the receptive center with d-dimensional vector of fixed constants. The relative offset is given by $\Delta\mathbf{s} = \mathbf{s} - s_0$. Introducing MPND (Goodman & Kotz, 1973), we define the RF using as probability density function (PDF) of $\mathbf{s}$ as following:

$$\beta \exp\left(-\|\mathbf{C}\Delta\mathbf{s}\|_p^p\right), \tag{1}$$

where $\mathbf{C}$ is $d \times d$ inverse of covariance matrix, $\|\cdot\|_p^p$ denotes the L_p-norm raised to the p-th power, β is normalization factor[1], and $p \geq 1$. In Figure 1e, we show some examples of MPNDs with varying values of p and $\mathbf{C}$. [2]

[1] $\beta = [(2\Gamma(1+p^{-1}))^d \cdot \det(\mathbf{C})]^{-1}$ where Γ is gamma function, and $\det(\cdot)$ denotes the determinant.

[2] When we optimize for MPND, $\mathbf{C}$ is initialized with $\begin{bmatrix} 1/\sigma & 0 \\ 0 & 1/\sigma \end{bmatrix}$ where σ determines the scale.

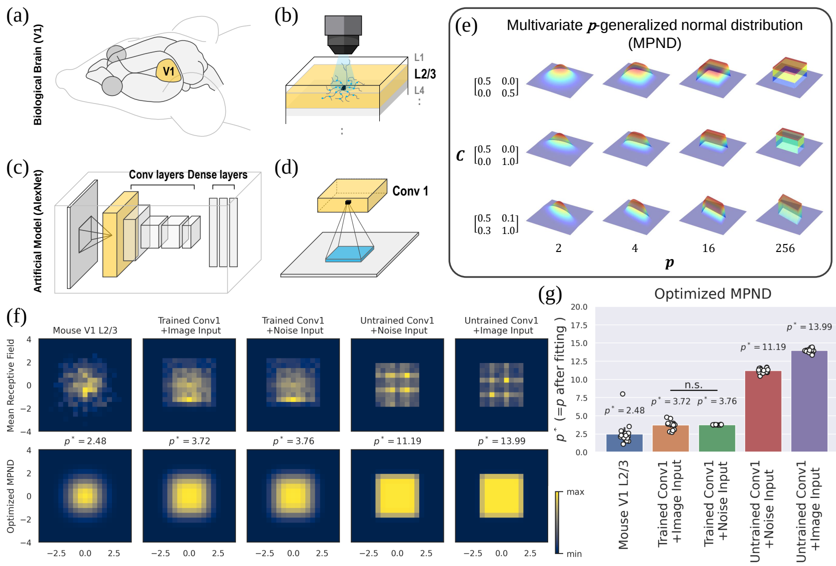

Figure 1: **Local receptive fields from biological and artificial systems can be mathematically reconciled by introducing multivariate p-generalized normal distribution** (a-b) Illustration of receptive fields in V1 of mouse brain (a) at layer 2/3 (b). (c-d) Illustration of receptive fields in AlexNet (c) at Conv1 layer (d). (e) Shapes of MPND with varying parameters of $\mathbf{C}$ and p. (f) Top, visualization of mean receptive fields of functional synapses in mouse V1 layer 2/3 (column 1) and AlexNet Conv1 with varying conditions (columns 2-5). Bottom, optimized MPND over receptive fields shown in the first row. (g) p after MPND optimization; Using Welch's t-test with Holm-Bonferroni's multiple comparisons correction, all possible combinations between groups were statistically significant (p-value<0.05) except for 'n.s.' (non-significant) denoted in the figure; n=17 for all conditions. We optimized MPND parameters of p and σ, where $\mathbf{C} = \begin{bmatrix} 1/\sigma & 0 \\ 0 & 1/\sigma \end{bmatrix}$, $\sigma_{\text{init}} = 0.5$, and $p_{\text{init}} = 2$

Constructing biological and artificial RFs from the functional synapses To compare the biological and artificial RFs, we first prepared the 2D offsets of functional synapses relative to the post-synaptic units in both systems: mouse V1 L2/3 and AlexNet Conv1 (Fig. 1a-d; See details in Appendix A.1 and A.2). To match the scale difference between the two systems, we standardized the relative offsets with zero mean and unit variance. For the artificial system, we prepared 4 different cases, using both ImageNet-1k pre-trained and randomly-initialized AlexNet[3] with inputs of noises or images (See inputs and corresponding RFs in Appendix A.4). We constructed 2D probability mass functions (PMFs) from the collected offsets, which we call biological or artificial RFs.

MPND effectively models both biological and artificial RFs For the comparison of biological and artificial RFs, we optimized parameters of p and σ in MPND (Fig. 1e, Eq. 1) over PMFs of biological or artificial RFs (Fig. 1f and g). We show that optimized p^* of functional synaptic input patterns of biological neurons were optimized at near 2 (Gaussian-distributed; See details in Appendix A.3). In contrast, the local RF pattern of pre-trained AlexNet's Conv1 was optimized at the range of $3.7 \sim 3.8$, and the untrained one was optimized at the range of $11 \sim 14$ (where input types were less effective). An intriguing observation is the decrease in the value of p for the pre-trained AlexNet's first convolutional layer (Conv1), bringing it closer to the biological RF. Based on these findings, we propose that both biological and artificial RFs can be effectively modeled with MPND. Given these findings, we propose to consider that the value of p close to 2 is indicative of biological RF, while a higher p represents RFs to be more artificial.

3 L_p-CONVOLUTION: INTRODUCING MPND IN THE CONVOLUTION

In the early stages of CNN development, large kernels were not widely adopted, with their use predominantly confined to the initial layers (Krizhevsky et al., 2012; Szegedy et al., 2015; 2017).

[3]For clarity, we refer to *trained* as pre-trained models and to *untrained* as randomly-initialized models.

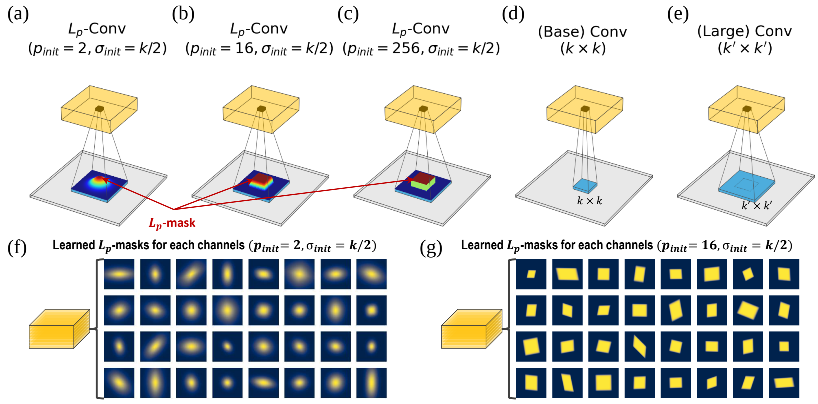

Figure 2: **Graphical illustration of L_p-Conv layers and visualization of learned L_p-masks after traning** (a-c) L_p-Conv layers with $\sigma_{\text{init}} = k/2$ and varying $p_{\text{init}} = \{2, 16, 256\}$; L_p-masks overlaid on kernels (red arrows). (d-e) Conventional Conv layers with the kernel size of $k \times k$ (Base) or $k' \times k'$ (Large). (f-g) Visualized 32 example learned L_p-masks from L_p-converted AlexNet Conv1 after training with Tiny Imagenet dataset

Attempts to enlarge kernel size sometimes led to a decline in classification performance (Peng et al., 2017). We confirmed this in Table 1 Conv (Base) vs Conv (Large), simply enlarging the kernel sizes of previously successful CNNs such as AlexNet, VGG, or ConvNext showed a marked drop in performance. Consequently, the more favored strategy was stacking smaller kernels (1x1, 3x3) (Simonyan & Zisserman, 2014; He et al., 2016).

Based on our observation in Figure 1 artificial RFs become closer to biological RFs with training (decrease in p of MPND by ImageNet-1k training in AlexNet Conv1), we asked two intriguing questions: 1) Can introducing brain-inspired concept of flexible RF provide a solution to the of large kernel problem in CNN? 2) Can CNNs with RFs close to biological ones align better with the representation of the brain? To answer these questions, we introduce L_p**-convolution**: overlaying channel-wise trainable L_p**-masks** onto the kernels of CNNs.

L_p**-convolution** Here, we propose the L_p-convolution, which is compatible with various convolutions. We formulate the L_p-convolution based on the MPND in the convolutional layer by employing channel-wise L_p-masks, which are overlaid on convolutional filters (Fig. 2a-c).

First, we define the relative height and width offsets, $\Delta\mathcal{S} \in \mathbb{R}^{2\times K_h \times K_w}$ from the kernel center, $(K_h/2, K_w/2)$, as follows

$$\Delta\mathcal{S}_{\cdot,h,w} = (\Delta h, \Delta w)^T = (h - \frac{K_h}{2}, w - \frac{K_w}{2})^T$$

for $h \in [0, K_h - 1], w \in [0, K_w - 1]$, where K_h and K_w denote the kernel height and kernel width, respectively. Empirically, we utilize the normalized value between 0 and 1. It is noted that $\mathcal{S}_{\cdot,h,w} \in \mathbb{R}^2$ denotes all values corresponding to h-th height and w-th width.

Second, we propose the L_p**-mask**, structured mask matrix, derived from the offset and MPND. Our L_p-mask, $\mathcal{M} \in [0, 1]^{C_o \times K_h \times K_w}$ for all output channel C_o, is a soft mask that is proportional to Eq. 1 without a normalization factor β as following

$$\mathcal{M}_{o,h,w} = \exp\left(-\|\mathcal{C}_{o,\cdot,\cdot}\Delta\mathcal{S}_{\cdot,h,w}\|_p^p\right), \tag{2}$$

where $\mathcal{C} \in \mathbb{R}^{C_o \times 2 \times 2}$ is the set of 2×2 covariance matrix for each output channel. In other words, L_p-mask calculates the soft mask $\mathcal{M}_{o,\cdot,\cdot}$for each o-th output channel independently, and each soft mask handles the positional correlation, height and width position, from the offset, $\Delta\mathcal{S}$. Our L_p-mask in Eq. 2 corresponds to RF in Eq. 1, when $K_h = K_W = C_o = 1$.

Third, we propose the L_p-convolution by applying L_p-mask into the convolutional weights $\mathcal{W} \in R^{C_i \times C_o \times K_h \times K_w}$, where C_i is the number of input channel. For each i-th input channel $\mathcal{X}_i$ and convolutional filter weights $\mathcal{W}_i$, we formulate the corresponding convolution output $\mathcal{Y}_i$ as

$$\mathcal{Y}_i = \phi(\mathcal{X}_i * (\mathcal{W}_i \odot \mathcal{M})), \tag{3}$$

where ϕ is non-linear function and $*$ is the convolution operation.

We note that $\mathcal{C}$ and p are trainable parameters[4]. To ensure the positive definite property of $\mathcal{C}_o$ and satisfy the L_p-norm property with $p \geq 1$, we can employ Cholesky decomposition for $\mathcal{C}$ and value clipping for p.

As shown in Equation 3, our L_p-convolution is a generalized version of the traditional convolution. In our settings, it is noted that L_p mask converges to a binary mask as p approaches infinity. Empirically, L_p mask becomes a binary mask for sufficiently large p, as shown in Fig. 1. If all elements of L_p mask equal one, then L_p-convolution degrades to the traditional convolution. In this situation, both traditional convolution and our L_p convolution have square-shaped RFs for each layer. In other words, our L_p-convolution takes task or data-dependent RFs with varying $\mathcal{M}$ by optimizing the $\mathcal{C}$ and p as shown in Figure 2. Therefore, L_p-convolution has task-specific RFs with varying distortion, scale, and rotation levels.

In practical terms, we replaced all existing Conv2d layers in the baseline CNN model with L_p-Conv layers by applying a function called **LpConvert** to the baseline CNN model (See pseudo-code in Appendix A.21). To provide further insight into the conformational changes of L_p-masks during model training, we present examples of 32 random L_p-masks from Conv1 of an AlexNet model trained with TinyImageNet (f and g in Fig. 2).

4 L_p-CONVOLUTION BENEFITS LARGE KERNEL CNNS

L_p-convolution on traditional models To test whether L_p-convolution is robustly applicable to traditional CNN architectures, we conducted vision classification tasks (See detailed experimental settings in Appendix A.9) using the CIFAR-100 and TinyImageNet datasets on models of AlexNet, VGG-16, ResNet-18, ResNet-34, and the ConvNext-tiny (Liu et al., 2022). As can be seen in the Table 1, applying L_p-Masks with approximately double the kernel size generally improves classification performance, whereas simple kernel enlargement fails to achieve similar improvements (highlighted in red). Furthermore, we found that the optimal choice of the hyperparameter p_{init} varies depending on the model architecture [5]. However, the best performance was most often observed when $p_{\text{init}} = 2$ (which is close to biological RFs), with 9 out of 10 cases achieving either the top or second-best results. These results suggest that explicitly applying Gaussian-like structured sparsity to large kernels during the convolution operation may lead to better optimization.

Biologically-inspired L_p-convolution improves robustness To assess the robustness of Lp-Models, we utilized CIFAR-100-C, a dataset specifically designed for evaluating robustness using corrupted validation data (Hendrycks & Dietterich, 2019). We focused on data corruptions at severity level 1 across all tested architectures (see the attached PDF, Figure 1, for all architectures). Our findings reveal that Lp-Convolution of $L_{p=2}$ significantly enhances robustness, as demonstrated in Table 2.

L_p-convolution on modern large kernel CNN Next, we explored the impact of L_p-Convolution on modern large kernel CNNs like RepLKNet (Ding et al., 2022), which utilizes kernels up to size 31×31. By integrating L_p-convolution into these models, we aimed to investigate whether our L_p-convolution method can enhance the performance of large kernel CNNs. Specifically, we conducted experiments with RepLKNet using and without L_p-Convolution. For the Lp2-RepLKNet model, all Conv2d layers were replaced with Lp-Conv ($p = 2$) layers without modifying the kernel sizes. As can be seen in Table 4, L_p-Convolution achieves a performance improvement with little to no increase in computational cost even in modern large kernel CNN architecture. This indicates that Lp-Convolution can be easily incorporated as a flexible option for researchers aiming to optimize model architectures.

