Organic Convolution Model of Ventral Visual Path Reproduces Curvature Selective Shape Tuning Found in Area V4

Carl Gold[1]

[1]Independent Researcher, Berkeley, CA, USA.

Contributing authors: carl.steven.gold@gmail.com;

Abstract

This modeling study investigates whether an orderly convergence of neuronal selectivities from cortical areas V1 and V2 can produce curvature selective receptive fields found in area V4. A model of the ventral visual pathway from V1 to V4 is composed of approximately 500,000 individual integrate and fire units. The V1 and V2 models are based on recent findings about the composition of V2 receptive fields. A novel proposal is made for how V4 neurons may create selectivity for varying degrees of local curvature through the orderly convergence of afferent inputs from V2. The study employs a novel method for simulating individual spikes in large numbers of model neurons using tensor programming and GPU hardware: Assuming that convergent functional micro-architectural patterns repeat in topographically organized visual space, the details of individual unit depolarization and spike time is modeled using convolution operations combined with a model for the time course of post-synaptic potentials. The few parameters in the model are set manually, and this is sufficient to qualitatively reproduce the V4 recordings. This demonstrates that convolution technologies can be used for the realistic simulation of large numbers of neurons at an intermediate level of biological detail.

Keywords: Spiking, Convolution, Neural Network, V4

1 Introduction

1.1 Visual Area V4

Visual object identification is processed in the ventral visual pathway through primary visual areas V1 and V2, and an intermediate area V4, before object recognition is processed in area IT, or inferior temporal cortex (IT) [29, 50]. It has been known for many years that there are neurons in area V1 that encode local orientation [16, 20]. A recent study using natural images and extended projection pursuit regression [43] found that V2 contains neurons in which V1 receptive fields converge into either "ultralong" (aligned) or "complex-shaped" combinations (multiple oriented sub-unit) [34]. The functional properties of V4 are less well understood: studies suggest that area V4 occupies an intermediate level in the hierarchy of visual processing, responding to moderately complex stimuli, but there is not consensus on the precise nature of V4's processing [45].

Some researchers have observed that V4 neurons seem tuned to respond to different degrees of curvature. One study used fast reverse correlation sequences of stimuli to map the orientation and curvature response of individual neurons in local regions throughout the receptive field [39]. As illustrated in Figure 1, the stimuli consisted of oriented bars combined to form approximate curves ranging from straight to tightly curved. It

was found that the V4 neurons responsive to these stimuli can be categorized into those selective for low curvature, medium curvature and high curvature. Only a subset of V4 neurons were found to have spatially invariant shape tuning: V4 neurons selective to medium and high curvatures had spatially variable tuning in stereotypical patterns, exemplified by the units illustrated in Figure 1. While some studies advocate using natural images or videos as stimuli to understand visual cortex (e.g. [8, 26] which studied V1 as will be described below), the results in [39] are exceptional for the purpose of modeling receptivity to *oriented* components of visual stimuli because they detail the response of neurons to mutually exclusive and systematically exhaustive combinations of positions, orientations and curvature stimuli parts.

Another recent study supports the notion of orientation and curvature tuning in V4: [19] used intrinsic optical imaging of V4 and found a pattern of curvature domains that were tuned to respond to either straight, medium curvature or high curvature. In [19] the stimuli consisted of conventional straight line gratings along with gratings having different degrees of curvature, all presented at different orientations. The results in [19] are complimentary to the results in [39], suggesting that neurons sensitive to different degrees of curvature are in fact central to the functioning of V4.

Other researchers have found that V4 neurons are responsive to simple shapes made from combinations of convex and concave boundaries and often respond to a single preferred stimuli throughout the receptive field [40]. At the same time, studies using similar methods found that some V4 neuron seem to prefer texture [15]. More recently, it was shown that a given convex or concave shape preference tends to be selective for similar stimuli with a range of sizes [25].

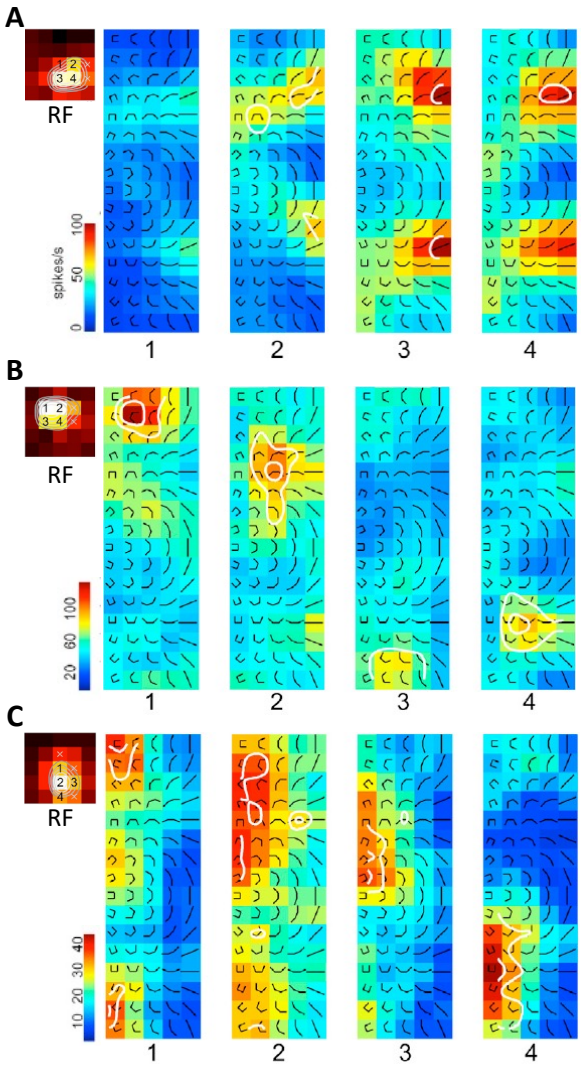

Fig. 1 Location specificity of shape tuning in typical V4 neurons, reproduced from [39] with permission. **A** Low-curvature selective. **B** Medium curvature selective. **C** High curvature selective. 1-4: shape tuning for curvature (x-axis) and orientation (y-axis) at indicated positions in the receptive field (RF).

1.2 Deep Neural Networks

In the past decade, deep convolution neural networks (DCNN's) have dramatically improved performance in artificial intelligence (AI) visual recognition applications using stochastic gradient descent training algorithms broadly known as deep learning (DL) [32]. However, the ideas behind DCNN are not new [31]. In fact, the idea of convolution is based on observations about the ventral visual pathway: First, there is the observation that selectivity to simple oriented stimuli in earlier layers seem to repeat more or less interchangeably over positions in topographically organized visual space. Also there is an alteration of approximately linear filter operations with non-linear pooling operations [11].

There has been considerable interest among computational neuroscientists in whether DCNN and DL can be the basis for explaining the object recognition pathway [23, 52]. However, there are

questions as to whether DL models can serve as models for the physical processes in the central nervous system: DL requires feedback carrying gradient information with the same weight as the feedforward signal, but such signals have not been observed [5]. It has also been noted, even by DL researchers, that no learning system can guarantee that the same weights will be learned by neurons at different locations [3] raising questions about the plausibility of the convolution model. DL models ignore the fact that organic neurons emit binary spikes and instead rely on non-linear transfer functions, which are described as representing rate codes. But it has been shown that object recognition in humans and animals is too fast to use a rate code [49], and DL research has not produced a biologically plausible model for object recognition using fast, spiking computation. DL models also generally ignore the fact that real neurons are constrained by their type to have entirely excitatory or entirely inhibitory influence on efferent connections [12]. At the same time, there have recently been some serious attempts to investigate whether biological learning mechanisms like spike timing dependent plasticity (STDP) in a rate coded model can implement biological deep learning [6, 22].

A more meta problem with DL as a mechanistic explanation of brain function is the assumption that a neural network is initially in a random state and the ensuing structure is the result of a learning algorithm. As described in [53], evolution puts competitive demands on new born organisms and organic brains must be "initialized" in a highly non-random manner: The initial state of organic neural networks contains enough structure to allow perception, basic sensory processing, reflexes, as well as the ability learn more detailed information from the environment. [53] also shows that the structure of brains must be encoded concisely in the genome, because there is not enough information storage in the genome to encode even a small fraction of the necessary connections.

A few studies have attempted to use DL techniques to understand neurophysiological data from V4: [42] examined a state of the art DL model and found units that respond to the shapes used in [40]. [52] proposed a goal driven approach and trained a DL model to match the response of IT neurons; the network trained to match IT also matched V4 response properties. These studies are described by their authors as "top-down" in the sense that they create models to perform the functions of higher cortical areas and then try inspect the resulting micro-architecture to gain insight into lower cortical areas. This is in contrast to neurophysiology which is described as "bottom-up" in that it aims to understand the functional micro-architectures of lower cortical areas directly. [8] also took the goal driven approach by analyzing a pre-trained DCNN trained with DL and added output non-linearities and noise to the unit activations to come up with spike counts. They found that intermediate convolution layers in the network best match V1, but did not describe relationships between the model and other visual areas.

Other recent studies also used DCNN with DL to model cortical area V1: [54] compared a variety of models including Gabor filters and a DCNN trained with DL, to model the response of V1 cells to complex artificial stimuli. They found that DL gives the best match to the response of individual units and suggest that V1 neurons respond to more complex shapes than just oriented bars. [26] also used DL to predict the firing rate of recordings of V1 neurons in response to natural images. They found the predictions were accurate if they used an additional "all-to-all" layer as part of the training. However, the top-down modeling and firing rate matching studies do not address the biological implausibilities of the underlying DL modeling approach. So there remain questions as to whether DL models can serve as a mechanistic explanation for the physical processes underlying perception in the ventral visual stream.