[4] C and p are updated with the standard backpropagation process. L_p-mask, $\mathcal{M}$, is dynamically generated during forward process using C and p.

[5] Note that although p is a parameter that can be adjusted during training, it was not fine-tuned to the point where it overlaps with the initial settings of each p_{init} condition (See Appendix A.16).

Table 1: Top-1 performance on the CIFAR-100 and TinyImageNet datasets in CNNs are reported with 5 trials (mean±std). The symbol † indicates that both $\mathcal{C}$ and p are frozen parameters during training. $k \times k$, default kernel size. $l \times l$, large kernel size. $l = 2 \times \lceil \frac{k}{2} \rceil + k (\approx 2k)$. For all L_p-Conv layers, $\mathcal{C}$ was initialized with $1/\sigma_{\text{init}}$ of diagonals and 0 of off-diagonals, where $\sigma_{\text{init}} = k/2$, where k represents the default kernel size in each layer of the baseline CNN. Statistical comparison results using Welch's t-test with the base model are marked as follows: 'ns' (p-value ≥ 0.05), '*' ($0.01 \leq$ p-value < 0.05), '**' ($0.001 \leq$ p-value < 0.01), and '***' (p-value < 0.001). The text in bold denotes the best performance, while underlined signifies the second best. Gray indicates a baseline performance and red indicates a decrease in performance.

			CIFAR-100				
Layer	Kernel	p_{init}	**AlexNet**	**VGG-16**	**ResNet-18**	**ResNet-34**	**ConvNeXt-T**
(Base) Conv	$k \times k$	-	66.05 ± 0.33	70.26 ± 0.29	71.22 ± 0.18	72.47 ± 0.23	58.36 ± 6.48
(Large) Conv	$l \times l$	-	***54.53 ± 0.65	**64.82 ± 2.92	***72.80 ± 0.27	***73.52 ± 0.11	^{ns}54.13 ± 1.14
$^{\dagger}L_p$-Conv		256	^{ns}65.95 ± 0.32	**71.03 ± 0.38	^{ns}71.24 ± 0.23	^{ns}72.61 ± 0.27	^{ns}60.34 ± 2.80
L_p-Conv		16	****67.12 ± 0.37**	**70.87 ± 0.23	***72.35 ± 0.30	***73.32 ± 0.23	^{ns}61.30 ± 1.71
L_p-Conv	$l \times l$	8	**66.85 ± 0.18	**71.14 ± 0.29	***72.26 ± 0.28	***73.37 ± 0.15	^{ns}59.94 ± 5.04
L_p-Conv		4	*66.68 ± 0.28	***71.71 ± 0.36	***73.00 ± 0.15	***74.07 ± 0.22	^{ns}59.34 ± 7.53
L_p-Conv		2	^{ns}66.13 ± 0.33	*****72.88 ± 0.30**	*****73.86 ± 0.14**	*****74.95 ± 0.11**	^{ns}**62.61 ± 3.03**
			TinyImageNet				
Layer	Kernel	p_{init}	**AlexNet**	**VGG-16**	**ResNet-18**	**ResNet-34**	**ConvNeXt-T**
(Base) Conv	$k \times k$	-	52.25 ± 0.35	67.75 ± 0.07	66.63 ± 0.51	69.22 ± 0.11	70.25 ± 0.45
(Large) Conv	$l \times l$	-	***35.52 ± 0.46	^{ns}66.96 ± 1.50	***68.33 ± 0.19	^{ns}69.46 ± 0.36	^{ns}68.66 ± 1.50
$^{\dagger}L_p$-Conv		256	^{ns}52.60 ± 0.12	^{ns}67.72 ± 0.18	^{ns}66.37 ± 0.55	^{ns}69.27 ± 0.27	^{ns}70.45 ± 0.44
L_p-Conv		16	***53.98 ± 0.50	***69.29 ± 0.25	**67.72 ± 0.43	**70.00 ± 0.33	^{ns}70.62 ± 0.30
L_p-Conv	$l \times l$	8	**54.07 ± 0.91	***69.72 ± 0.16	*67.63 ± 0.45	***69.81 ± 0.23	^{ns}70.52 ± 0.36
L_p-Conv		4	*****54.30 ± 0.48**	***69.79 ± 0.30	**68.20 ± 0.50	**69.99 ± 0.44	^{ns}**70.74 ± 0.37**
L_p-Conv		2	***54.13 ± 0.53	*****69.96 ± 0.45**	*****68.45 ± 0.36**	*****70.43 ± 0.24**	^{ns}70.72 ± 0.31

Table 2: sSummarized robustness experiments on various Lp-CNNs. The table shows the number of wins for each method across different architectures.

Robustness	**Base**	**Large**	$L_{p=2}$	$L_{p=4}$	$L_{p=8}$	$L_{p=16}$
ConvNeXt-T	0	1	**18**	0	0	0
VGG-16	0	0	**19**	0	0	0
AlexNet	0	1	1	4	**13**	0
ResNet-18	0	**13**	6	0	0	0
ResNet-34	0	6	**13**	0	0	0
Win Counts	0	21	**57**	4	13	0

Table 3: Effect of L_p-Convolution on large kernel CNN. RepLKNet was trained on TinyImageNet for 150 epochs with an architecture of [2,2,6,2] blocks and [64,128,256,512] channels. Presented as mean(std).

Model	**Kernel sizes**	**Top-1 Acc. (%)**	**Param (M)**	**Flops (G)**
RepLKNet	[31-29-27-13]	66.2 (0.38)	11.65	2.54
Lp2-RepLKNet	[31-29-27-13]	**67.1 (0.35)**	11.79	2.54

Transfer learning with L_p-convolution on pretrained models There are countless pretrained CNN models in existence that have achieved state-of-the-art performance using large-scale datasets. Actively reusing these models can offer significant time and economic advantages. Theoretically, an L_p-convolution layer with L_p-masks frozen $\mathbf{C}_{\text{init}}$ and $p_{\text{init}} = \infty$ can function identically to the original base model. By employing pretrained weights for the central parameters of the CNN's convolution layer and initializing surrounding weights to zero in the enlarged kernel while using a high p value, it is possible to train the L_p-mask without significantly deviating from the model's original performance (Fig. 3).

To investigate this concept, we performed transfer learning experiments using the ImageNet-pretrained ConvNeXt-V2 Tiny model (Woo et al., 2023) across five distinct datasets, Oxford flowers (Nilsback & Zisserman, 2008), DescribableTextures (dtd) (Cimpoi et al., 2014), Oxford Pets (Parkhi et al., 2012), FGVC Aircraft (Maji et al., 2013), and UCF101 (Soomro et al., 2012). For a fair comparison, we conducted experiments with identical hyperparameters for each convolution layer. All datasets were trained with 16-shot learning, and the models were evaluated after training for 100 epochs without applying early stopping. As indicated in Table 4, implementing L_p-masks with high values of $p = 16$ enhances transfer learning performance beyond that of the pretrained baseline model, whereas simply increasing the kernel size results in a significant decline in performance. Our results demonstrate that L_p-convolution can be effectively integrated with existing successful pretrained models to push performance beyond previous level.

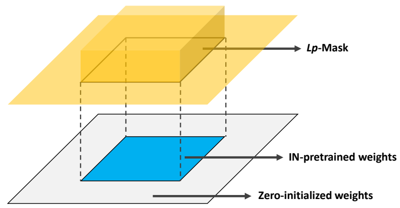

Figure 3: **Schematic illustration of transfer learning setup with L_p-Convolution** As described in the figure, the pretrained weights were surrounded by zero-initialized parameters, and an L_p-Mask (ranges 0 to 1) is overlaid. This configuration helps to start without significantly deviating from the original model's performance.

Table 4: The result of ConvNeXt-V2 pretrained ImageNet 1K for transfer learning with Lp-Convolution using 5 different datasets. We run 16-shot transfer learning and average accuracy over 5 different trials. Avg stands for the average performance across five datasets. Win count indicates the number of dataset that the convolution layer achieved. Underline: improvement over baseline model. **Bold**, best results among all different models.

	Oxford flowers	dtd	Oxford Pets	FGVC Aircraft	UCF101	Avg	Win count
Conv (Base)	95.84	61.12	90.43	**55.03**	65.98	73.68	1
Conv (Large)	51.57	8.22	11.33	4.63	20.48	19.24	0
Lp-Conv (p=2)	95.59	58.78	89.36	54.25	63.92	72.38	0
Lp-Conv (p=4)	95.51	58.20	89.16	43.34	63.94	70.03	0
Lp-Conv (p=8)	95.33	60.53	90.64	54.37	66.16	73.41	0
Lp-Conv (p=16)	**95.92**	**61.71**	**90.71**	54.92	**66.43**	**73.94**	**4**

5 Conformational adaptability of L_p-masks in Sudoku challenge

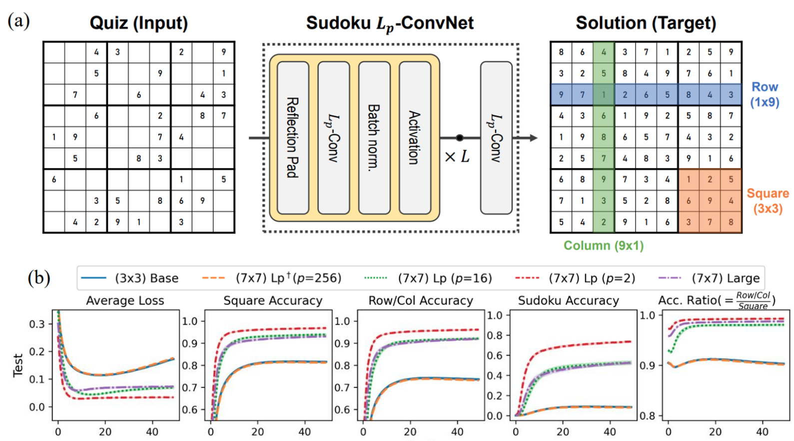

Figure 4: **L_p-convolution enhances Sudoku solving efficiency by effectively balancing accuracy between square and row-column puzzles** (a) Graphical illustration of Sudoku quiz and Sudoku L_p-ConvNet; (left), example Sudoku quiz as an input; (middle), basic block repeated L times ($L = 10$, yellow) contains sequential layers of 1) reflection padding, 2) L_p-Conv, 3) batch normalization, and 4) activation layers; (right), the example Sudoku solution as a target. In Sudoku, a 9×9 square must be filled in with numbers from 1-9 with no repeated numbers in 9x1 rows (blue), 1x9 columns (green), or 3x3 squares (orange). (b) Loss and accuracy curves during test session. '(3×3)' or '(7×7)' denotes the size kernel. '$L_p^\dagger$' denotes parameters of L_p-mask is frozen. 'Large' denotes a simple enlargement of the kernel, without a mask.

Our experimental results so far indicate that a trainable L_p-Mask benefits large kernel optimization. To gain deeper mechanistic insight into this effect, we conducted a specialized experiment, 9×9 Sudoku-solving task (Park, 2018; Oinar, 2021; Amos & Kolter, 2017; Palm et al., 2018; Wang et al., 2019) which aims to solve multiple goals simultaneously (See experimental details in Appendix A.5). As Sudoku challenge necessitates simultaneously achieving three objectives—ensuring every row, column, and box contains all numbers from 1 to 9—(Fig. 4a), we assume that L_p-Masks can evolve their conformation along with training in a task-dependent manner.

L_p-convolution in Sudoku solving: balancing square and row-column imbalances We show that introducing L_p-convolution alleviates the imbalance between square accuracy and row-column accuracy. In Figure 4b, the (3×3) Base model or (7×7) $L_p^\dagger(p_{\text{init}} = 256)$ model exhibited an imbalance in Square-to-Row/Column accuracy and showed signs of overfitting after approximately 15 epochs. Next, we tested two trainable L_p-masks with $p_{\text{init}} = 2, 16$, which resemble a biological RF ($p = 2$) and an artificial RF in ($p = 16$), respectively. In (7×7) $L_p(p_{\text{init}} = 2)$ model, we observed Square-to-Row/Column accuracy become more balanced, resulting in remarkable improvement in overall Sudoku accuracy (red in Fig. 4b). We speculated that this alleviation of Square-to-Row/Column accuracy imbalance in (7×7) $L_p(p_{\text{init}} = 2)$ could be attributed to the task-dependent adaptation of L_p-masks' conformation. To test this possibility, we have designed ablation experiments on (7×7) $L_p(p_{\text{init}} = 2)$ model.

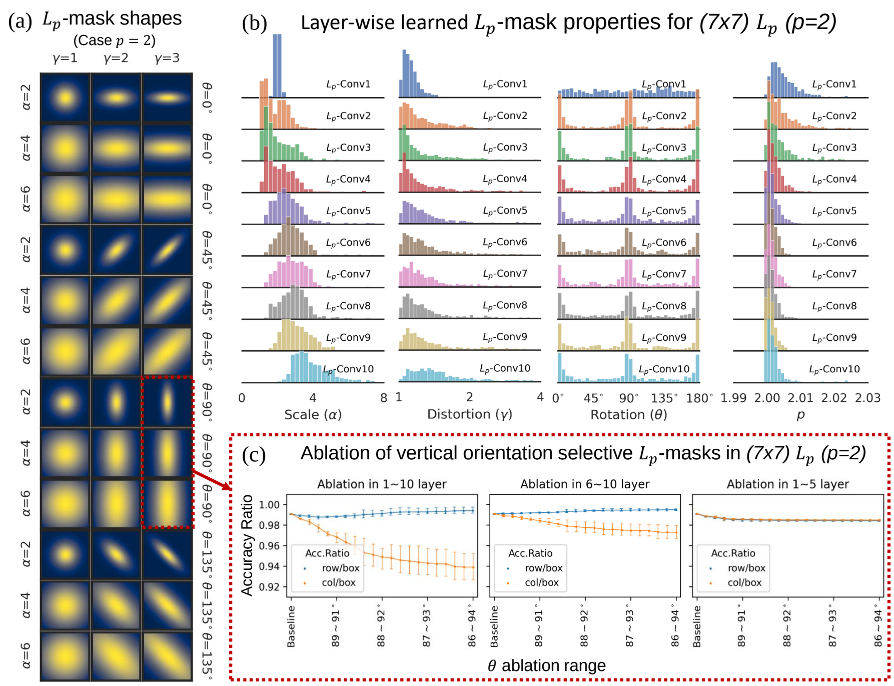

Figure 5: **Task-dependent conformational adaptation of L_p-masks** (a) The shapes of L_p-masks when $p = 2$ and varying properties of scale (α), distortion (γ), and rotation (θ), which are derived from the singular value decomposition of $\mathcal{C}$; Red box indicates column selective L_p-masks which are ablation targets in (c). (b) Layer-wise distribution of learned L_p-mask properties. (c) Selectively ablation of L_p-masks near 90° by gradually increasing the ablation; Ablation in all 10 L_p-conv layers (left), first 5 layers (middle), and last 5 layers (right) respectively.