Apart from DL models, one early example of a mathematical model using alternating selectivity and invariance operations was the HMAX model [44]. The same model was later extended specifically to match known V4 shape tuning [9]. Another example of a mathematical model for V4 is the spectral receptive field model which describes units by orientation and spatial frequency functions [13].

1.3 Organic Convolution

The aim of this study is to determine if simple convergent micro-architectural patterns modeled after observations of V1, V2 and V4 can reproduce

recordings showing fine curvature and orientation tuning observed in a sample of V4 neurons [39]. The model microarchitectures are intended to meet the criteria of [53] in that they are simple and could be encoded efficiently in the genome. Organic convolution (OC) is the name for the model because it tries to synthesize DCNN and neurophysiology with an emphasis on agreement with neurophysiological principles and biophysics.

In OC, simple micro-architectural connection patterns are assumed to repeat across the topographically organized visual areas to provide a foundation for visual processing. If this is the case then forms of learning other than DL may suffice to complete the picture of cortical processing, but biologically plausible learning mechanisms are beyond the scope of this study. In OC the mechanism for micro-architectural pattern formation is assumed to be a genetically controlled process that guides local micro-architectural formation.[1] The concerted action of local development rules would enable neurons at distributed locations to have approximately the same patterns of connections, which satisfies a major objection to convolution like computations occurring in organic brains.

This study is also demonstrates how tensor based computation technologies used for DCNN DL experiments can be used for experiments modeling individual spikes, rather than relying on rate code abstractions: The model, named Spike Time Convolution (STC), adds the time course of post synaptic potentials (PSP's) onto a standard DCNN model, as described in detail in the Methods. But note that OC does not necessarily imply STC because the principals of OC can be explored in simplified binary models as well; STC adds a higher level of realism.

[1]Studies have already demonstrated this kind of structure : Pyramidal cells in mouse V1 display orientation selectivity at the time of *first eye opening* [28]. Both mammalian [24] and fly [7] visual systems detect motion by having specific subtypes of bipolar cells, each having a different characteristic delay, connect at different preferred distance along each dendrite of the receiving amacrine cell. A variety of pathways and signals control the shape of axonal structures and even the specific connections between neurons of different types[38] as well as the specific subtypes making up topographically mapped neuronal structures [46]. Also, spontaneous waves of activity originating from the retina in utero [1] are now implicated with having an important role in shaping normal neuronal development. [37] proposed a model in which the receptive fields of V1 simple cells result from the spatial structures of the correlations in neural activity among the competing inputs in spontaneous activity.

The network architecture is summarized in Figure 2. The functional micro-architectural models used for V1 are based on well known receptive field properties [20] and the models for V2 are based on suggestions in a recent study [34] as described below. The convergent patterns leading to V4 selectivity from V2 afferents are novel proposals: V4 curvature selectivity may be created by convergent patterns giving preference to complex-shaped V2 afferents having either medium or high sub-unit separation angle, and having an orientation that is convex around the midpoint of the receptive field. The latter property requires that dendrites are orientation selective in their afferent connection preference; and the preference is based on the dendrites own orientation in the topographic visual map.

As detailed in the Methods, the first layer of the network is an input layer that serves as a lumped model of the visual pathway from the retina to V1 L4 (Figure 2B). The V1 L4 model layer contains both excitatory and inhibitory units for each input location that have independent random spike arrival times within a window of a few milliseconds [47]. As a result, the V1 L2/3 model (Figure 2C), the first convolution layer, has both excitatory and inhibitory inputs for the convolution filters, albeit on separate channels with different random times. With this structure, the model follows Dale's law and aligns with the facts that inhibitory units are most common in L4, along with a large population of stellate excitatory cells [21]. The L2/3 V1 layer contains excitatory neurons (putative pyramidal cells) with receptive fields selective for "simple" oriented bars as first described by Hubel and Weisel [20]. The micro-architecture to create an oriented bar selective neuron was first proposed in [20] and is to have a line or stripe of excitatory afferents surrounded by inhibition. In the literature these receptive fields are often described as Gabor filters in reference to mathematically defined filters (products of Gaussian and sinusoidal signals) for signal processing [35].

The model includes a layer representing V2 L4 (Figure 2E) which separately pools each type of inputs from V1 L2/3 and includes both excitatory and inhibitory units that are afferents to V2 L2/3. The model for V2 L2/3 units (Figure 2F) is based on the findings in [34], which used a

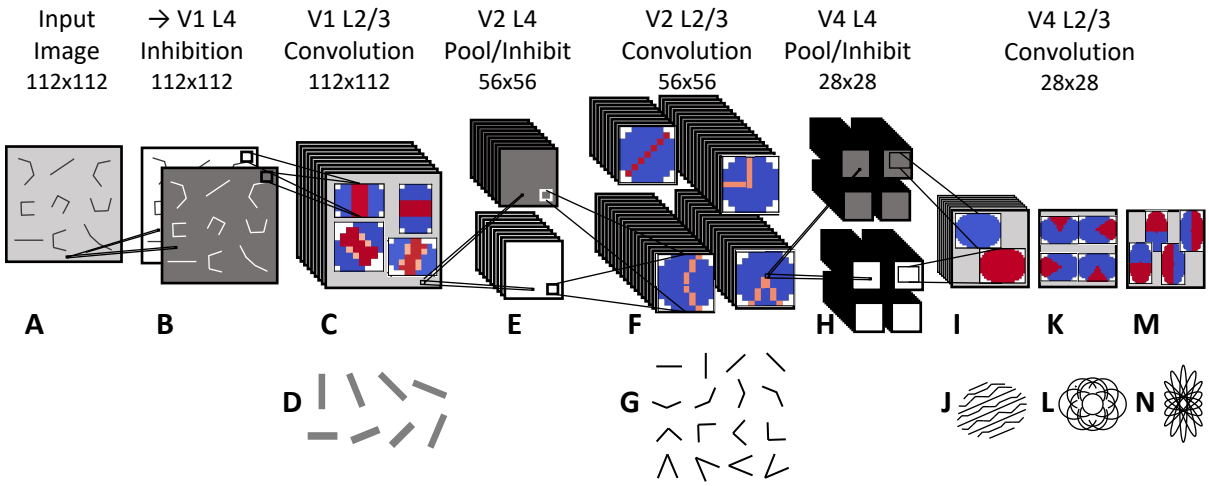

Fig. 2 Organic Convolution Model Architecture **A** Input **B** Layer 1 is a lumped model for visual processing from the Retina to V1 L4 including both excitatory and inhibitory units. **C** V1 L2/3 Model consists of 8 simple oriented detection convolution filters, as detailed in Figure B4. **D** Preferred stimuli for the V1 L2/3 units. **E** The V2 L4 model contains pool-like operations with separate excitatory and inhibitory units. **F** The V2 L2/3 model consists of convergent patterns based on recordings of ultralong and complex-shaped units [34] converted into convolution filters as shown in Figure B5. **G** Preferred stimuli for V2 L2/3 units. **H** The V4 L4 model is the same as the V2 L4 model, operating on the V2 L2/3 units outputs. **I** Convolution filters for the V4 low curve selective units, as detailed in Figure 4. **J** Preferred stimuli for V4 Low curvature selective model units. **K** Convolution filters for the V4 medium curve selective units, as detailed in Figure 6. **L** Preferred stimuli for the V4 medium curvature selective units. **M** Convolution filters for the V4 high curve selective units, as detailed in Figure 8 **N** Preferred stimuli of the V4 high curvature selective units.

linear-nonlinear model [11] and projection-pursuit regression [43] to map the receptive fields of V2 neurons. [34] found that some V2 units had receptive fields they described as "ultralong" because they had Gabor-like receptive fields but high aspect ratio. Another type of neuron identified in [34] are described as "complex-shaped" because they had a small number of Gabor-like receptive field sub-units with each sub-unit at a different orientation; the preferred stimuli of the V2 L2/3 units are illustrated in Figure 2G. [34] proposed that simple convergent aggregation of V1 Gabor-like receptive fields would achieve these selectivities, and more details of the implementation used here are provided in the Supplementary Methods. The model also includes an L4 model for V4 (Figure 2H) in the same manner as in V2: Duplication of the pooling operation with inhibitory units. [2]

[2]These receptive field properties are similar to the computational units in [44] and the resulting model at this stage is a spiking convolutional version of the HMAX model.

2 Results

2.1 V4 Low Curvature Selective Model

Figure 3 compares the shape tuning of the V4 L2/3 neuron model with the recording. For both recording and simulation, the most significant activity for the unit occurs at 3 stimuli locations. The most preferred stimuli are straight or minimally curved and at a small angle counter clockwise from horizontal. However, there is broad tuning around the most preferred stimulus and non-zero activity as high as 50% of the maximum for stimuli that are at orientations as much as 22.5° away and at medium curvatures.

The main deficiency of the model in reproducing the recording is that the model does not have very weak responses (25% of the maximum) to as wide a range of orientation and curvatures as is seen in the recording. Also, the model neuron does not have any responses in the stimuli locations outside of top 4, while the recordings show weak responses. Lower background firing is most likely due to the simple noise model used, as detailed in the Methods. The response of the in vivo neurons to positions outside the main receptive field may

be due to lateral connections that are not part of the model.

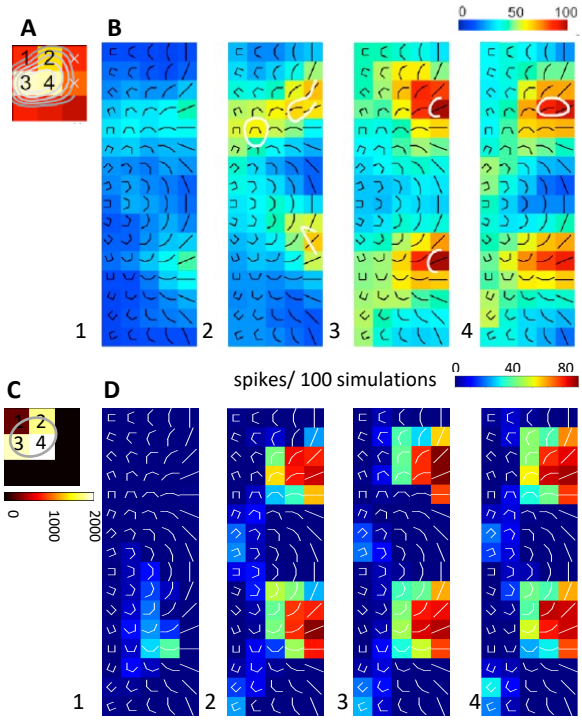

Fig. 3 Low curvature selective unit recording (top) and simulation (bottom) **A** Receptive field for recording. **B** Shape tuning for recording, 1-4 show the shape tuning for curvature (x-axis) and orientation (y-axis) at indicated positions in the receptive field (A). **C** Receptive field for simulation. The approximate location and extent of the simulated neurons dendritic tree in the topographic space of the afferent inputs is shown. **D** Shape tuning for simulation, as in B, at positions in the simulated receptive field (C). The simulation qualitatively reproduces the main features of the low curvature selective neuron recording.

The convergent functional micro-architectural pattern for the V4 L2/3 unit that produces the simulation result in Figure 3 is shown in Figure 4 (it was shown in summary in Figure 2I). Figure 4A shows a pictogram representation of the micro-architectural pattern, while Figures 4B and C show the convolution filters with which the pattern is modelled.. The pattern is to make connections to only those afferent inputs that are selective for ultralong or complex-shapes with slightly angled (22.5° to 45°) sub-units. Also, excitatory connections are made for afferents with a preferred orientation of 22.5° from horizontal, but with broad tuning including orientations from 0° to 45°. Afferents from inputs selective for other orientations of nearly straight stimuli are accepted from inhibitory L4 units, as shown in Figure 4D, while afferents from inputs selective for more highly angled complex-shaped units are ignored (no connections, or zero weight).

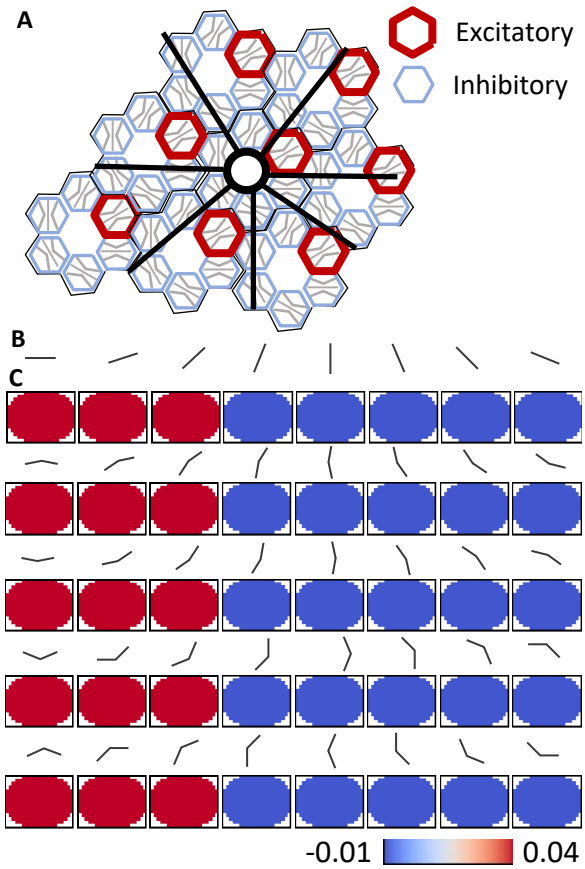

Fig. 4 Functional micro-architecture for V4 L2/3 neurons selective for low curvature. **A** Illustration of a functional micro-architecture in which dendrites pass within afferent inputs with different orientations and complex-shaped angles and select only those inputs within a range. **B**. Pictorial representation of different orientation and complex-shaped input preferences. **C** 15x20 convolution filters for the different channels of inputs.

In comparison to the model neurons for V1 and V2, the V4 model neuron has a relatively large receptive field: 15x20 in the reduced frame of V4, equivalent to 60x80 in the original 112×112 pixel visual field model. (In comparison, the V1 neurons have receptive fields of 6×9 and the V2 neurons have fields of 18×18 in the frame of V1.) This large receptive field is required to match the

recordings, as shown in Figure 3 and the model does not match the weak secondary stimulation.

Due to the large receptive field, the total excitation in the V4 L2/3 units is much larger than in the V1 and V2 functional micro-architectures. The total excitation for the low curvature selective V4 L2/3 neuron model is approximately 164, in units of the neurons threshold, while the total inhibition is 52. The high amount of excitation is required to match the recordings: As can be seen from Figure 3, a small stimulus in a fraction of the neurons receptive field is sufficient to cause a spike. Because the response is relatively uniform in the receptive field, it follows that there must be many times more excitative inputs available than what is required to respond to a small stimulus. The preferred stimuli were illustrated in Figure 2I: it is a classic parallel line grating, but with sone caveats: it allows imperfectly straight lines, and it requires a matching stimuli in only a small portion of the receptive field to trigger a spike.

2.2 V4 Medium Curvature Selective Model

The output for a model medium curve selective V4 L2/3 unit is shown in Figure 5 and compared to the recording from [39]. For both the model and recording the receptive field is highly varied over multiple stimuli locations. There is strong response to medium curvatures, though also broad tuning that includes activation to straight and highly curved stimuli. The orientation of the most preferred stimuli are convex around the center of the receptive field. The main shortcoming of the model to reproduce the recording is that the orientations of some of the preferred stimuli are different by around 10°-20°, and also the model lacks background firing to the less preferred stimuli found in the recordings.

The convergent functional micro-architecture for V4 L2/3 neurons with medium curvature selectivity (Figure 5) is illustrated in Figure 6. Figure 6A shows a pictogram representation of the micro-architectural pattern, while Figures 6B and C show the convolution filters with which the pattern is modelled.. In this pattern, dendrites connect to excitative afferent inputs in the mid-range of complex-shaped angles from 45° to 90°. To match the preference of the neuron for inputs that are convex around the receptive field the

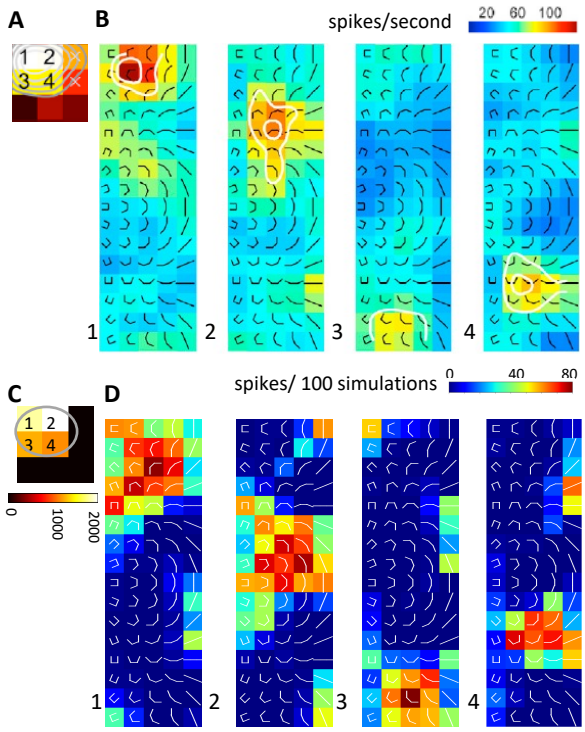

Fig. 5 Medium curvature selective unit recording (top) and simulation (bottom) **A** Receptive field for recording. **B** Shape tuning for recording, 1-4 show the shape tuning for curvature (x-axis) and orientation (y-axis) at indicated positions in the receptive field (A). **C** Receptive field for simulation. The approximate location and extent of the simulated neurons dendritic tree in the topographic space of the afferent inputs is shown. **D** Shape tuning for simulation, as in B, at positions in the simulated receptive field (C). The simulation qualitatively reproduces the most significant features of the medium curvature selective neuron recording.

model neuron dendrites use a connection pattern based on the dendrite's own direction in the topographically arranged cortical layer: The preference is to connect with excitatory complex shape selective afferent inputs that have the long dimension of their preferred stimuli perpendicular to the dendrites own direction, and the complex shape angle open (convex) to the center of the receptive field.