Ablation of orientation selective masks reveals L_p-masks' conformational adaptability Contrary to a previous large-kernel model that introduces unstructured sparsity directly into filters (Liu et al., 2023), L_p-convolution with $p = 2$ introduces structured sparsity based on a Gaussian distribution. This approach facilitates covariance analysis of the Gaussian distribution, thereby enhancing interpretability. Using Singular Value Decomposition (SVD) on $\mathcal{C}$, we extracted three interpretable properties of scale (α), rotation (θ), and distortion (γ) (See conformational analysis in Appendix A.5). Figure 5a shows conformations of L_p-masks inverse calculated from α, θ, and γ.

Quantitative analysis of (7×7) $L_p(p_{\text{init}} = 2)$ model revealed an increase in scales when layer deepened (Fig. 5b, see visualization in Appendix A.7), with orientations of horizontal (0,180°) and vertical (90°) directions. This indicates the task-dependent adaptation of L_p-masks, which provide flexible and structured RFs in visual processing. To confirm these orientation-selective masks contribute to the balanced Sudoku-solving task, we conducted an ablation test. We classified masks with high distortion ($\gamma > 3$) as orientation-selective masks. Among these, we selectively ablated near 90° by gradually increasing the range (close in shape with the red dashed box in Fig. 5a) while tracking changes in column and row accuracies. While row and box accuracy exhibited a consistent decrease, column accuracy sharply decreased as the θ range increased (Fig. 5c and Appendix A.8), with this trend was notable in the later layers. Together, these results indicate that the conformational adaptability of L_p-masks enables balanced learning in the Sudoku-solving task, thereby contributing to overall performance enhancement.

6 Employing L_p-Convolution for biological systems

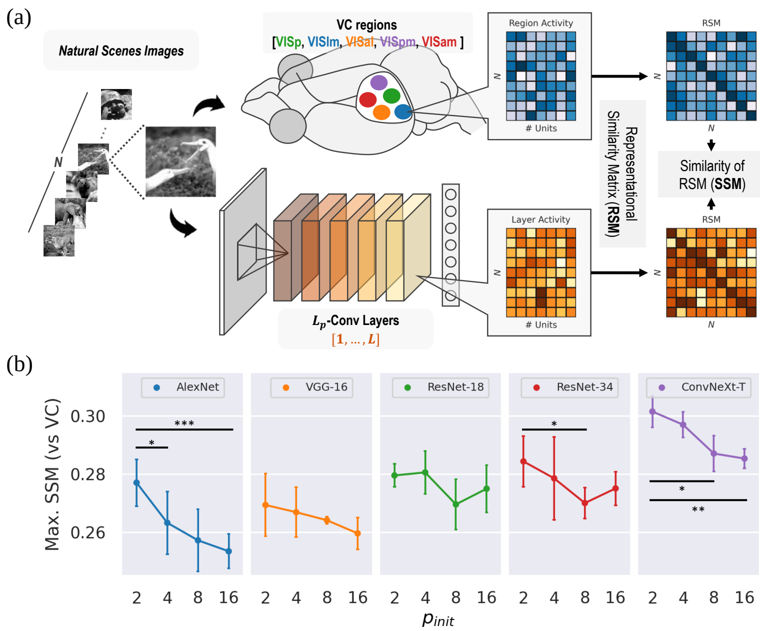

Figure 6: **Representational similarity between biological brain and artificial models using natural images** (a) Schematical illustration of representational similarity analysis (RSA) from neural activities of mouse VC L2/3 subregions and convolutional layers of TinyImageNet-trained CNNs; Unit activities in both mouse brain or CNNs were obtained from N number of image inputs; $N \times N$ RS matrix was constructed for every subregions or layers by measuring the correlations across unit activities. The similarity of the RSMs (SSM) between the V1 region and CNN Conv layer was measured by Kendall's rank correlation coefficient. (b) For the comparison across L_p-models, the maximum SSM was collected among all pair-wise SSM scores across regions and layers; the two-sample Student's t-test for statistical analysis, *: p<0.05, **: p<0.01, ***: p<0.001.

Representational similarity analysis between L_p-CNNs and visual cortex Figure 6 a illustrates our approach to assessing the alignment of representations between biological and artificial systems. We utilized the standardized dataset from Allen Brain Observatory (de Vries et al., 2020), which uses 118 images of Natural Scenes and corresponding neural activities recorded from the mouse visual cortex (VC). Our method builds upon established Representational Similarity Analysis (RSA) techniques (Khaligh-Razavi & Kriegeskorte, 2014; Devereux et al., 2013; Diedrichsen & Kriegeskorte, 2017), to compare the representations of CNNs (Bakhtiari et al., 2021; Shi et al., 2019) (See details in Appendix A.11).

Based on the observation that L_p-convolution tends to perform better as it approaches biologically observed RFs with $p_{\text{init}} = 2$, we posed the question of whether the neural representation of artificial models with biological RFs aligns more closely with the representation of VC. To address this, we compared the neural representations of TinyImageNet-trained CNNs from Table 1 by presenting the 118 Natural Scene images, with the mouse VC representations. To facilitate model comparison, we extracted the representative value, maximum SSM, chosen from pair-wise SSMs across the convolutional layers and the VC subregions (See pair-wise SSMs in Appendix A.12). In the results, models with p_{init} closer to 2 generally exhibited better alignment with the brain (Fig. 6b). In summary, we find that L_p-convolution tends to achieve better alignment with the brain as it approximates a Gaussian distribution.

Neural activity prediction of V1 with L_p-CNNs To closely relate to computational neuroscience, neural activity prediction is frequently used alongside representational similarity analysis. Using the recent multimodal dataset with V1 neural activities from freely moving mice (Xu et al., 2024), we applied L_p-convolution on the CNN of MMV1 model and compared the effect of different p values. As can be seen in Table 5, L_p-Convolution with $p_{\text{init}} = 2$ gives the best prediction performance, which implies that biologically inspired Gaussian sparsity can also benefit CNN architecture for neural activity prediction.

Table 5: V1 activity prediction with L_p-CNNs. The MMV1 model is a CNN followed by a GRU, where the CNN has 3 Conv layers with kernel sizes of [7-7-7] (see details in (Xu et al., 2024)). For the GRU, we used a sequence length of 4, which corresponds to 192 ms of history. Here, **cc** represents cross-correlation (higher is better), and **MSE** represents mean squared error (lower is better). Presented as mean(std) from $n = 3$ mice.

MMV1	$L_{p=2}$	$L_{p=4}$	$L_{p=8}$	$L_{p=16}$	**Base**	**Large**
cc	0.595 (0.046)	0.571 (0.076)	0.572 (0.071)	0.584 (0.066)	0.585 (0.052)	0.577 (0.048)
MSE	0.0722 (0.012)	0.0828 (0.021)	0.0786 (0.017)	0.0756 (0.018)	0.0746 (0.016)	0.0787 (0.013)

7 Conclusion and Remarks

In this study, we introduced a novel L_p-convolution, based on the MPND, with the objective of narrowing the gap between artificial and biological RFs, and subsequently crafting neural network modules more aligned with biological structures. Brain-inspired L_p-convolution enables the cultivation of diverse-shaped RFs with Gaussian-based structured sparsity, adaptable to various rotations, distortions, and scales, and tailored for specific tasks. Significantly, L_p-convolution showcases its adaptability and compatibility across an extensive spectrum of CNN models, from the conventional to the contemporary, underscoring its proficiency, especially in contexts involving large kernels. We believe our research serves as a noteworthy illustration of the symbiotic relationship between advancements in artificial intelligence and our understanding of neural processes.

The advent of ViTs marked a paradigm shift from traditional CNN models, with the Swin Transformer emphasizing the significance of both attention mechanisms and large receptive fields, thereby renewing interest in large kernel CNNs (Dosovitskiy et al., 2020; Liu et al., 2021b; Vaswani et al., 2021). Recent innovations such as RepLKNet and SLaK have showcased performance comparable to ViTs, highlighting the potential of large kernel CNNs in modern computer vision (Ding et al., 2022; Liu et al., 2023). However, the effectiveness of large kernels in historically successful CNN models remained unexplored until now. In this paper, we succesfuly implemented large-kernel convolution by overlaying trainable masks to the filters with Gaussian-based structured sparsity, for adjustments in receptive fields tailored to specific tasks. Compared to previous unstructured sparsity approach which may necessitate extensive hyperparameter tuning, our trainable masks streamline the optimization process by automatically adjusting key parameters, thereby facilitating the application of large kernel training in both traditional and modern CNN architectures.

In biological systems, both anatomical and functional studies have shown that local connectivity patterns and population receptive fields in the visual system display sparse, circular and Gaussian-like distributions in the early visual cortex (Lerma-Usabiaga et al., 2021; Seeman et al., 2018; Hage et al., 2022). These findings prompted us to investigate whether Gaussian sparsity could serve as a beneficial inductive bias in CNNs. Given the fundamental differences in the hardware architectures between CNNs and the brain, it was suspected that introducing Gaussian sparsity might not be an effective choice from an engineering perspective, and its efficacy may not have been effectively tested previously. In our research, we addressed this gap by experimenting with Gaussian sparsity in CNNs that feature large kernels, and systematically comparing artificial and biological connectivity patterns through the introduction of the MPND model.

ACKNOWLEDGMENTS

This work was supported by the Institute for Basic Science in Korea IBS-R001-D2.

REFERENCES

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.

Shahab Bakhtiari, Patrick Mineault, Timothy Lillicrap, Christopher Pack, and Blake Richards. The functional specialization of visual cortex emerges from training parallel pathways with self-supervised predictive learning. *Advances in Neural Information Processing Systems*, 34:25164–25178, 2021.

Sergey Bartunov, Adam Santoro, Blake Richards, Luke Marris, Geoffrey E Hinton, and Timothy Lillicrap. Assessing the scalability of biologically-motivated deep learning algorithms and architectures. *Advances in neural information processing systems*, 31, 2018.

M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Q RandAugment Le. Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 702–703.

Saskia EJ de Vries, Jerome A Lecoq, Michael A Buice, Peter A Groblewski, Gabriel K Ocker, Michael Oliver, David Feng, Nicholas Cain, Peter Ledochowitsch, Daniel Millman, et al. A large-scale standardized physiological survey reveals functional organization of the mouse visual cortex. *Nature neuroscience*, 23(1):138–151, 2020.

Barry J Devereux, Alex Clarke, Andreas Marouchos, and Lorraine K Tyler. Representational similarity analysis reveals commonalities and differences in the semantic processing of words and objects. *Journal of Neuroscience*, 33(48):18906–18916, 2013.

Jörn Diedrichsen and Nikolaus Kriegeskorte. Representational models: A common framework for understanding encoding, pattern-component, and representational-similarity analysis. *PLoS computational biology*, 13(4):e1005508, 2017.

Jörn Diedrichsen, Eva Berlot, Marieke Mur, Heiko H Schütt, Mahdiyar Shahbazi, and Nikolaus Kriegeskorte. Comparing representational geometries using whitened unbiased-distance-matrix similarity. *arXiv preprint arXiv:2007.02789*, 2020.

Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11963–11975, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

Irwin R Goodman and Samuel Kotz. Multivariate θ-generalized normal distributions. *Journal of Multivariate Analysis*, 3(2):204–219, 1973.

Travis A Hage, Alice Bosma-Moody, Christopher A Baker, Megan B Kratz, Luke Campagnola, Tim Jarsky, Hongkui Zeng, and Gabe J Murphy. Synaptic connectivity to l2/3 of primary visual cortex measured by two-photon optogenetic stimulation. *Elife*, 11:e71103, 2022.

Haldan Keffer Hartline. The response of single optic nerve fibers of the vertebrate eye to illumination of the retina. *American Journal of Physiology-Legacy Content*, 121(2):400–415, 1938.

Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Bernhard Hellwig. A quantitative analysis of the local connectivity between pyramidal neurons in layers 2/3 of the rat visual cortex. *Biological cybernetics*, 82(2):111–121, 2000.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pp. 646–661. Springer, 2016.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *Journal of neurophysiology*, 28(2):229–289, 1965.

Seyed-Mahdi Khaligh-Razavi and Nikolaus Kriegeskorte. Deep supervised, but not unsupervised, models may explain it cortical representation. *PLoS computational biology*, 10(11):e1003915, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pp. 1188–1196. PMLR, 2014.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Garikoitz Lerma-Usabiaga, Jonathan Winawer, and Brian A Wandell. Population receptive field shapes in early visual cortex are nearly circular. *Journal of Neuroscience*, 41(11):2420–2427, 2021.

Grace W Lindsay. Convolutional neural networks as a model of the visual system: Past, present, and future. *Journal of cognitive neuroscience*, 33(10):2017–2031, 2021.

Jack Lindsey, Samuel A Ocko, Surya Ganguli, and Stephane Deny. A unified theory of early visual representations from retina to cortex through anatomically constrained deep cnns. *arXiv preprint arXiv:1901.00945*, 2019.