If the orientation of the complex shape is outside the preferred range, then a connection with an inhibitory afferent is formed. As shown in Figure 6 the tuning for orientation is broad and any given dendrite will make an excitatory connection to afferent inputs with orientations in a range of around 70°. As illustrated in Figure 6A, the preferred inputs for the medium curvature selective V4 L2/3 units are convex around the

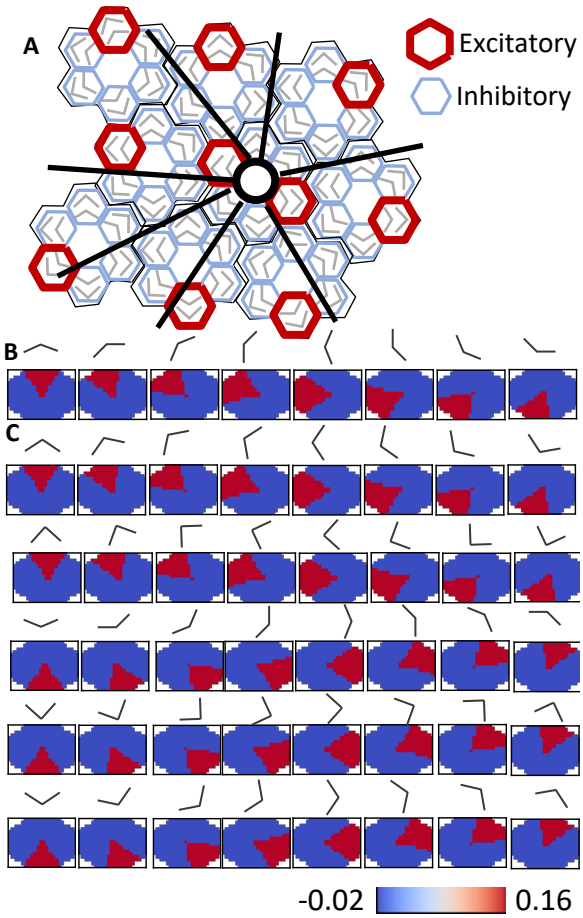

Fig. 6 Functional micro-architectural pattern for V4 L2/3 neurons selective for medium curvature. **A** Illustration of a functional micro-architecture in which dendrites travel through afferent inputs with moderately angled complex shaped selectivity and a range of orientations and select those inputs with angles open towards the center of the receptive field. **B** Pictorial representation of selectivity of V4 L4 inputs, having medium angle complex-shapes and a range of orientation. **C** 20×24 convolution filters for the different channels of afferents.

midpoint of the receptive field. The overall preferred stimuli set for the pattern were illustrated in Figure 2L: it is a set of overlapping curves that are individually narrower than the overall receptive field. As was the case for the low curvature selective V4 neurons, partially matching stimuli trigger a spikes indicating excitative capacity in excess of the threshold. The total excitation for the medium curvature selective V4 L2/3 neuron model is approximately 507 (in units of the neurons threshold), while the total inhibition is 185.

2.3 V4 High Curvature Selective Model

The output of a high curvature selective area V4 L2/3 model neuron is shown in Figure 7, and compared with the high curvature selective recording from [39]. For this neuron, the stimuli locations triggering the highest response lie along a vertical strip of three locations (Fig 7B1, 7B2 and 7B4), with one additional high response receptive field location on the right side of the main strip (Fig 7B3). The neuron responds primarily to highly curved stimuli, with broad tuning that includes moderate responses to low curvature and straight lines, especially for the most responsive stimuli locations (Figure 7B2).

The orientation of the preferred stimuli varies by stimuli location. Like in the medium curvature selective cell, the response tuning curve at each stimuli location is broad but it peaks around curves oriented convexly towards the center of the receptive field: Location 1 (top) is more strongly tuned to curves open downward, location 2 (center) is tuned strongly for both upward and downward facing curves, location 3 (on the right) is responsive to left opening curves and location 4 (on the bottom) is responsive to upward facing curves. But the convex orientation of the most preferred stimuli does not appear consistent because of the broad tuning of the response.

The model response is qualitatively similar to that of the recording in the preference for high curvatures and it approximately matches the orientation selectivity of the recording as well. The main mismatch between the model and recording is that the real neuron shows broader tuning to orientations with high curvature. The recording also shows a weak response (around 25% of the maximum) for low curvature stimuli more frequently than the simulation.

The convergent functional architecture for the high curvature selective V4 L2/3 cell model is shown in Figure 8. Figure 8A shows a pictogram representation of the micro-architectural pattern, while Figures 8B and 8C show the convolution filters with which the pattern is modelled.. This is essentially the same micro-architectural pattern as for the medium curvature selective units, although matching higher angle complex-shaped inputs and with broader orientation tuning: Dendrites in the high curvature selective model connect to high

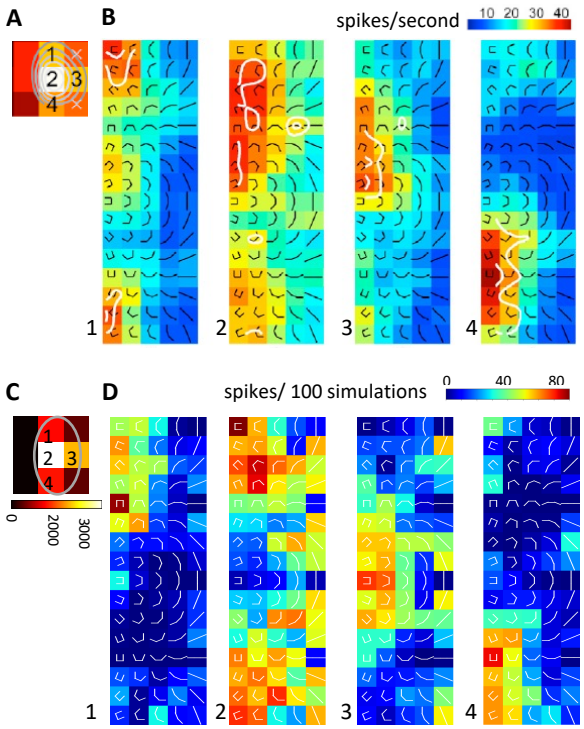

Fig. 7 High curvature selective unit recording (top) and simulation (bottom) **A** Receptive field for recording. **B** Shape tuning for recording, 1-4 show the shape tuning for curvature (x-axis) and orientation (y-axis) at indicated positions in the receptive field (A). **C** Receptive field for simulation. The approximate location of the simulated neurons dendritic tree in the topographic visual field is shown. **D** Shape tuning for simulation, as in B, at positions in the simulated receptive field (C).

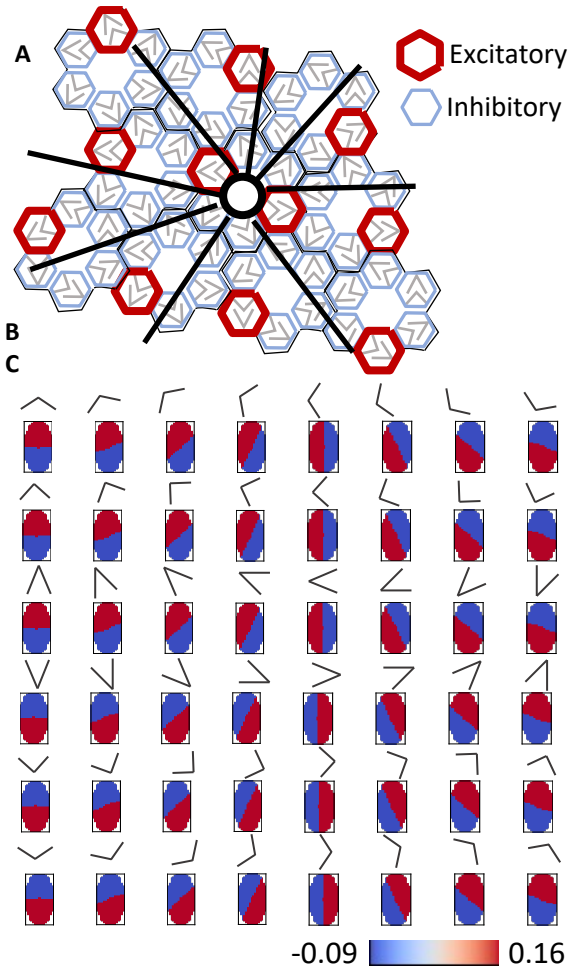

Fig. 8 Functional micro-architectural pattern for model V4 L2/3 neurons selective to high curvatures. **A** Illustration of a functional micro-architecture in which dendrites travel through afferent inputs with highly angled complex shaped selectivity and a range of orientations, and selecting those inputs with angles open towards the center of the receptive field. **B** Pictorial representation of selectivity of V4 L4 afferents, having high angle complex-shapes and a range of orientation. **C** 16×32 convolution filters for the different channels of afferents.

angle (67.75° to 112.5°) complex-shaped V2 afferents and prefers those oriented convexly around the midpoint of the receptive field. However, there is broad orientation tuning. Non-preferred orientations are inhibited and low to medium angle complex shape preferring afferents are ignored. A summary of the preferred stimuli set was shown in Figure 2N: narrow curves oriented convexly around the receptive field center. The total excitation for the high curvature selective V4 L2/3 neuron model is approximately 1770 (in units of the neurons threshold), while the total inhibition is 870.

2.4 Spike Propagation

The distribution of spike times resulting from the model is illustrated in Figure 9**A** which shows box plots for the spike times for each layer of the spiking convolutional neural network. As described in the Methods, the simulation assumes zero propagation delay between cortical layers and areas to maximize memory efficiency. Note that equal delays between all units in one layer and the next has no impact on the computational model. In the model without delays, it takes a median of 8.2 ms for the spiking wave to reach the V4 L2/3 layer, and a max of 12ms; more than 99% of spikes that will arrive have arrived by 10ms and 10ms was used as the time limit for the simulations shown in Figures 3, 5 and 7.

Figure 9**B** shows the resulting spike times assuming delays between cortical layers and areas: A 1 ms delay is used for spikes to travel between cortical layers, assuming 0.5 m/s conductance velocity for un-myelinated axons [41, 48] and 0.5 mm distance between L4 and L2/3 in macaque visual cortices [51]. A 4 ms delay is used for spikes to travel between cortical areas [10, 48]. Under these assumptions it takes around 20 ms for a spike wave to propagate from the model V1 L4 input layer to V4 L2/3.

Figure 9**C** shows the percentage of neurons spiking in each layer, normalized by the proportion of stimuli in the neurons receptive fields: For the V1 and V2 models, each neuron has at most 1 out of 9 stimuli in its receptive field using the protocol specified in [39], while in V4 each unit has a receptive field covering on average 5 out of 9 stimuli. The population spike rates are 5-13% in V1 and V4 but they are an order of magnitude lower in the V2 layer, due to the high number of variations on the complex-shaped receptive field patterns. However, the population response rates in the model are probably not predictive of overall population activity in vivo activity because the model is only designed to reflect a narrow range of neural activity in response to a specific class of stimuli.

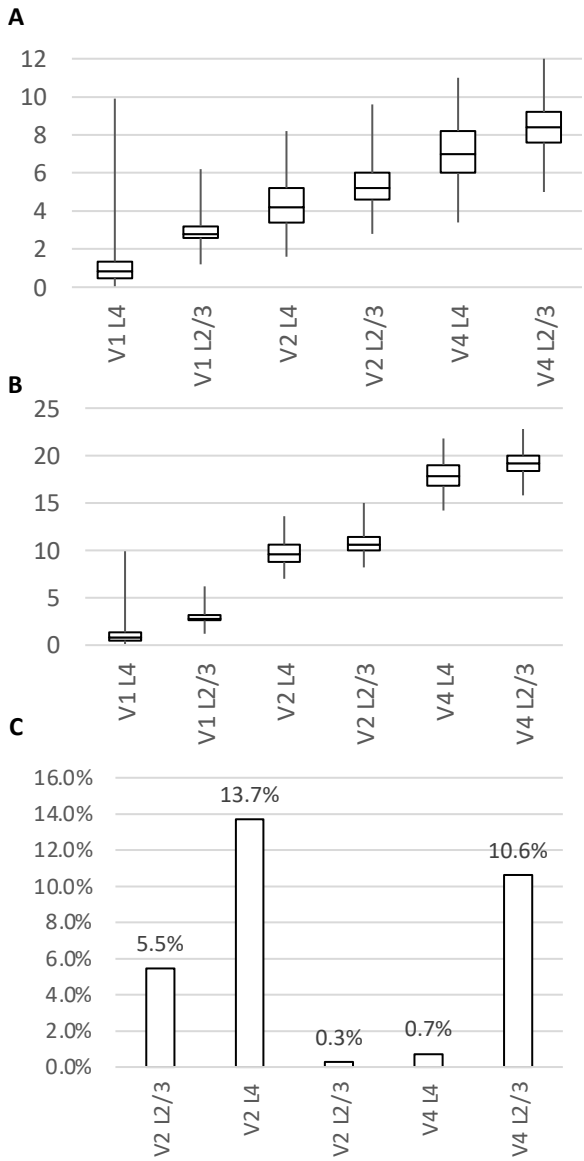

Fig. 9 Spike Propagation **A** Spike time distribution in the model with no delays between cortical layers and areas. **B** Spike time distribution assuming delays between cortical layers and between cortical areas. **C** Percentage of neurons spiking per stimuli, normalized by the percentage of stimuli in the neurons receptive field. No results are shown for real valued LGN model layer.

3 Discussion

3.1 Conclusions

The Organic Convolution (OC) model is a simplified but biophysically legitimate model for the feed forward propagation of spikes in cortex-like layers. The biophysical parameters for the time constants governing the SRM PSP equations (1) are within plausible ranges, so OC seems to meets the speed requirement in visual recognition [49] by showing how convergent processing from V1 to V4 can occur with a simple cascade of spikes.

The debatable part of the biological realism of the OC model is whether axons and dendrites can self-wire into the proposed functional micro-architecture patterns. The functional micro-architecture patterns seem biologically plausible in that they define local receptive fields and are quite simple. The evidence from V1 [28] and V2 [24] is that neurons do pre-wire into computationally useful patterns, but proof will require direct observation using connectomics [4] as well as elucidation of the mechanisms that guide such pattern formation [46]. The proposed functional micro-architectural patterns in this study are suggestions for the types of patterns that are sufficient to match V4 recording data but are not definitive.

This study does not include a computational-developmental model (like that in [37]) that can generate the proposed convergent micro-architectural patterns and this may be seen as making the results inconclusive. To rebut this point of view by analogy, consider that biologists often observe stripes of pigment on the bodies of animals or plants and reach conclusions about the function of those stripes for camouflage, signals, etc. But no one expects the biological mechanisms of pigment stripe formation to involve an energy minimizing interaction with the environment. In the same way, we can reason about the functions that simple patterns in neuronal structures may play without presently understanding the mechanism of their generation.

For practical reasons, AI studies of neural networks have focused on algorithms to learn the large number of neural network parameters from labeled data. If the ideas behind the OC model are correct, then evolution has found developmental mechanisms to create simple patterns that perform some of the same function without learning. The approach in this study is to use models with as few parameters as possible, so that a small amount of manual experimentation sufficed to find good enough parameterization to match the data under consideration. A recent proposal is that "goal driven" computational neuroscience studies should use DL and suggests that designed parameters is retrograde for neuroscience, just as it is for artificial intelligence [23, 52]. On the other hand, it can be argued that biophysicists seeking mechanistic explanations for cognitive processes should favor parsimony in the underlying model over convenience in finding useful parameterizations. If there exist models so simple that they can be parameterized with little effort, and those models can meaningfuly reproduce neurophysiological results, that is taken as evidence in favor of the "critique of pure learning" [53].

It is probably not realistic to assume that all neurons of a given type will be regularly shaped nor is it plausible that all functional micro-architecture connection patterns of similar units at different locations are exactly identical. The OC model in this study makes more sense when the micro-architectural patterns are viewed as a model for the "average" or "ideal" version of each unit type. Cortex may contains many units that follow a micro-architectural pattern in any area of the receptive field, but each individual unit must have its own irregular shape determined by its dendritic growth pattern around the other elements in the neuropil. When combined such units may functionally achieve a robust version of the average micro-architectural pattern. Investigating OC models with realistic numbers of units, and more variety in unit shape and micro-architectural connections is an attractive area for further research.

This study is a demonstration of a novel method for simulating waves of individual spikes in neural networks with convergent functional micro-architectures, but most of the issues raised in this discussion are not resolved definitively. But it is clear that convolutional neural networks and tensor technologies can be used to experiment with models of convergent functional micro-architectures in spiking neural networks at scale. Such models may be quite simple and still help explain neurophysiological data.

3.2 Ideas and Speculation

For understanding the nature of visual processing in V4, the OC model may suggest a synthesis of the points of view in different neurophysiological approaches: Studies suggesting V4 responds in a place invariant manner to shapes defined by closed paths [25, 40] may be reconciled with the fine shape and orientation tuning shown in [39] because the functional micro-architectures that are tuned to respond to medium and high curvature will also tend to prefer certain closed paths that approximate the receptive field boundary. Real neurons will have natural irregularities in their dendritic trees that may lead an individual neuron to respond most strongly to closed paths with various deformations as suggested by [25, 40]. Similarly, the preference of some V1 neurons for specific shapes other than simple oriented bars found in [54] is not in contradiction with the suggestion here that the average or aggregate V1 response may be simple and uniform across locations as presumed by the organic convolution hypothesis. Individual neurons must vary due to the random aspects of their existence in the neuropil, but to ascribe functional significance to all of the variation seen in cortical cells may be "overfitting" the conceptual model of V1. Regardless, the recent results in [19] seem in agreement with [39]

and the OC model in suggesting that the function of V4 may be to represent local variations in orientation and curvature. The best way to resolve these questions would be to characterize both the local orientation curvature tuning and the optimal composite paths for individual neurons in a single study, and see if a single model can reproduce both behaviors.

The coding used in the OC model is a population place code: each unit responds to the presence of a particular pattern of inputs at a particular location in the visual field with broad tuning that overlaps with neighbors, and there are functionally identical units tiling the receptive field. Given the place code in the OC model, the interpretation of the model is the same whether considered as a binary network or a spiking network. If the OC model is an accurate representation of brain activity, it would suggest that the venerable McCullough-Pitts neuron model [36] may go a long way in explaining important classes of brain activity. Also, the OC model is compatible with earliest arrival spike time coding [49]: If the inputs to area V1 in the OC model are ordered so that the earliest arrivals are from the most intense stimuli, it may enhance the place code defined by the functional micro-architectures.

The experiments in this study model a single volley of spikes as the basic unit of computation, but such a system may display the characteristics of a rate code as well: With the assumption of repeated waves of activity, the place code neurons best matching a stimuli would fire with the fewest failures and at a higher overall rate. But if the hypotheses of OC are correct then the "rate code" is an epiphenomenon and not really a code at all - the rate is the result of repeated feedforward waves of spiking convolutional place code computations with noise.

An important related question is whether and how an OC model like the one in this study may be useful for object recognition. The convolution filters used in the OC model in this study are not exactly like those discovered in trained machine learning models, but they may still serve a similar purpose. Initial experiments suggest a network like the OC model may be used to perform machine learning tasks by feeding the final convolution output into a fully connected network that is trained with gradient descent [32]. If OC is found to be a good representation of biology, and it can also function in object recognition applications, it may serve as a bridge for understanding the relationship between the current generation of artificial neural networks and the brain.