Tianlong Liu, Shiwei andrs Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Tommi Kärkkainen, Mykola Pechenizkiy, Decebal C Mocanu, and Zhangyang Wang. More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. In *International Conference on Learning Representations, ICLR 2023*, 2023.

Yahui Liu, Enver Sangineto, Wei Bi, Nicu Sebe, Bruno Lepri, and Marco Nadai. Efficient training of visual transformers with small datasets. *Advances in Neural Information Processing Systems*, 34:23818–23830, 2021a.

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

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. *Advances in neural information processing systems*, 29, 2016.

Niru Maheswaranathan, Lane T McIntosh, David B Kastner, Josh B Melander, Luke Brezovec, Aran Nayebi, Julia Wang, Surya Ganguli, Stephen A Baccus, and Stanford University. Deep learning models reveal internal structure and diverse computations in the retina under natural scenes. *BioRxiv*, pp. 340943, 2018.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Danijela Marković, Alice Mizrahi, Damien Querlioz, and Julie Grollier. Physics for neuromorphic computing. *Nature Reviews Physics*, 2(9):499–510, 2020.

David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pp. 416–423. IEEE, 2001.

José Maurício, Inês Domingues, and Jorge Bernardino. Comparing vision transformers and convolutional neural networks for image classification: A literature review. *Applied Sciences*, 13(9): 5521, 2023.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Chingis Oinar. How to solve sudoku with convolutional neural networks (cnn). `https://github.com/chingisooinar/sudoku-solver.pytorch`, 2021.

Adriana Olmos and Frederick AA Kingdom. A biologically inspired algorithm for the recovery of shading and reflectance images. *Perception*, 33(12):1463–1473, 2004.

Bruno Olshausen and K Millman. Learning sparse codes with a mixture-of-gaussians prior. *Advances in neural information processing systems*, 12, 1999.

Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

Rasmus Palm, Ulrich Paquet, and Ole Winther. Recurrent relational networks. *Advances in neural information processing systems*, 31, 2018.

Kyubyong Park. Can convolutional neural networks crack sudoku puzzles? https://github.com/Kyubyong/sudoku, 2018.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4353–4361, 2017.

Roman Pogodin, Yash Mehta, Timothy Lillicrap, and Peter E Latham. Towards biologically plausible convolutional networks. *Advances in Neural Information Processing Systems*, 34:13924–13936, 2021.

L Federico Rossi, Kenneth D Harris, and Matteo Carandini. Spatial connectivity matches direction selectivity in visual cortex. *Nature*, 588(7839):648–652, 2020.

Stephanie C Seeman, Luke Campagnola, Pasha A Davoudian, Alex Hoggarth, Travis A Hage, Alice Bosma-Moody, Christopher A Baker, Jung Hoon Lee, Stefan Mihalas, Corinne Teeter, et al. Sparse recurrent excitatory connectivity in the microcircuit of the adult mouse and human cortex. *elife*, 7:e37349, 2018.

Charles Scott Sherrington. The integrative action of the nervous system. In *Scientific and Medical Knowledge Production, 1796-1918*, pp. 217–253. Routledge, 2023.

Jianghong Shi, Eric Shea-Brown, and Michael Buice. Comparison against task driven artificial neural networks reveals functional properties in mouse visual cortex. *Advances in Neural Information Processing Systems*, 32, 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

Hidenori Tanaka, Aran Nayebi, Niru Maheswaranathan, Lane McIntosh, Stephen Baccus, and Surya Ganguli. From deep learning to mechanistic understanding in neuroscience: the structure of retinal prediction. *Advances in neural information processing systems*, 32, 2019.

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

J Hans Van Hateren and Arjen van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 265(1394):359–366, 1998.

Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12894–12904, 2021.

Po-Wei Wang, Priya Donti, Bryan Wilder, and Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning*, pp. 6545–6554. PMLR, 2019.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/`, 2019.

Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16133–16142, 2023.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

Aiwen Xu, Yuchen Hou, Cristopher Niell, and Michael Beyeler. Multimodal deep learning model unveils behavioral dynamics of v1 activity in freely moving mice. *Advances in Neural Information Processing Systems*, 36, 2024.

Qi Yan, Yajing Zheng, Shanshan Jia, Yichen Zhang, Zhaofei Yu, Feng Chen, Yonghong Tian, Tiejun Huang, and Jian K Liu. Revealing fine structures of the retinal receptive field by deep-learning networks. *IEEE transactions on cybernetics*, 52(1):39–50, 2020.

Guangyu Robert Yang and Xiao-Jing Wang. Artificial neural networks for neuroscientists: a primer. *Neuron*, 107(6):1048–1070, 2020.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Anthony Zador, Sean Escola, Blake Richards, Bence Ölveczky, Yoshua Bengio, Kwabena Boahen, Matthew Botvinick, Dmitri Chklovskii, Anne Churchland, Claudia Clopath, et al. Catalyzing next-generation artificial intelligence through neuroai. *Nature communications*, 14(1):1597, 2023.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Yajing Zheng, Shanshan Jia, Zhaofei Yu, Jian K Liu, and Tiejun Huang. Unraveling neural coding of dynamic natural visual scenes via convolutional recurrent neural networks. *Patterns*, 2(10), 2021.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 13001–13008, 2020.

Haoran Zhu, Boyuan Chen, and Carter Yang. Understanding why vit trains badly on small datasets: an intuitive perspective. *arXiv preprint arXiv:2302.03751*, 2023.

A Appendix / supplemental material

A.1 Biological RFs

For the biological RF analysis, we have analyzed *in vivo* intracortical connectivity dataset of Rossi et al. (2020) collected from the mouse primary visual cortex (V1) (Fig. 1a). It contains both excitatory (CaMK2a-positive) and inhibitory (Gad2-positive) layer 2/3 neuronal spatial connectivity distribution (Fig. 1b) which was determined by recording GCaMP6 signals (calcium activities) of pre-and post-synaptic pairs (Fig. 8 and 7).

Given a post-synaptic neuron positioned at (x_0, y_0), (Fig. 1b black), and the N_b number of functional synapse positions $x_n, y_n \in (-\infty, \infty)$ for $n = 1, 2, ..., N_b$, then relative offsets are defined as $(\Delta x_n, \Delta y_n) = (x_n - x_0, y_n - y_0)$. We summarize functional synapse positions for biological RF as following

$$\Delta \mathbf{s}_{\text{b}} = [(\Delta x_n, \Delta y_n)]_{n=1}^{N_b}. \tag{4}$$

A.2 Artificial RFs

For the artificial RF analysis, we used untrained or pre-trained AlexNet[6] with inputs (224 $\times$ 224) of 17 images either generated from Gaussian noises or selected among 118 Natural Scenes images datasets (See details in Appendix A.10 and Fig. 8). When image inputs were shown to AlexNet, we extracted RFs of the functional synapse from the first convolutional layer (Conv1) (Fig. 1, c and d).

Given the input $\mathcal{X} \in \mathbb{R}^{C_i \times H \times W}$ and weights parameters for Artificial RFs $\mathcal{W}_{\text{ARF}} \in \mathbb{R}^{C_i \times K_h \times K_w}$, the post-synaptic unit in the Convolution layer, receives weighted-input $\mathcal{Z} \in \mathbb{R}^{V \times C_i \times K_h \times K_w}$, the results of element-wise multiplication between partial input and filters, where $\mathcal{Z}_{v=m*(H-K_h+1)+n} = \mathcal{X}_{\cdot,m:m+K_h-1,n:n+K_w-1} \odot \mathcal{W}$ (Fig. 1d black) for $0 \leq m \leq H - K_h$ and $\leq n \leq W - K_w$. We calculate the weighted input as the convolution operation without summation across width, height and input channel. As a result, weighted input $\mathcal{Z}$ has $V \times C_i \times K_h \times K_w$ shape, where $V = (H - K_h + 1) \times (W - K_w + 1)$, and $\mathcal{Z}_v \in \mathbb{R}^{C_i \times K_h \times K_w}$ denotes the v-th element of $\mathcal{Z}$. Here, C_i, H, W, K_h and K_w denote the number of input channels, input height, input width, kernel height and kernel width, respectively. For simplicity, we assume that there is stride one and no zero-padding. For $h \in [0, \cdots, K_h - 1]$ and $w \in [0, \cdots, K_w - 1]$, the relative offsets from the kernel center are defined as follows

$$\Delta s = (\Delta h, \Delta w) = (h - \frac{K_h}{2}, w - \frac{K_w}{2}). \tag{5}$$

Since spatial connectivity pattern in the biological synapse is measured by the functional calcium activities and given as coordinates, we applied a similar approach to that of CNN layers. We collected N_a functional weighted-inputs (functional synapses) where N_a represents the number of cases where each elements of $|\mathcal{Z}|$ exceeds a threshold θ. Here, we defined θ as the standard deviation of $|\mathcal{Z}|$ [7]. This selection process yielded a different set of functional synapses input-dependent manner. We summarize functional synapse positions for artificial RF as following

$$\Delta \mathbf{s}_{\text{a}} = \{(v, k, \Delta h, \Delta w) | \mathcal{Z}_{v,k,h,w} > \theta\}, \text{ where } |\Delta \mathbf{s}_{\text{a}}| = N_a. \tag{6}$$

[6]For the pre-trained model, we used the torchvision's ImageNet-1k pre-trained model

[7]We determined the activity threshold based on a common method used in neuroscience to extract meaningful patterns in neural activity, which is similar to calculating the Z-score and typically setting a threshold at a range of 2 to 3 standard deviations to identify values that are statistically significant.

A.3 GAUSSIAN DISTRIBUTED FUNCTIONAL SYNAPSES OF POST-SYNAPTIC NEURON IN MOUSE V1 LAYER 2/3

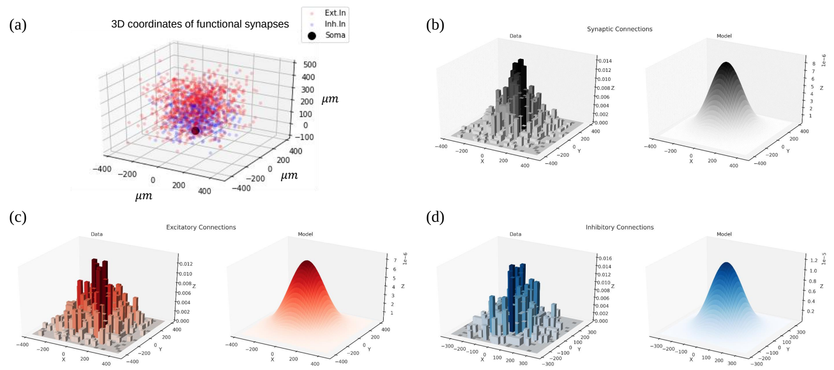

Figure 7: **Distribution of post-synaptic functional synapses in mouse V1 layer 2/3** (a) Using dataset from Rossi et al. (2020), 3D scatter plot represent relative positions of both excitatory (red) and inhibitory (blue) functional synapses from the soma of the post-synaptic neuron. (b-d) 2d histogram (left) and Gaussian fitted probability density function (right), showing the laminar organization of functional synapses for all (b), excitatory (c), and inhibitory (d)

A.4 INDIVIDUAL RECEPTIVE FIELDS COLLECTED FROM BOTH BIOLOGICAL AND ARTIFICIAL SYSTEMS

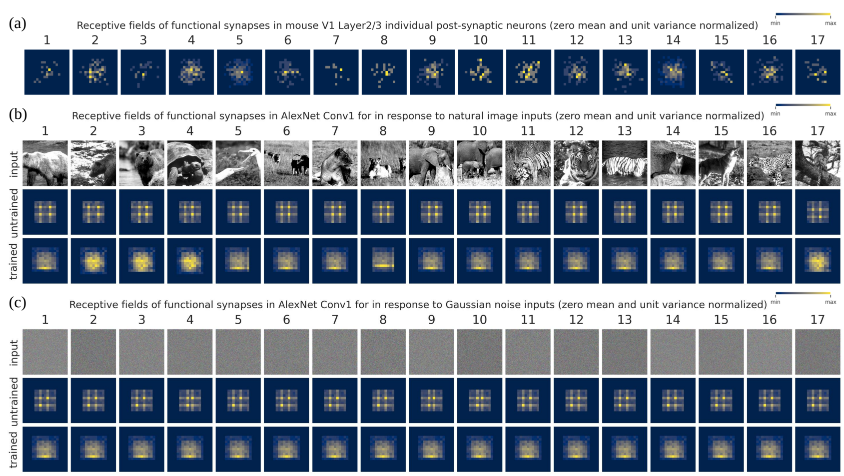

Figure 8: **Biological and artificial receptive fields with visual stimulus** The receptive field discussed in this figure specifically refers to the spatial connectivity patterns of synapses. Note that this differs from receptive fields typically associated with low-level visual feature selectivity. (a) Receptive fields of individual neurons in V1 Layer2/3 from the dataset Rossi et al. (2020) (b) Receptive fields of untrained or ImageNet-1k pretrained AlexNet's Conv1 layer when Natural Scenes images were shown (c) Receptive fields of untrained or ImageNet-1k pretrained AlexNet's Conv1 layer when Gaussian RGB noise were shown. All receptive fields are zero mean unit variance normalized.

A.5 Experimental details for Sudoku Challenge

We utilized the extensive 1M-sudoku dataset (Park, 2018), a resource also utilized in prior works (Amos & Kolter, 2017; Palm et al., 2018; Wang et al., 2019). Sudoku, a widely popular number puzzle, involves organizing digits in a grid such that each row (1×9), column (9×1), and box (3×3) contains all numbers from 1 to 9. In the Sudoku challenge, where achieving these three objectives simultaneously is essential for complete Sudoku solving, there is an advantage that we can test the applicability and effectiveness of evolving RFs in the L_p-convolution. To achieve this, we compared five distinct models of Sudoku CNN: (3×3) Base model, (7×7) Large model, (7×7) $L_p(p_{\text{init}} = 2)$, (7×7) $L_p(p_{\text{init}} = 16)$, and finally (7×7) $L_p^{\dagger}(p_{\text{init}} = 256)$ (frozen p and $\mathbf{C}$ model) 2. For numerical stability, we clipped $p \geq 2$ during the training of the Sudoku-solving task. The inputs, targets, and model architecture are outlined in Figure 5a. As illustrated, our model comprises repeated Conv2dSame blocks, originally introduced in SudokuCNN (Oinar, 2021). Each Conv2dSame block encompasses Reflection padding, followed by a conventional Convolutional or L_p-Convolutional layer, Batch Normalization, and an activation function. The Convolutional layer has 256 channels, and the number of blocks is set at $L = 10$.

A.6 Conformational analysis of L_p-masks

We defined the properties of scale (α), rotation (θ), and distortion (γ) through Singular Value Decomposition (SVD) on $\mathbf{C}$ for each output channel, as shown in the following equation:

$$\mathbf{C} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T. \tag{7}$$

Here, $\mathbf{U}$ and $\mathbf{V}$ represent 2×2 unitary matrices containing the left and right singular vectors. $\mathbf{\Lambda}$ is a diagonal matrix containing the singular values (λ_0, λ_1). Rotation is quantified as $\theta = \arctan\left(\frac{\sin(\mathbf{V}^T[1])}{\cos(\mathbf{V}^T[1])}\right)$ (in degrees), providing a measure of rotational transformation. Distortion is quantified as $\gamma = \frac{\lambda_0}{\lambda_1}$, offering valuable information about the deformation present in the data. Scale is quantified as $\alpha = \sqrt{\left(\frac{1}{\lambda_0}\right)^2 + \left(\frac{1}{\lambda_1}\right)^2}$, indicating the size of mask. In Figure 5a, we show the example shapes of L_p-masks by reverse calculating $\mathbf{C}$ from the given α, θ, γ.

A.7 Layer-wise visualization of L_p-masks for Sudoku-$L_p$convNet

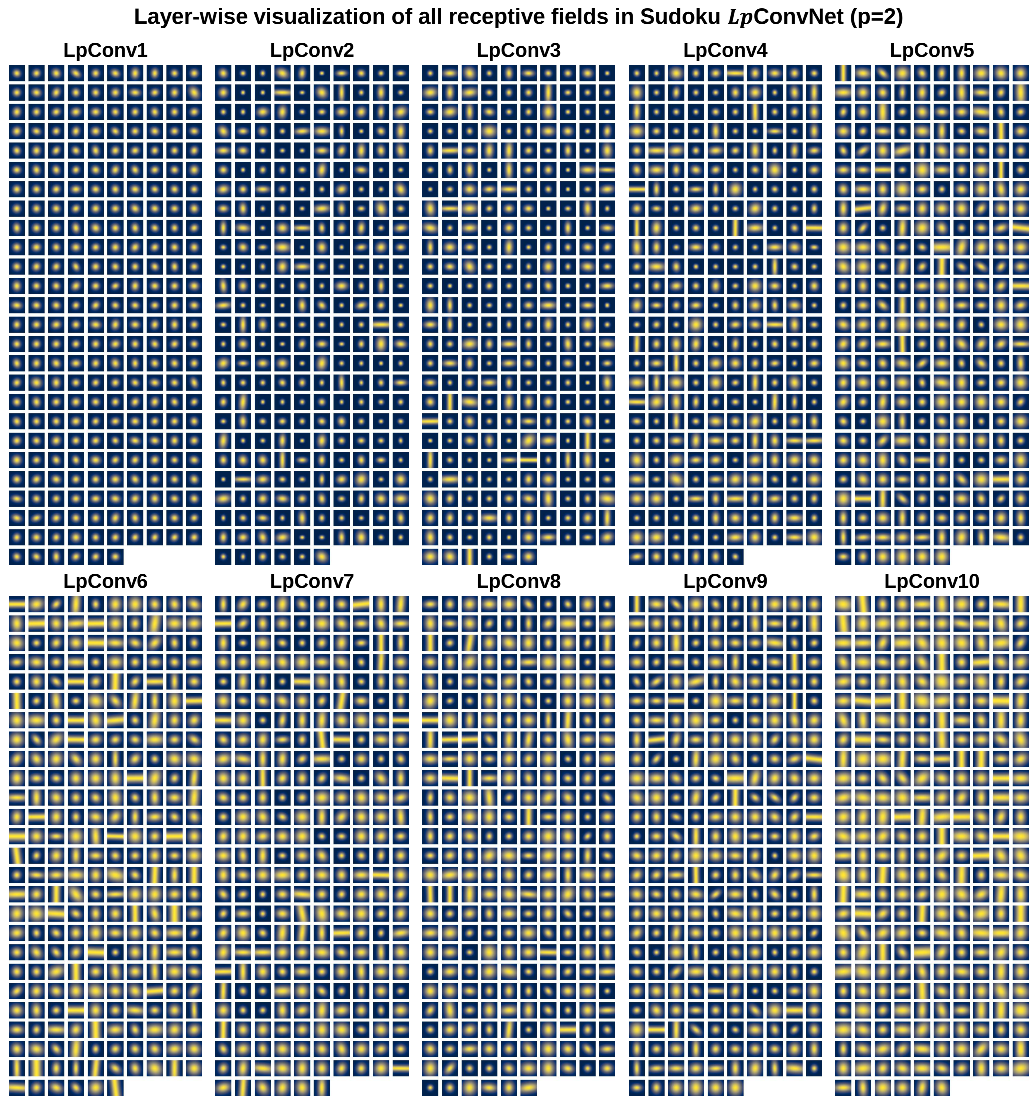

Figure 9: **Layer-wise visualization of L_p-masks for Sudoku-$L_p$convNet** All learned L_p-masks after Sudoku task training of L_p-ConvNet($p_{\text{init}} = 2$). With an increase in layer depth, the sizes of masks get larger.

A.8 ACCURACY CURVES FOR SUDOKU ABLATION EXPERIMENTS ON L_p(P=2)

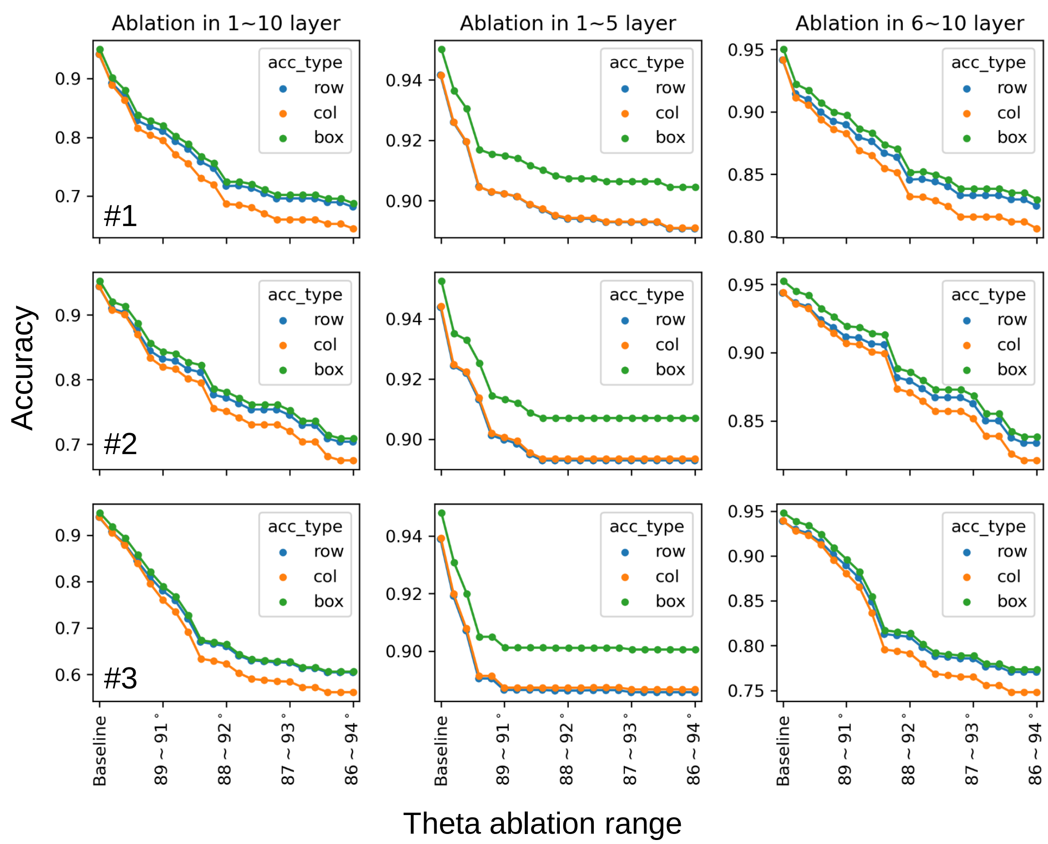

Figure 10: **Accuracy curves for Sudoku ablation experiments on L_p(p=2)**. 3 individual experiments (rows) of vertical L_p-masks ablations with 3 different conditions (columns; layer 1 to 10, left; layer 1 to 5, middle; layer 6 to 10, right) on L_p(p=2)

A.9 Experimental details of vision classification task

We conducted our training on two datasets: CIFAR-100 (Krizhevsky et al., 2009) and TinyImageNet (Le & Mikolov, 2014). CIFAR-100 comprises 32×32 pixel images distributed across 100 classes, while TinyImageNet consists of 224×224 pixel images spanning 200 categories. Following standard procedures, we reported top-1 accuracy with corresponding mean and standard deviation. Our implementation is based on the **PyTorch** framework (Paszke et al., 2019), making extensive use of the **timm** repository (Wightman, 2019). We adopted a training strategy rooted in DeiT (Touvron et al., 2021), incorporating techniques such as RandAugment (Cubuk et al.), Mixup (Zhang et al., 2017), Cutmix (Yun et al., 2019), random erasing (Zhong et al., 2020), and stochastic depth (Huang et al., 2016). The optimization process employed AdamW (Loshchilov & Hutter, 2017) with a default momentum value of 0.9 and a weight decay set at 5×10^{-2}. We initialized our learning rate at 1×10^{-3} and implemented a cosine learning rate schedule. All models underwent training for 300 epochs, utilizing a batch size of 128. For CIFAR-100, training was conducted on 2 GTX 1080ti GPUs, while 2 Tesla V100 GPUs were used for TinyImageNet.

A.10 The Allen Brain Observatory dataset

The Allen Brain Observatory dataset (de Vries et al., 2020) constitutes a comprehensive standardized in vivo examination of physiological activity within the mouse visual cortex. It encompasses recordings of visually-induced calcium responses from neurons expressing GCaMP6f. This dataset encompasses cortical activity from nearly 60,000 neurons originating from six distinct visual areas, four layers, and twelve transgenic mouse Cre lines. These recordings were gathered from 243 adult mice in reaction to a diverse set of visual stimuli. In this study, we focused on utilizing the collective neural responses from five visual areas (VISal, VISam, VISl, VISp, VISpm), Layer 2/3 (depth range of 175mm to 275mm), and three mouse lines (Cux2-CreERT, Emx1-IRES-Cre, Slc17a7-IRES2-Cre) when presenting a dataset of natural scenes to the mice. This dataset comprised 118 natural images obtained from three different databases (Berkeley Segmentation Dataset (Martin et al., 2001), van Hateren Natural Image Dataset (Van Hateren & van der Schaaf, 1998), and McGill Calibrated Colour Image Database (Olmos & Kingdom, 2004)). Further details regarding the experiment can be found in (de Vries et al., 2020). In our study, we employed both images and neural responses for experiments involving representational similarity analysis to evaluate the correspondence between CNNs and the visual cortex, mirroring earlier investigations (Shi et al., 2019; Bakhtiari et al., 2021).

A.11 Representation similarity analysis

While the details of RSA are expertly addressed in Diedrichsen & Kriegeskorte (2017), let us briefly cover our specific approach. We leveraged the codebase provided by Bakhtiari et al. (2021). In RSA, we generate response matrices ($\mathbf{R}$) for brain regions and neural network layers, with dimensions $N \times M$ (where N is the number of image inputs and M is the neuron count). Using Pearson correlation, we compute similarities within matrix $\mathbf{R}$ to construct the $N \times N$ Representation Similarity Matrix (RSM). Additionally, following the methodology of Bakhtiari et al. (2021), we applied noise correction by normalizing the RSAs using the noise ceiling values. These values were obtained through comparisons of representations across different mice. For example, the noise ceiling value for VISp is derived by calculating the RSMs of VISp from different animals and taking their median. To assess the similarity between RSMs (SSM), we employ Kendall's τ for robust agreement, which helps mitigate potential bias from measurement noise Diedrichsen et al. (2020).

A.12 Pair-wise representation similarity analysis between all CNN layers and V1 subregions

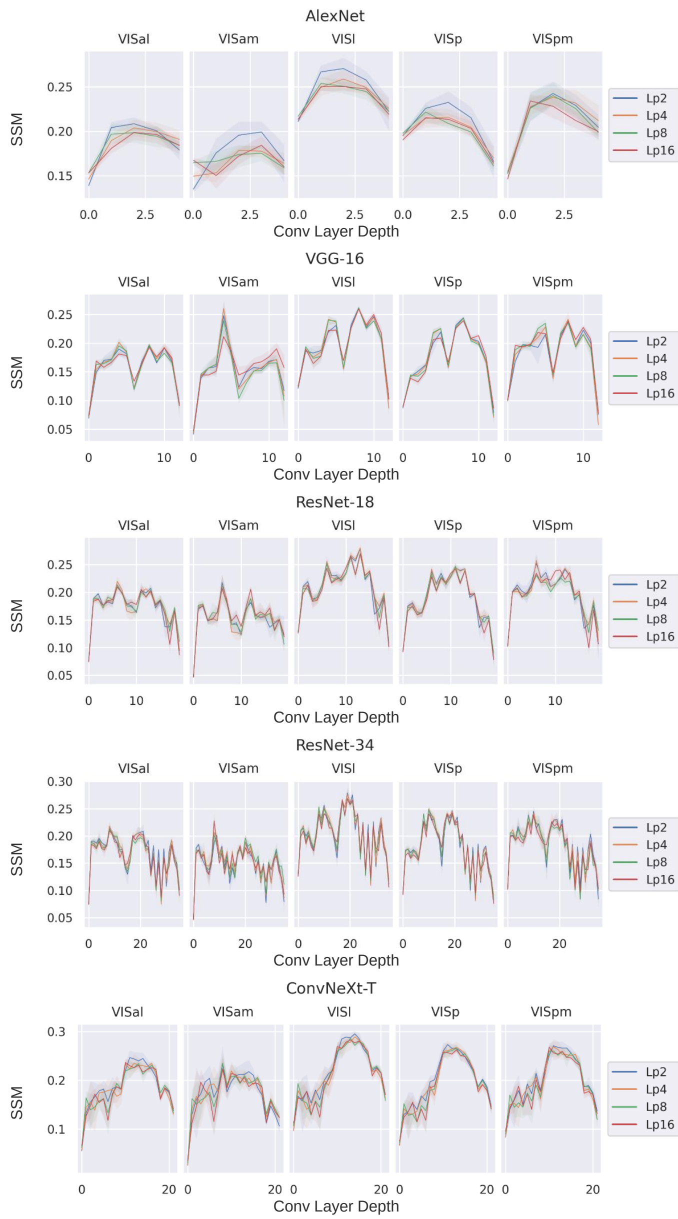

Figure 11: **Pair-wise representation similarity analysis between all CNN layers and VC subregions**. We show the SSM scores for all pairs of Conv layers from CNNs and VC subregions. y-axis, SSM score; x-axis, Conv layer depth. For Max. SSM, we choose the highest SSM among all pair-wise SSMs.