4 Materials and Methods

4.1 Spiking Neuron Model

The spiking neuron model used for the Spike Time Convolution (STC) model is the simplified Spike Response Model (SRM) described in [17]. The SRM model is an abstraction of the integrate and fire neuronal biophysics but it captures the key properties of spike integration and timing for single action potentials. Each model neuron has a potential level given by a the sum of the excitatory and inhibitory post synaptic potentials (PSP). The time course of the PSP's are described by a difference of decaying exponentials. For any given synapse (indexed by i) and given an input spike arrival time s the influence of the PSP is given by the kernel:

$$\epsilon_i(t \mid s) = \frac{w_i}{Z}[e^{(\frac{t-s}{\tau_{fall}})} - e^{(\frac{t-s}{\tau_{rise}})}]\Theta(t-s) \qquad (1)$$

In equation 1 w_i represents the synaptic input sign and weight. Weights are positive for excitatory and negative for inhibitory, depending on the type of the afferent input and the weight is determined by the micro-architectural connection pattern, as described below; Z is a normalization constant chosen so that the maximum value of the PSP is w_i; $\Theta(x)$ denotes the Heaviside step function. The time constants of $\tau_{rise} = 1$ms and $\tau_{fall} = 10$ms where used for the simulations.[3] The simulations of each wave of cortical activity lasted 10 ms, with a discretization step of 0.2 ms in the simulated PSP dynamics.

Given the PSP kernel function in equation 1 and a set of input spike times on a set S of afferent inputs indexed by i the neuron spikes at the time

[3] PSP time constants at synapses are generally faster, around 0.5 ms τ_{rise} and dendritic trees have 5-10 ms passive τ_{fall}. But it is also observed that the resulting *somatic* depolarization is generally slower, due to active conductances and cable effects in the dendritic tree : depolarization rise times at the soma have time constants of 1-2 ms or more and decay time constants greater than ten milliseconds. [30]

Table 1 Network Summary

Model Layer	Model For	Layer Size	Neuron Patterns	Orientations	Inhibitory Units?	Total Channels	Equivalent Neurons
1	V1 L4	112×112	1	1	Y	2	25,088
2	V1 L2/3	112×112	1	8	N	8	100,352
3	V2 L4	56×56	1	8	Y	16	48,400
4	V2 L2/3	56×56	6	16	N	88	202,752
5	V4 L4	28×28	1	16	Y	176	137,984
6	V4 L2/3	28×28	3	8	N	8	10,976
						Total	525,552

$$\arg\min_{t} \sum_{i \in S} \epsilon_i(t \mid s_i) >= 1.0 \tag{2}$$

where the the threshold in equation 2 is set to 1.0 for model interpretability: with this definition, all synaptic weights w in the convolution kernels (Figures B4, B5, 4, 6 and 8) are expressed in terms of the positive excitatory threshold. The constant Z in equation 1 is determined by first calculating the time that PSP achieves its maximum, as described in the Appendix.

4.2 Network Model

The model network consisted of six layers, representing L4 and L2/3 in cortical areas V1, V2, and V4. As shown in Table 1, each layer represents a number of model neurons that is the product of the layer area and the number of different types of units at each position. The layer area starts as the number of pixels in the input layer, and at deeper layers the dimension shrinks due to pooling operations. The number of units at each position in the model is given by the number of functional micro-architectural patterns and the number of orientations at which the pattern is instantiated. Each unit type is modeled as a channel in the tensor simulation as described below. Also, for model layers representing cortical L4 the number of units is doubled because of including both excitatory and inhibitory neurons. The total number of equivalent neurons represented by the model is approximately 525,000.

As explained in the Introduction, the model focuses on a simple feedforward wave of spikes and does not include other details of the cortical circuitry, such as lateral connections within cortical layers [27] or cortical micro-circuits to L5/6 [14]. The model also does not include inter-cortical circuits giving feedback from V4 to V2 and V1 and from V2 to V1, nor the interaction between feedback and lateral circuits [33]. However, due to the short duration of stimuli presentation (16 ms) in [39] it is expected that the other inter-cortical circuits will not contribute significantly to the response for these stimuli.

Conceptually, the model assumes that the feedforward dynamics in the ventral pathway is driven by episodic arrival of inputs in V1 from the lateral geniculate nucleus (LGN), synchronized with precision on the order of a few milliseconds, which has been observed in multiple studies, as reviewed in [47]. The model as implemented is memoryless in that it only models single episodes of activity.

One gap in the realism of the model is that delays of all kinds are ignored, as was described in the Results section 2.4. Failure in spike propagation is also ignored, and there are only two sources of noise in the model: Random variation in the input intensity and the random distribution of the arrival times. The random arrival time distribution is set to the level of synchronization seen in V1 [18], so the random intensity variation must be viewed as a lumped source of randomness that includes other un-modeled noise sources.

Memoryless simulation of a forward wave of single spikes was a technical necessity so that a simulation fit entirely within the available GPU memory. The memory required for the simulation (500K neurons for 10 ms at 0.2 ms time step) was just under capacity for the 16GB of available GPU memory on the Tesla V100 GPU used for the experiments.

Listing 1 Binary Convolution

```
1  import numpy as np
2  import tensorflow as tf
3
4  class BinaryLayer(tf.keras.layers.Layer):
5      def __init__(self, conv_weights : np.array,
6                           stride : int):
7          self.conv_weights= tf.constant(conv_weights)
8          self.stride = stride
9
10     def call(self, inputs : tf.Tensor) -> tf.Tensor:
11         activation = tf.nn.conv2d(input=inputs,
12                                   filters=self.conv_weights,
13                                   padding='SAME',
14                                   strides=[1, self.stride, self.stride, 1])
15         spiked = tf.greater_equal(activation, 1.0)
16         output = tf.cast(spiked, tf.float32)
17         return output
18
19
20 def binary_simulation(binary_layers : list[BinaryLayer],
21                         stimuli : tf.data.Dataset):
22     model = tf.keras.Sequential()
23     for layer in binary_layers:
24         model.add(layer)
25     model.compile()
26     output = model.predict(stimuli)
```

4.3 Tensor Simulations

4.3.1 Convolution With a Binary Neuron Model

As an introduction to the Spike Time Convolution (STC) model, listing 1 is the Python code for a convolutional neural network layer in which the model neuron is a simple binary threshold unit with no time dependence [36]. Listing 1 uses the `tensorflow` and `keras` packages: The class `BinaryLayer` descends from `tf.keras.layers.Layer`. `BinaryLayer` has convolution filter weights (and a stride) defined at initialization. The `call` function in Listing 1 defines the transformation from input tensors to output tensors for the layer. The convolution itself is performed with the `tensorflow` model function `nn.conv2d`. To perform a simulation, multiple `BinaryLayer`'s are added to a `tf.keras.Sequential` model object, as shown on lines 20-26 of Listing 1. After adding the layers, the model is compiled and outputs can be generated from a `tf.data.Dataset` holding the stimuli.

The code shown in Listings 1 is a simplification of that used for the simulations shown in the Results section in one important respect: Listing 1 perform the convolution using a single convolution filter size (lines 11-14). It is conventional to use a single filter size in DCNN applications for machine learning. For the Organic Convolution experiments described here, the filter sizes are determined by the receptive fields observed in recordings and they vary depending on the unit, particularly in the V1 and V4 models.

4.3.2 Spike Time Convolution With the Spike Response Model

Python code to model one layer of a Spike Time Convolutional neural network implementing the Spike Response Model is shown in Listing 2. The layer class `SpikingLayer` inherits from the `BinaryLayer` in Listing 1 and shares the same code for the convolution operation. The `SpikingLayer` has additional parameters in the

Listing 2 Spike Time Convolution with the Spike Response Model

```
import numpy as np
import tensorflow as tf
from binary_convolution import BinaryLayer

class SpikingLayer(BinaryLayer):
    current_time = None # Class variable for simulation time

    def __init__(self, psp_trise : float, psp_tfall : float,
                 inhibitory_inputs : bool, **kwargs):
        BinaryLayer.__init__(self, **kwargs)
        # Save scalar variables and calculate scale (not shown)

    # input_shape : batch size, y dimension, x dimension, number of channels
    def build(self, input_shape : tuple[int, int, int, int]):
        # Tensor of -1.0/time constants and scale for use in the PSP calculation
        self.psp_fall_z = tf.scalar_mul(-1.0/self.psp_tfall, tf.ones(input_shape))
        self.psp_rise_z = tf.scalar_mul(-1.0/self.psp_trise, tf.ones(input_shape))
        self.psp_scale = tf.scalar_mul(self.psp_scale, tf.ones(input_shape))
        # Initialize a tensor holding the input sign (excitatory or inhibitory)
        if not self.inhibitory_inputs:
            self.input_signs = tf.ones(input_shape)
        else:
            excite_signs = tf.ones(shape=(input_shape[0],input_shape[1],
                                          input_shape[2], input_shape[3]/2))
            inhib_signs = tf.scalar_mul(-1.0, excite_signs)
            self.input_signs = tf.concat([excite_signs, inhib_signs], axis=3)
        # Initialize a tensor for output spike times; 0 indicates no spike
        output_shape = (input_shape[0], input_shape[1]/self.stride,
                        input_shape[2]/self.stride, self.conv_weights.shape[3])
        self.spike_times = tf.zeros(shape=output_shape)

    def call(self, inputs : tf.Tensor) -> tf.Tensor:
        # Calculate time delta since spike arrival
        delta_t = tf.subtract(SpikingLayer.current_time, inputs)
        # if past the spike time, psp=scale*exp(-dt/tfall)-exp(-dt/trise)
        psp_unsigned = tf.where(tf.logical_and(tf.greater(delta_t, 0.0),
                                               tf.greater(inputs, 0.0))
                                tf.multiply(self.psp_scale, tf.subtract(
                                    tf.exp(tf.multiply(self.psp_fall_z, delta_t)),
                                    tf.exp(tf.multiply(self.psp_rise_z, delta_t)))),
                                tf.zeros(input_shape))
        # Apply the signs of the input layer to the PSP's
        psp_signed = tf.multiply(psp_unsigned,self.input_signs)
        # Call to the base class binary convolution using the PSP's as inputs
        conv_output = super(SpikingLayer,self).call(psp_signed)
        # Find newly spiking neurons
        new_spike = tf.logical_and(tf.equal(conv_output, 1.0),
                                   tf.equal(self.spike_times, 0.0))
        # Add new spike times to the outputs
        current_t = tf.scalar_mul(SpikingLayer.current_time,
            tf.ones_like(self.spike_times))
        self.spike_times = tf.where(new_spike, current_t, self.spike_times)
        return self.spike_times
```

initializer for the PSP time constants (τ_{rise} and τ_{fall} in Equation 1) and a flag indicating whether the inputs include inhibitory units. Note that the `SpikingLayer` has a class variable that is used to hold a reference to the current simulation time (Listing 2 line 6). The full details of the `SpikingLayer` initialization are omitted from Listing 2 for brevity; these are copying the parameters to member variables, and calculating the PSP scaling constant given by Equations B1 and B2.