A.13 VISUALIZATION OF FUNCTIONAL RECEPTIVE FIELDS OF PRE-TRAINED ALEXNET

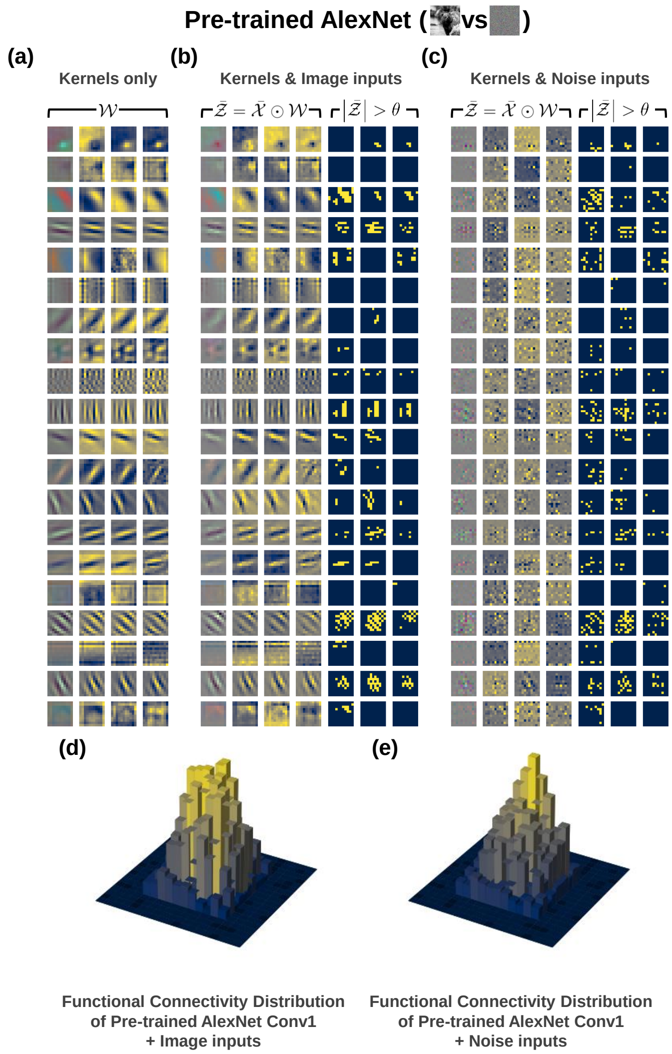

Figure 12: **Visualization of functional receptive fields of pre-trained AlexNet Conv1 with image or noise inputs**. Visualization of first 20 kernels of total 64 without inputs (a; column orders: RGB, R, G, B), with image inputs (b; column orders: RGB, R, G, B, R, G, B) with noise inputs (c; column orders: RGB, R, G, B, R, G, B). (d) Histogram of functional connectivity from (b). (e) Histogram of functional connectivity from (c). $\mathcal{W}$, Weight; $\bar{\mathcal{X}}$, kernel-sized input; $\bar{\mathcal{Z}}$, kernel-sized output; θ, activity threshold; $\odot$, element-wise product. See Appendix A.2. for methodological details.

A.14 VISUALIZATION OF FUNCTIONAL RECEPTIVE FIELDS OF UNTRAINED ALEXNET

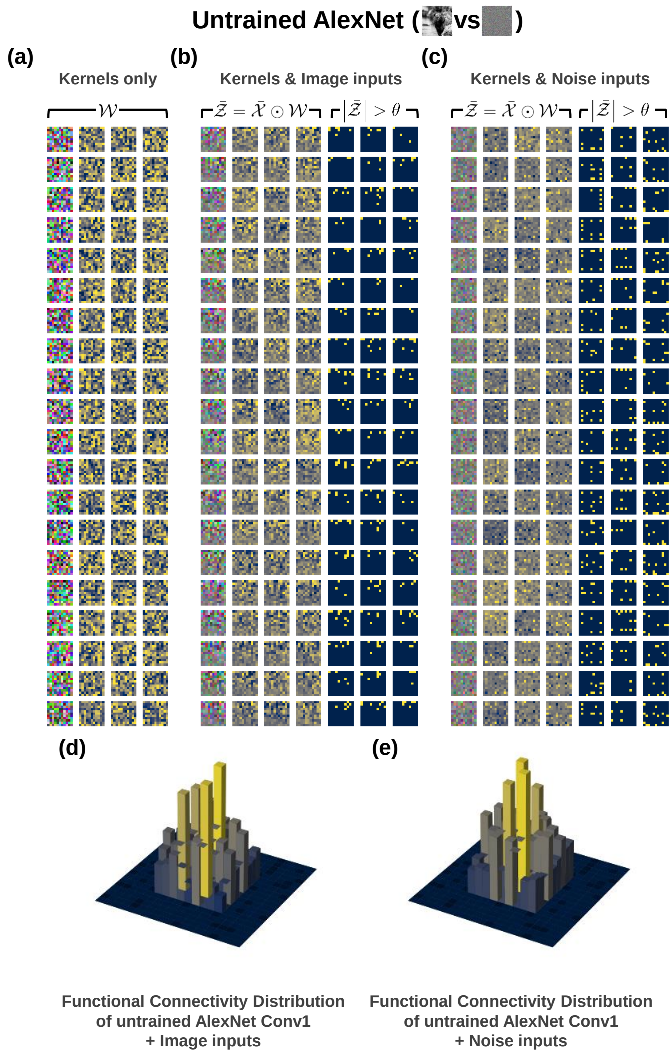

Figure 13: **Visualization of functional receptive fields of untrained AlexNet Conv1 with image or noise inputs**. Visualization of first 20 kernels of total 64 without inputs (a; column orders: RGB, R, G, B), with image inputs (b; column orders: RGB, R, G, B, R, G, B) with noise inputs (c; column orders: RGB, R, G, B, R, G, B). (d) Histogram of functional connectivity from (b). (e) Histogram of functional connectivity from (c). $\mathcal{W}$, Weight; $\bar{\mathcal{X}}$, kernel-sized input; $\bar{\mathcal{Z}}$, kernel-sized output; θ, activity threshold; $\odot$, element-wise product. See Appendix A.2. for methodological details.

A.15 VISUALIZATION OF MPND WHEN $p < 2$

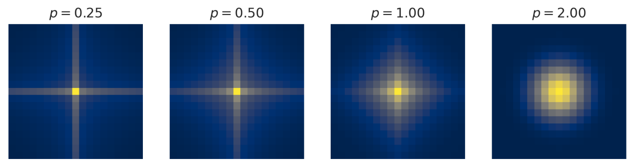

Figure 14: **Visualization of MPND when** $p < 2$ Given the value of $p = 1$, MPND distribution becomes diamond shape. When $p < 1$, the distribution becomes a cross-like shape.

A.16 POST-TRAINED p-DISTRIBUTION IN L_p-MASKS

A.16.1 OVERALL DISTRIBUTION IN CIFAR100

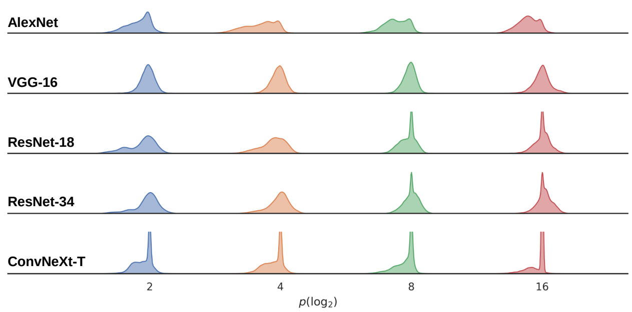

Figure 15: **CIFAR-100-trained p-distribution of L_p-masks**

A.16.2 OVERALL DISTRIBUTION IN TINYIMAGENET

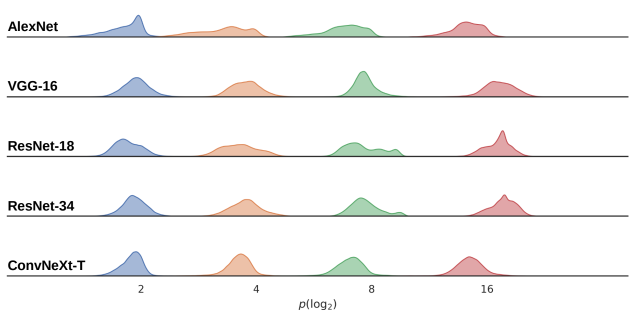

Figure 16: **TinyImageNet-trained p-distribution of L_p-masks**

A.16.3 LAYER-WISE DISTRIBUTION IN CIFAR100

Table 6: **Layer-wise p-distribution of CIFAR-100-trained L_p-masks** All values (Median $\pm$ Stdev) are calculated with p of all L_p-masks in each layer, from 5 different trials of CIFAR-100-trained models.

CIFAR-100

Model	**Layer**	**L_p-Conv (p=2)**	**L_p-Conv (p=4)**	**L_p-Conv (p=8)**	**L_p-Conv (p=16)**
AlexNet	1	1.99 ± 0.08	3.88 ± 0.16	7.75 ± 0.35	15.45 ± 0.68
	2	2.00 ± 0.06	3.93 ± 0.19	7.67 ± 0.52	15.35 ± 0.96
	3	1.91 ± 0.07	3.70 ± 0.16	7.64 ± 0.37	15.06 ± 0.79
	4	1.80 ± 0.13	3.38 ± 0.32	7.18 ± 0.36	14.73 ± 0.64
	5	1.82 ± 0.11	3.35 ± 0.29	7.11 ± 0.38	14.45 ± 0.68
ResNet-18	1	1.98 ± 0.11	3.85 ± 0.23	7.70 ± 0.49	15.39 ± 1.27
	2	2.01 ± 0.06	3.87 ± 0.13	7.91 ± 0.32	15.37 ± 0.84
	3	1.93 ± 0.05	3.68 ± 0.11	7.45 ± 0.25	15.30 ± 0.58
	4	1.95 ± 0.05	3.72 ± 0.12	7.61 ± 0.26	15.43 ± 0.61
	5	1.92 ± 0.05	3.65 ± 0.12	7.45 ± 0.23	15.55 ± 0.52
	6	1.98 ± 0.05	3.80 ± 0.10	7.58 ± 0.20	16.08 ± 0.56
	7	1.82 ± 0.05	3.45 ± 0.13	7.30 ± 0.26	14.92 ± 0.57
	8	1.71 ± 0.06	3.57 ± 0.17	8.00 ± 0.01	16.00 ± 0.00
	9	1.88 ± 0.05	3.61 ± 0.12	7.49 ± 0.21	15.47 ± 0.58
	10	1.93 ± 0.05	3.70 ± 0.11	7.50 ± 0.22	15.82 ± 0.48
	11	2.01 ± 0.06	3.90 ± 0.13	7.68 ± 0.21	16.28 ± 0.56
	12	1.79 ± 0.06	3.52 ± 0.13	7.35 ± 0.20	15.31 ± 0.51
	13	1.62 ± 0.05	3.42 ± 0.14	8.00 ± 0.01	16.00 ± 0.00
	14	1.93 ± 0.06	3.82 ± 0.11	7.72 ± 0.20	16.22 ± 0.56
	15	2.01 ± 0.05	3.96 ± 0.10	7.93 ± 0.20	16.54 ± 0.54
	16	2.09 ± 0.06	4.16 ± 0.11	8.07 ± 0.21	16.80 ± 0.60
	17	1.94 ± 0.05	3.84 ± 0.10	7.69 ± 0.22	15.54 ± 0.41
	18	1.74 ± 0.04	3.86 ± 0.12	8.00 ± 0.00	16.00 ± 0.00
	19	1.99 ± 0.06	4.02 ± 0.11	8.10 ± 0.21	16.19 ± 0.35
	20	2.02 ± 0.05	4.09 ± 0.09	8.21 ± 0.18	16.30 ± 0.25
ResNet-34	1	1.99 ± 0.12	3.87 ± 0.26	7.73 ± 0.44	15.51 ± 1.01
	2	2.01 ± 0.06	3.87 ± 0.12	7.99 ± 0.33	15.25 ± 0.85
	3	1.96 ± 0.05	3.75 ± 0.10	7.65 ± 0.28	15.35 ± 0.51
	4	1.96 ± 0.05	3.78 ± 0.11	7.83 ± 0.25	15.54 ± 0.58
	5	1.93 ± 0.05	3.71 ± 0.12	7.56 ± 0.26	15.56 ± 0.49
	6	1.96 ± 0.05	3.76 ± 0.11	7.65 ± 0.23	15.72 ± 0.57
	7	1.94 ± 0.05	3.74 ± 0.11	7.58 ± 0.24	15.80 ± 0.45
	8	2.01 ± 0.05	3.90 ± 0.11	7.72 ± 0.21	16.27 ± 0.52
	9	1.83 ± 0.05	3.52 ± 0.11	7.45 ± 0.28	14.83 ± 0.58
	10	1.69 ± 0.06	3.49 ± 0.17	8.00 ± 0.01	16.00 ± 0.00
	11	1.89 ± 0.05	3.69 ± 0.11	7.61 ± 0.22	15.60 ± 0.52
	12	1.93 ± 0.05	3.76 ± 0.12	7.69 ± 0.25	15.81 ± 0.49
	13	1.95 ± 0.05	3.79 ± 0.10	7.72 ± 0.21	15.95 ± 0.52
	14	1.99 ± 0.05	3.89 ± 0.12	7.85 ± 0.24	16.16 ± 0.52
	15	1.97 ± 0.05	3.85 ± 0.11	7.77 ± 0.21	16.06 ± 0.52
	16	2.03 ± 0.05	3.98 ± 0.11	7.98 ± 0.23	16.39 ± 0.52
	17	2.06 ± 0.06	4.07 ± 0.12	7.91 ± 0.23	16.67 ± 0.51
	18	1.80 ± 0.07	3.59 ± 0.12	7.41 ± 0.21	15.41 ± 0.56
	19	1.65 ± 0.05	3.48 ± 0.14	8.00 ± 0.01	16.00 ± 0.00
	20	1.93 ± 0.06	3.87 ± 0.11	7.78 ± 0.21	16.29 ± 0.51
	21	2.01 ± 0.06	4.03 ± 0.10	8.04 ± 0.21	16.45 ± 0.53
	22	2.00 ± 0.06	4.00 ± 0.10	8.01 ± 0.20	16.49 ± 0.56
	23	2.05 ± 0.06	4.10 ± 0.10	8.19 ± 0.21	16.59 ± 0.52
	24	2.04 ± 0.05	4.06 ± 0.10	8.12 ± 0.20	16.57 ± 0.55
	25	2.03 ± 0.06	4.11 ± 0.12	8.23 ± 0.24	16.83 ± 0.51
	26	2.05 ± 0.05	4.08 ± 0.10	8.15 ± 0.20	16.59 ± 0.53
	27	2.02 ± 0.07	4.09 ± 0.13	8.24 ± 0.27	16.89 ± 0.52
	28	2.05 ± 0.06	4.08 ± 0.11	8.14 ± 0.20	16.51 ± 0.52
	29	2.02 ± 0.06	4.11 ± 0.13	8.32 ± 0.28	16.86 ± 0.50
	30	2.16 ± 0.06	4.31 ± 0.10	8.39 ± 0.20	17.12 ± 0.61
	31	1.95 ± 0.05	3.86 ± 0.09	7.75 ± 0.19	15.63 ± 0.37
	32	1.79 ± 0.04	3.95 ± 0.11	8.00 ± 0.00	16.00 ± 0.00
	33	1.98 ± 0.05	3.99 ± 0.10	8.05 ± 0.19	16.12 ± 0.29
	34	2.02 ± 0.04	4.06 ± 0.08	8.16 ± 0.18	16.26 ± 0.26
	35	1.97 ± 0.06	3.98 ± 0.11	8.01 ± 0.21	16.01 ± 0.27
	36	2.07 ± 0.05	4.14 ± 0.10	8.25 ± 0.30	16.22 ± 0.45
VGG-16	1	2.07 ± 0.09	4.08 ± 0.17	8.22 ± 0.37	16.62 ± 0.85
	2	2.00 ± 0.06	3.90 ± 0.12	7.77 ± 0.25	15.88 ± 0.63
	3	1.98 ± 0.05	3.88 ± 0.13	7.95 ± 0.31	15.51 ± 0.80
	4	1.98 ± 0.06	3.83 ± 0.13	7.69 ± 0.26	15.60 ± 0.60
	5	1.96 ± 0.06	3.86 ± 0.13	7.94 ± 0.26	15.43 ± 0.68
	6	1.94 ± 0.07	3.82 ± 0.13	7.80 ± 0.25	15.87 ± 0.55
	7	1.94 ± 0.06	3.81 ± 0.12	7.69 ± 0.22	15.94 ± 0.58
	8	1.98 ± 0.06	3.93 ± 0.11	7.94 ± 0.26	15.94 ± 0.69
	9	2.02 ± 0.06	4.04 ± 0.10	8.07 ± 0.24	16.17 ± 0.67
	10	1.98 ± 0.07	3.98 ± 0.12	7.91 ± 0.18	16.59 ± 0.90
	11	1.91 ± 0.08	3.88 ± 0.12	7.80 ± 0.23	15.78 ± 0.44
	12	2.01 ± 0.06	4.06 ± 0.11	8.09 ± 0.20	16.19 ± 0.35
	13	2.00 ± 0.05	4.01 ± 0.09	8.00 ± 0.20	15.95 ± 0.43
ConvNeXt-T	1	2.06 ± 0.11	3.99 ± 0.15	7.97 ± 0.39	15.71 ± 0.95
	2	1.88 ± 0.06	3.66 ± 0.15	7.09 ± 0.34	14.47 ± 0.55
	3	1.86 ± 0.06	3.59 ± 0.16	7.04 ± 0.31	14.33 ± 0.54
	4	1.86 ± 0.07	3.61 ± 0.16	7.00 ± 0.35	14.43 ± 0.56
	5	1.96 ± 0.07	3.99 ± 0.15	7.88 ± 0.34	15.71 ± 0.54
	6	1.84 ± 0.08	3.77 ± 0.14	7.61 ± 0.29	15.41 ± 0.55
	7	1.83 ± 0.10	3.72 ± 0.16	7.53 ± 0.32	15.27 ± 0.60
	8	1.80 ± 0.09	3.72 ± 0.19	7.65 ± 0.29	15.38 ± 0.56
	9	1.83 ± 0.07	3.61 ± 0.24	6.93 ± 0.48	14.20 ± 0.71
	10	1.93 ± 0.07	3.89 ± 0.15	7.85 ± 0.23	16.00 ± 0.00
	11	1.94 ± 0.07	3.87 ± 0.16	7.83 ± 0.23	16.00 ± 0.00
	12	1.95 ± 0.08	3.78 ± 0.19	7.70 ± 0.35	16.00 ± 0.00
	13	1.95 ± 0.07	3.90 ± 0.17	7.89 ± 0.26	16.00 ± 0.00
	14	1.93 ± 0.07	3.85 ± 0.17	7.82 ± 0.26	16.00 ± 0.00
	15	1.96 ± 0.07	3.93 ± 0.14	7.88 ± 0.20	16.00 ± 0.00
	16	1.97 ± 0.07	3.91 ± 0.15	7.88 ± 0.25	16.00 ± 0.00
	17	1.97 ± 0.07	3.93 ± 0.14	7.90 ± 0.20	16.00 ± 0.00
	18	2.01 ± 0.06	3.97 ± 0.14	7.98 ± 0.18	16.00 ± 0.00
	19	1.83 ± 0.06	3.64 ± 0.13	7.32 ± 0.26	15.08 ± 0.46
	20	2.00 ± 0.00	4.00 ± 0.00	8.00 ± 0.00	16.00 ± 0.00
	21	2.00 ± 0.00	4.00 ± 0.00	8.00 ± 0.00	16.00 ± 0.00
	22	2.00 ± 0.00	4.00 ± 0.00	8.00 ± 0.00	16.00 ± 0.00

A.16.4 LAYER-WISE DISTRIBUTION IN TINYIMAGENET

Table 7: **Layer-wise p-distribution of TinyImageNet-trained L_p-masks**All values (Median ± Stdev) are calculated with p of all L_p-masks in each layer, from 5 different trials of TinyImageNet-trained models.c

TinyImageNet

Model	Layer	L_p-Conv (p=2)	L_p-Conv (p=4)	L_p-Conv (p=8)	L_p-Conv (p=16)
AlexNet	1	1.91 ± 0.22	3.62 ± 0.31	7.21 ± 0.41	14.57 ± 0.89
	2	1.99 ± 0.06	3.91 ± 0.28	7.52 ± 0.70	15.21 ± 1.33
	3	1.91 ± 0.11	3.46 ± 0.25	7.15 ± 0.66	14.78 ± 1.34
	4	1.71 ± 0.16	2.97 ± 0.37	6.51 ± 0.58	13.86 ± 0.91
	5	1.70 ± 0.18	2.82 ± 0.47	6.34 ± 0.75	13.41 ± 1.14
ResNet-18	1	2.12 ± 0.33	4.04 ± 0.41	7.92 ± 0.70	15.93 ± 1.82
	2	2.07 ± 0.08	4.02 ± 0.17	7.83 ± 0.49	16.56 ± 1.32
	3	1.98 ± 0.08	3.75 ± 0.17	7.36 ± 0.39	16.97 ± 1.42
	4	1.99 ± 0.09	3.75 ± 0.17	7.35 ± 0.36	16.92 ± 1.29
	5	1.98 ± 0.08	3.68 ± 0.18	7.19 ± 0.30	17.19 ± 1.26
	6	2.02 ± 0.10	3.85 ± 0.18	7.45 ± 0.28	17.52 ± 1.29
	7	1.84 ± 0.11	3.35 ± 0.22	6.77 ± 0.42	15.33 ± 1.00
	8	1.90 ± 0.08	4.05 ± 0.23	8.83 ± 0.31	17.29 ± 0.32
	9	1.89 ± 0.12	3.48 ± 0.25	7.06 ± 0.43	15.96 ± 1.07
	10	1.98 ± 0.10	3.63 ± 0.24	7.17 ± 0.41	16.97 ± 1.20
	11	2.02 ± 0.11	3.82 ± 0.23	7.40 ± 0.37	17.15 ± 1.18
	12	1.82 ± 0.11	3.30 ± 0.23	6.71 ± 0.35	15.50 ± 1.02
	13	1.75 ± 0.06	3.77 ± 0.18	8.64 ± 0.25	17.16 ± 0.27
	14	1.84 ± 0.12	3.39 ± 0.23	7.07 ± 0.37	16.47 ± 1.08
	15	1.93 ± 0.12	3.57 ± 0.28	7.24 ± 0.42	17.77 ± 1.15
	16	2.01 ± 0.11	3.82 ± 0.24	7.56 ± 0.41	17.49 ± 1.10
	17	1.75 ± 0.08	3.19 ± 0.16	6.68 ± 0.26	15.60 ± 0.70
	18	1.82 ± 0.06	4.28 ± 0.17	9.25 ± 0.18	17.59 ± 0.16
	19	1.78 ± 0.10	3.25 ± 0.16	7.18 ± 0.38	17.55 ± 0.96
	20	1.69 ± 0.06	3.57 ± 0.13	8.29 ± 0.27	18.37 ± 0.67
ResNet-34	1	2.11 ± 0.24	4.02 ± 0.36	7.92 ± 0.59	15.75 ± 1.97
	2	2.10 ± 0.08	4.04 ± 0.17	7.93 ± 0.50	16.90 ± 1.29
	3	2.04 ± 0.09	3.87 ± 0.20	7.48 ± 0.53	17.23 ± 1.30
	4	2.04 ± 0.09	3.88 ± 0.19	7.50 ± 0.49	17.14 ± 1.28
	5	2.02 ± 0.09	3.80 ± 0.22	7.36 ± 0.40	17.41 ± 1.27
	6	2.02 ± 0.08	3.80 ± 0.17	7.39 ± 0.35	17.48 ± 1.32
	7	2.05 ± 0.08	3.83 ± 0.19	7.35 ± 0.43	17.65 ± 1.18
	8	2.10 ± 0.10	3.96 ± 0.17	7.53 ± 0.32	18.15 ± 1.33
	9	1.87 ± 0.12	3.46 ± 0.25	6.93 ± 0.53	15.58 ± 0.98
	10	1.86 ± 0.07	3.93 ± 0.20	8.67 ± 0.30	17.14 ± 0.30
	11	1.87 ± 0.12	3.54 ± 0.24	7.21 ± 0.44	16.33 ± 1.07
	12	1.96 ± 0.13	3.67 ± 0.28	7.27 ± 0.48	17.16 ± 1.17
	13	1.96 ± 0.12	3.67 ± 0.22	7.34 ± 0.42	17.24 ± 1.21
	14	2.06 ± 0.11	3.88 ± 0.25	7.64 ± 0.55	17.96 ± 1.18
	15	1.98 ± 0.11	3.75 ± 0.22	7.48 ± 0.38	17.63 ± 1.25
	16	2.12 ± 0.12	4.05 ± 0.25	7.98 ± 0.61	18.33 ± 1.11
	17	2.11 ± 0.13	4.03 ± 0.24	7.62 ± 0.50	18.18 ± 1.24
	18	1.88 ± 0.12	3.47 ± 0.26	6.90 ± 0.42	15.96 ± 1.15
	19	1.73 ± 0.06	3.73 ± 0.18	8.61 ± 0.25	17.13 ± 0.26
	20	1.89 ± 0.12	3.52 ± 0.24	7.20 ± 0.39	16.64 ± 1.10
	21	1.94 ± 0.15	3.56 ± 0.32	7.27 ± 0.51	17.63 ± 1.20
	22	1.90 ± 0.11	3.61 ± 0.21	7.40 ± 0.38	17.34 ± 1.13
	23	1.99 ± 0.12	3.81 ± 0.27	7.55 ± 0.53	18.43 ± 1.07
	24	1.87 ± 0.11	3.64 ± 0.19	7.57 ± 0.34	17.72 ± 1.07
	25	2.00 ± 0.10	3.92 ± 0.22	7.91 ± 0.47	18.69 ± 0.85
	26	1.85 ± 0.09	3.71 ± 0.16	7.69 ± 0.29	17.94 ± 0.95
	27	2.03 ± 0.09	4.02 ± 0.20	8.18 ± 0.48	18.64 ± 0.82
	28	1.86 ± 0.08	3.77 ± 0.16	7.79 ± 0.27	17.98 ± 0.88
	29	2.07 ± 0.10	4.11 ± 0.21	8.35 ± 0.53	18.54 ± 0.83
	30	1.95 ± 0.09	3.90 ± 0.20	7.69 ± 0.38	18.75 ± 1.07
	31	1.98 ± 0.10	3.71 ± 0.21	7.31 ± 0.29	15.82 ± 0.70
	32	1.86 ± 0.07	4.41 ± 0.22	9.44 ± 0.21	17.74 ± 0.18
	33	1.82 ± 0.09	3.31 ± 0.16	6.90 ± 0.30	16.64 ± 1.02
	34	1.88 ± 0.09	3.55 ± 0.19	7.43 ± 0.29	18.29 ± 1.02
	35	1.87 ± 0.12	3.47 ± 0.19	7.54 ± 0.41	17.82 ± 0.97
	36	1.90 ± 0.06	3.78 ± 0.13	8.07 ± 0.26	18.92 ± 0.69
VGG-16	1	2.07 ± 0.16	4.11 ± 0.24	8.29 ± 0.72	17.52 ± 1.71
	2	2.15 ± 0.10	4.17 ± 0.21	8.18 ± 0.52	17.27 ± 1.11
	3	2.08 ± 0.11	4.09 ± 0.23	7.99 ± 0.54	17.38 ± 1.30
	4	2.15 ± 0.10	4.12 ± 0.21	7.92 ± 0.49	17.76 ± 1.16
	5	1.95 ± 0.10	3.79 ± 0.22	7.65 ± 0.52	17.52 ± 1.39
	6	1.98 ± 0.12	3.78 ± 0.29	7.55 ± 0.56	18.26 ± 1.25
	7	2.04 ± 0.14	3.98 ± 0.31	7.72 ± 0.60	18.05 ± 1.36
	8	1.95 ± 0.13	3.72 ± 0.30	7.50 ± 0.54	16.84 ± 1.32
	9	1.91 ± 0.14	3.68 ± 0.29	7.69 ± 0.48	18.04 ± 1.27
	10	2.00 ± 0.15	3.90 ± 0.29	7.91 ± 0.50	18.38 ± 1.64
	11	1.82 ± 0.11	3.43 ± 0.19	7.28 ± 0.36	16.61 ± 1.28
	12	1.83 ± 0.10	3.51 ± 0.18	7.52 ± 0.27	16.78 ± 1.29
	13	1.96 ± 0.07	3.86 ± 0.14	7.64 ± 0.26	16.09 ± 0.80
ConvNeXt-T	1	1.98 ± 0.20	4.01 ± 0.37	8.16 ± 1.06	16.22 ± 2.12
	2	1.95 ± 0.10	3.70 ± 0.22	7.42 ± 0.49	14.91 ± 0.92
	3	1.97 ± 0.11	3.71 ± 0.21	7.26 ± 0.45	14.63 ± 0.93
	4	1.96 ± 0.12	3.68 ± 0.23	7.25 ± 0.45	15.15 ± 0.99
	5	1.88 ± 0.15	3.92 ± 0.36	7.66 ± 1.09	15.48 ± 2.55
	6	1.94 ± 0.11	3.65 ± 0.22	7.11 ± 0.44	14.38 ± 1.01
	7	1.96 ± 0.09	3.58 ± 0.22	7.01 ± 0.48	13.86 ± 0.88
	8	1.95 ± 0.11	3.59 ± 0.24	7.02 ± 0.45	14.12 ± 0.96
	9	1.83 ± 0.13	3.71 ± 0.47	7.11 ± 1.02	14.22 ± 2.32
	10	1.94 ± 0.09	3.51 ± 0.20	6.64 ± 0.43	13.76 ± 0.89
	11	1.93 ± 0.09	3.50 ± 0.20	6.68 ± 0.42	13.87 ± 0.84
	12	1.93 ± 0.10	3.50 ± 0.20	6.62 ± 0.40	13.56 ± 0.84
	13	1.94 ± 0.11	3.54 ± 0.20	6.64 ± 0.39	13.75 ± 0.80
	14	1.94 ± 0.10	3.53 ± 0.21	6.66 ± 0.40	13.68 ± 0.77
	15	1.93 ± 0.11	3.56 ± 0.22	6.71 ± 0.40	13.75 ± 0.75
	16	1.94 ± 0.11	3.58 ± 0.19	6.80 ± 0.40	14.14 ± 0.85
	17	1.94 ± 0.11	3.59 ± 0.29	7.03 ± 0.42	14.44 ± 0.88
	18	1.95 ± 0.10	3.61 ± 0.25	7.01 ± 0.39	14.55 ± 0.80
	19	1.74 ± 0.08	3.64 ± 0.25	7.32 ± 0.51	15.15 ± 1.17
	20	1.87 ± 0.10	3.62 ± 0.21	7.20 ± 0.34	14.69 ± 0.77
	21	1.87 ± 0.10	3.65 ± 0.19	7.21 ± 0.35	14.55 ± 0.77
	22	1.92 ± 0.09	3.77 ± 0.17	7.45 ± 0.33	15.02 ± 0.76