Neural network layers in the `keras` framework have the option to define a `build` function that performs additional setup after initialization, given the input shape. (The input shape is unknown at initialization.) The `SpikingLayer` defines a `build` function to perform the following setup tasks:

1. Define tensor versions of the PSP time constants having the same shape as the input (lines 16-18)
2. For layers with both excitatory and inhibitory input, create a tensor of signs to apply to the PSP summation so that the inhibitory units reduce activation (lines 19-26).
3. Create a tensor variable to hold the spike times for every unit in the layer (lines 28-30).

The `call` function of the `SpikingLayer` takes an input tensor that represents the spike times of the units in the previous layer. The spike time tensor stores zero to indicate no spike. As will be explained below, the simulation starts at a time one discretization step after 0, and the class variable `current_time` is set before the `call` method is invoked (Listing 3).

In `call` for the `SpikingLayer`, the main steps are as follows:

1. Calculate the time since the spike of each unit that has fired (line 34); negative values result for inputs that have not spiked, but these are screened out at the next step.
2. Calculate the current PSP activations with the SRM model (line 36). This includes logic equivalent to the step function in equation 1.
3. For those inputs that are inhibitory, apply the required sign to the PSP activation (line 42)
4. Invoke the base class `call` function which performs the convolution with the current PSP values as the inputs (line 44). This step converts the input dimension to the output dimension, and emits a 1 for every neuron that has fired.
5. Find what inputs have newly fired by comparing the current convolution outputs to the previous spike times and store it in another tensor (line 46).
6. Store the current time in the member variable `spike_times` for all of the units that have newly spiked (lines 49-52)
7. Return the newly updated member variable of spike times as the new output vector (line 53)

After defining the layers, the tensor graph for a complete spiking simulation must be created before passing in inputs and executing the graph, which is shown in Listing 3. The first step in the simulation graph is to set the input times in the first `SpikingLayer` (the input layer) to random times drawn from a gamma distribution. The main tensor graph is created in a double loop over time, and layer (lines 17-20.) The first step in the simulation loop for a point in time is to set the `current_time` class variable to a tensor constant for use in all the layer calculations (Listing 3 line 19.) Then the layers are called in succession with the output of each layer as the input to the next (lines 20-23); note that for `keras.Layer` the `call` function is overloaded with the `()` operator and does not need to be named explicitly (line 23).

To perform a simulation, a base class `keras.Model` is created from the first input and the final output; the standard `tf.keras.Sequential` model cannot be used for this specialized use case. After creation, the `Model` can be used to generate outputs as usual by first compiling the model, and then generating outputs with the `Model.predict` function.

Note that the `BinaryLayer` shown in listing 1 is not just an illustrative abstraction: During research and development, functional micro-architectural patterns were first tested with a network of binary neurons just like in listing 1. The behavior of the binary model is a non-random version of the behavior of the spiking model, and the speed of the binary simulations makes it useful for prototyping. After prototyping with a binary model, the weights were increased so the units would spike under noisy conditions.

Listing 3 Spike Time Convolution Simulation

```
import tensorflow as tf
from spiking_convolution import SpikingLayer


def spiking_simulation(spiking_layers : list[SpikingLayer],
                       stimuli : tf.data.Dataset,
                       input_shape : tuple[int,int,int,int]
                       dt : float, tmax : float):
    # Set input layer spike times
    spiking_layers[0].spike_times = tf.random.gamma(shape=input_shape,
                                                    alpha=gamma_shape,
                                                    beta=1.0 / gamma_scale)
    # Define first input and output reference
    first_input = tf.keras.Input(shape=input_shape[1:])
    output = None
    # Loop incrementing current time and calling layers in sequence
    for tidx, t in enumerate(np.arange(dt, tmax, dt)):
        # Set current time static variable to the tensor constant
        SpikingLayer.current_time = tf.constant(t)
        for layer_num in range(0, self.numLayers):
            input_for_layer = output if layer_num > 0 else first_input
            # Call the layer using the `()` operator
            output = spiking_layers[layer_num](input_for_layer)
    # Define the model from first input to final output
    model = tf.keras.Model(inputs=first_input, outputs=output)
    # Compile the model
    model.compile()
    # Generate output
    output = model.predict(stimuli)
```

Acknowledgments. Thanks to John Reynolds and Anirvan Nandy for allowing reproduction of their figures and for helpful discussions and suggestions.

Competing Interests. Some machine learning applications of the techniques described here may be covered by filed patent applications U.S. Pat. App. Ser. No. 16/125,818 and provisional application 63337083. The code will always be freely available for academic research.

Code Availability. Upon publication the source code from this study will be available at http://github.com/carl24k; during the review process the source code is available by request.

Appendix A Network Response Details

Figure A1, A2 and A3 illustrates the response of the entire network to low, medium and high curvature stimuli in the central position in the visual field. The sub-plots show the responses of the L4 pool units corresponding to each type of functional micro-architecture in the model. The plots for V1 (row B) and V2 (rows E through O) are zoomed by 2.3X and 2X respectively. The following annotation key serves for all three figures:

- **A** Heatmap of average input intensity.
- **B** Pictogram representation of the preferred stimulus of the V1 units. See Figure B4 for details.

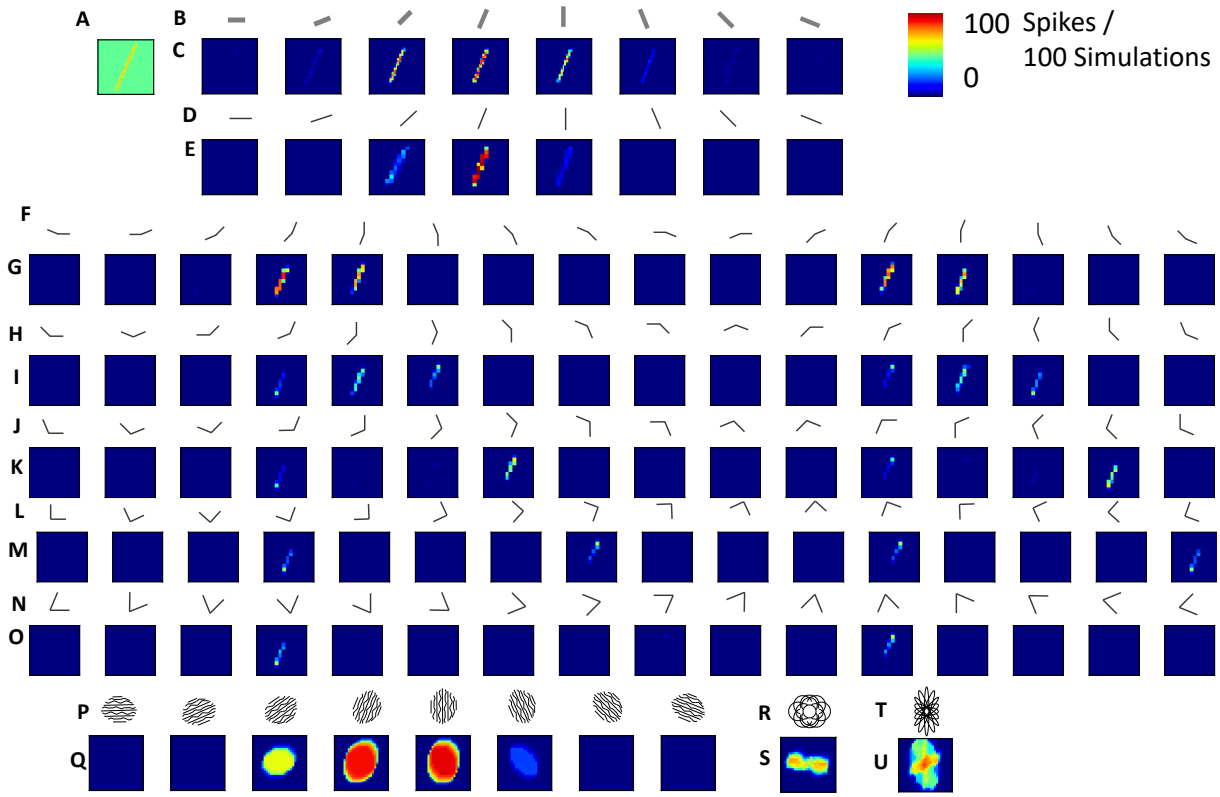

Fig. A1 Illustration of Network Response to Low Curvature Stimuli. See text for annotation key.