A.17 LEARNED L_p-MASK PROPERTIES (p=16)

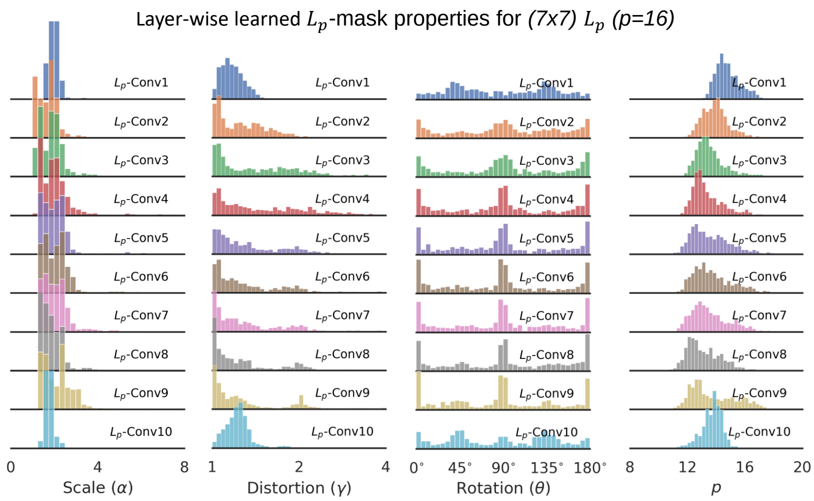

Figure 17: **Layer-wise distribution of learned L_p-mask properties (p=16)**

A.18 STATISTICS FOR COMPARISON OF OPTIMIZED MPND p VALUES IN ARTIFICIAL AND BIOLOGICAL RFS

Table 8: Holm-Bonferroni corrected multiple comparisons of each RFs using Welch's t-test. Lower diagonal elements denote corrected statistical p-values and upper diagonal elements denote degree of significance (n.s, not significant; *, 0.01<p<0.05; **, 0.001<p<0.01; ***, p<0.001;)

Welch's t-test (corrected p-values)	**Mouse V1 Layer 2/3**	**Trained + Image**	**Trained + Noise**	**Untrained + Noise**	**Untrained + Image**
Mouse V1 Layer 2/3	-	*	*	***	***
Trained + Image	1.10e-2	-	n.s	***	***
Trained + Noise	1.06e-2	7.6e-1	-	***	***
Untrained + Noise	1.23e-15	4.13e-28	1.49e-25	-	***
Untrained + Image	8.94e-14	1.59e-25	6.73e-23	1.39e-22	-

A.19 IMPACT OF L_p-CONVOLUTION (P=2) ON OTHER CNN ARCHITECTURES

Table 9: Top-1 performance (mean±std, 5 trials) on the CIFAR-100 datasets with L_p-Convolution applied in ConvNeXt-V2-T (Woo et al., 2023), ResNet-50 (He et al., 2016), ResNext-50 (Xie et al., 2017) and DenseNet-121 (Huang et al., 2017). The symbol ✓ indicates L_p-Converted ($p_{\text{init}} = 2$) or not. '***' denotes statistical comparison using Welch's t-test ($p < 0.001$).

	CIFAR-100			
L_p-Conv	**ConvNeXt-V2-T**	**ResNet-50**	**ResNeXt-50**	**DenseNet-121**
-	64.26 ± 0.41	73.17 ± 0.23	73.55 ± 0.57	74.12 ± 0.16
✓	*** 65.58 ± 0.25	*** 76.66 ± 0.19	*** 77.38 ± 0.36	*** 77.14 ± 0.18

A.20 RECEPTIVE FIELD

The term RF was initially confined to a specific area impacting a single neuron in the visual system, as outlined by Sherrington in 1906 and later by Hartline in 1938 (Sherrington, 2023; Hartline, 1938). It was defined as a distinct region in visual space capable of triggering electrical responses in retinal ganglion cells, thus highlighting immediate and localized neural interactions.

Modern interpretations, however, have substantially expanded the scope of receptive fields. The groundbreaking work of Hubel and Wiesel exemplifies this evolution, revealing how receptive fields process complex visual patterns through multiple layers in the primary visual cortex (V1). This progression from Hartline's narrower viewpoint to a more all-encompassing approach mirrors the intricate, multi-layered nature of sensory processing in the brain. The present-day definition, shaped by Hubel and Wiesel's insights, underscores the dynamic, multi-dimensional nature of neural responses.

In CNN, the concept of a receptive field aligns somewhat with Hartline's original idea, focusing mainly on local input processing (Fukushima, 1980; LeCun et al., 1989). In CNNs, RF typically denotes the localized interactions among successive layers, reflecting Hartline's emphasis on localized sensory inputs. **In our analysis, we align with a more restricted definition of the receptive field, akin to the original concepts from Sherrington and Hartline.** This approach corresponds to the local connectivity between adjacent layers in CNNs, aiming to provide a clearer and more focused understanding in both biological and computational contexts.

A.21 PyTorch-style pseudocode for L_p-Convolution

Algorithm 1 PyTorch-style pseudocode for L_p-Convolution

```
import torch
import torch.nn as nn
import torch.nn.functional as F

from torchvision.models import alexnet

class LpConv2d(nn.Conv2d):
    def __init__(self, p_init, sigma_init, in_channels, out_channels, kernel_size, stride, padding,
**kwargs):

        # Create parameters p & C
        params_p = torch.ones( out_channels ) * p_init
        params_C = torch.zeros( out_channels, 2, 2 )
        params_C[:,0,0] = 1/sigma_init
        params_C[:,1,1] = 1/sigma_init
        self.p = nn.Parameter( params_p )
        self.C = nn.Parameter( params_C )

    def forward(self, input):
        # Create channel-wise lp_masks from parameters p and C
        lp_masks = get_channel_wise_lp_masks(self.p, self.C)

        # Overlay lp_masks on weight
        masked_weight = weight * lp_masks
        return F.conv2d(input, masked_weight, bias, kernel_size, stride, padding, **kwargs)

def LpConvert(model, p_init):
    # Convert all nn.Conv2d layers into LpConv2d
    for i in range(num_layers):
        layer = model.layers[i]
        if isinstance(layer, nn.Conv2d):
            model.layers[i] = LpConv2d(
                p_init,
                sigma_init,
                in_channels=layer.in_channels,
                out_channels=layer.out_channels,
                kernel_size=layer.kernel_size,
                stride=layer.stride,
                padding=layer.padding,
                **layer.extra_args)
    return model

# Example LpConvert on Alexnet for TinyImageNet
base_model = alexnet(num_classes=200)
lp2_model = LpConvert(base_model, p_init=2)
```

A.22 Comparison of ViT and Lp2-CNNs

Model	Top-1 (%)	FLOPs (G)	Params (M)
ViT-32x32	49.88	4.37	87.6
ViT-16x16	**54.20**	16.87	86.0
AlexNet	52.25	0.71	57.82
Lp2-AlexNet	**54.13**	3.41	68.6
Lp2-VGG-16	69.96	83.74	200.5
Lp2-ResNet-18	68.45	9.86	61.5
Lp2-ResNet-34	70.43	19.93	116.6
Lp2-ConvNeXt-T	70.72	5.42	33.8

Table 10: Lp2-AlexNet achieves comparable performance to ViT-16x16 on TinyImageNet with significantly lower parameter counts and computational cost, demonstrating its efficiency.

A.23 The justification for using large, sparse kernels

The use of **large kernels** enables the model to **cover the input space more effectively** with fewer layers compared to smaller kernels (Ding et al., 2022; Luo et al., 2016). However, simply increasing the kernel size does not guarantee performance improvements, as shown in Table 1 (Base vs. Large). This is presumably due to larger kernels inadvertently incorporating irrelevant global information, which can hinder performance compared to smaller kernels that rely on locality inductive biases to extract local features hierarchically. This is where sparsity plays a key role. We introduce **sparsity** constraints to optimize the usage of large kernels, ensuring they **focus on only relevant global information** while mitigating the disadvantages of naïvely expanding kernel sizes, supported by Sudoku experiments.

A.24 Evidence for Gaussian Sparsity as Biological Constraints

A.24.1 1.Theoretical Evidence

Sparse Coding Theory : Sparse Coding Theory posits that neural systems optimize sensory representations by minimizing redundancy. Learning a sparse code for natural images leads to the emergence of simple-cell receptive field properties (Olshausen & Field, 1996). This process can be linked with Gaussian priors, where synaptic weights follow a Gaussian distribution with most connections being weak and a few strong, promoting efficient information encoding (Olshausen & Millman, 1999).

Effective Receptive Field (ERF) Theory : In convolutional neural networks, the actual influence of input pixels on an output neuron decreases in a Gaussian manner from the center of the theoretical receptive field (Luo et al., 2016). This means that while the theoretical receptive field defines the maximum possible area of influence, the ERF is effectively smaller and Gaussian-shaped, with central pixels contributing most significantly to the neuron's output.

A.24.2 Empirical Evidence

Supporting References : These two references demonstrate both anatomical and functional distribution of synapses predominantly following a Gaussian-like distribution in the visual cortex (Hellwig, 2000; Rossi et al., 2020).

Analysis Result : We have demonstrated Gaussian distribution with the in vivo functional synapse data (**?**)in alive mouse V1 in Appendix A.3.