- **C** Heatmap of firing rate of the V2 L4 bar selective unit response measured in spikes per 100 simulations.
- **D** Pictogram for the preferred stimulus of the ultralong units. See Figure B5 for details.
- **E** Heatmap of the V4 L4 ultralong selective pool unit responses.
- **F** Pictograms for the preferred stimulus of complex-shaped units having 22.5° angle between sub-units.
- **G** Heatmap of the V4 L4 22.5° complex-shaped selective pool unit responses.
- **H** Pictograms for the preferred stimulus of complex-shaped units having 45° angle between sub-units.
- **I** Heatmap of the V4 L4 45° complex-shaped selective pool unit responses.
- **J** Pictograms for the preferred stimulus of complex-shaped units having 67.5° angle between sub-units.
- **K** Heatmap of the V4 L4 67.5° complex-shaped selective pool unit responses.
- **L** Pictograms for the preferred stimulus of complex-shaped units having 90° angle between sub-units.
- **M** Heatmap of the V4 L4 90° complex-shaped selective pool unit responses.
- **N** Pictograms for the preferred stimulus of complex-shaped units having 112.5° angle between sub-units.
- **O** Heatmap of the V4 L4 112.5° complex-shaped selective pool unit responses.
- **P** Pictogram representation of the preferred stimulus of the V4 low curvature selective unit with 8 different preferred orientations. These units are defined by the functional micro-architectures and convolution kernels illustrated in Figure 4.
- **Q** Heatmap of the V4 low curvature selective unit responses.
- **R** Pictogram representation of the preferred stimulus of the V4 medium curvature selective unit. These units are defined by the functional micro-architectures and convolution kernels illustrated in Figured 6.

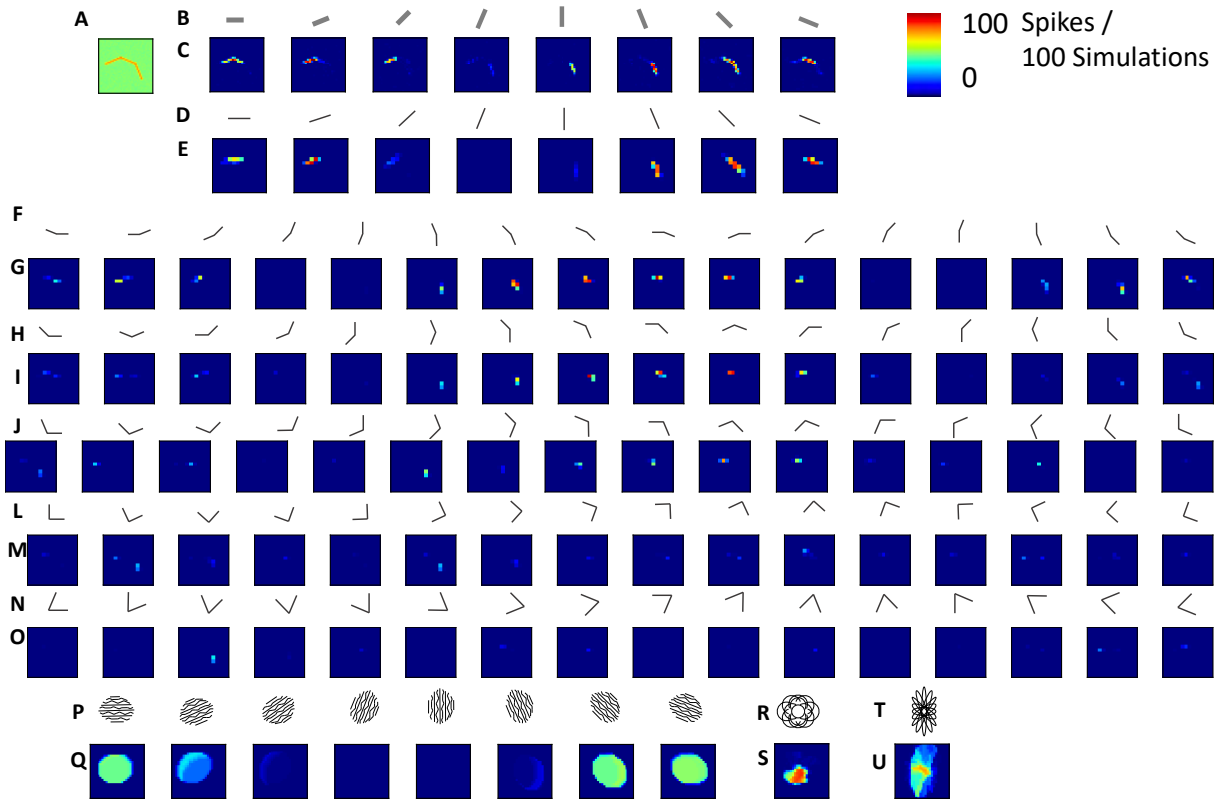

Fig. A2 Illustration of Network Response to Medium Curvature Stimuli. See text for annotation key.

- **S** Heatmap of the V4 medium curvature selective unit responses.
- **T** Pictogram representation of the preferred stimulus of the V4 high curvature selective units. These units are defined by the functional micro-architectures and convolution kernels illustrated in Figured 8.
- **U** Heatmap of the V4 high curvature selective unit responses.

Appendix B Supplementary Methnods

B.1 Convolution Models of Convergent Functional Micro-architectural Patterns

Convergent micro-architectural patterns are used to determine the convolution filter weights in the model. The patterns are implemented as rules giving the input preference for dendritic locations as a function of the position relative to the soma, using trigonometric functions. It is assumed that individual neurons have a baseline orientation property referring to a direction in the topographic map. The dendrite's preferences are determined relative to the neuron's baseline, which facilitates instantiating multiple versions of a pattern with different orientation preferences.

Given a rule defining the pattern, the following process was used to creates the convolution filters for the model:

1. Pick an ellipse defining the receptive field at the baseline orientation; points within the rectangular convolution filter dimensions but outside of the ellipse boundary are set to zero.
2. Dendrite paths are determined with a 2-dimensional[4] ball and stick model, traveling from the center of the receptive field to the edge of the ellipse.
3. For each rotated version of the pattern, the ellipse and all of the dendritic points are rotated to the new orientation. The smallest rectangle bounding the rotated ellipse defines the convolution filters for that channel.

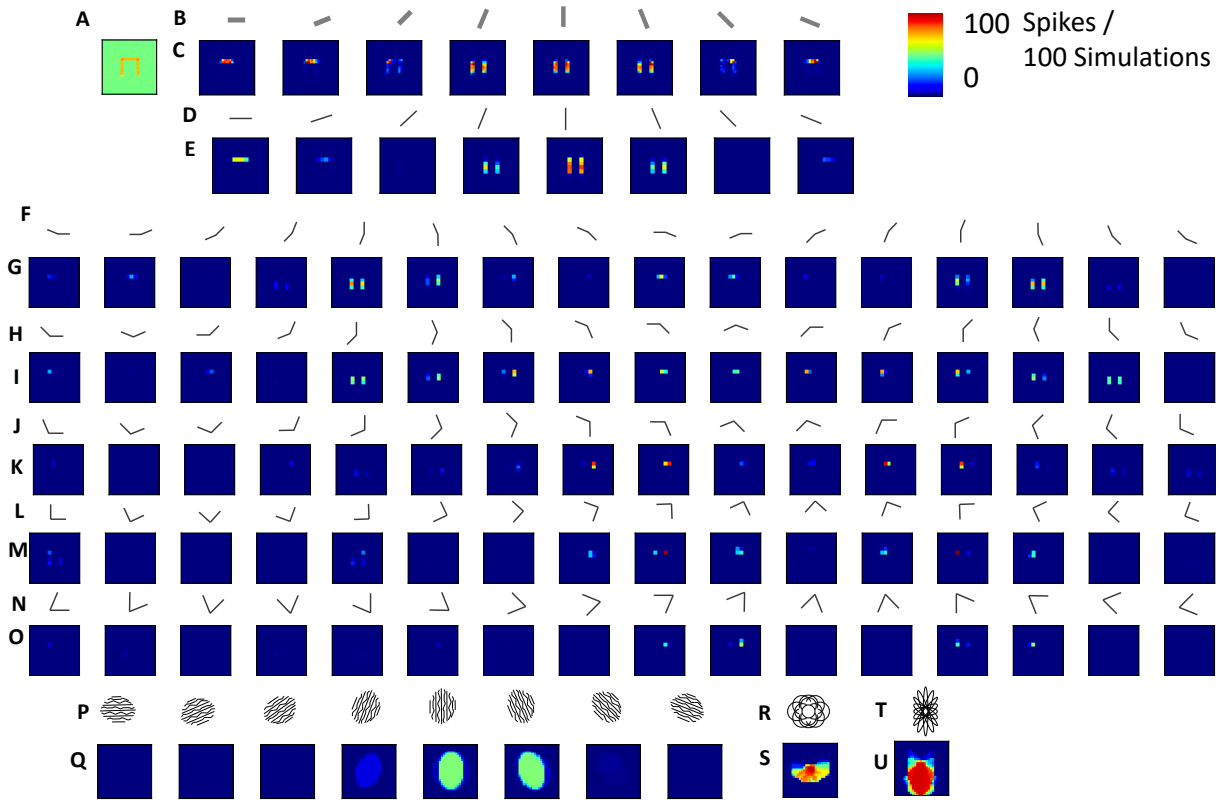

Fig. A3 Illustration of Network Response to High Curvature Stimuli. See text for annotation key.

Table B1 Functional Micro-Architecture Summary

Type	Receptive Field	# Dendrite	Excit-ation	Inhib-ition	Pattern Description
V1 L2/3	6×9	32	2.4	1.2	Gabor-like
V2 L4	2×2	8	4	0	Pool (Or)
V2 L2/3 Ultralong	9×9	32	3.2	7.7	Long Gabor-like
V2 L2/3 Complex	9×9	32	2.4	5.3	Gabor-like sub-units
V4 UL/C Pool	2×2	8	4	0	Pool (Or)
V4 Low Curve	15×20	96	164	52	Constant, low angle
V4 Medium Curve	16×22	96	507	185	Convex, medium angle
V4 High Curve	16×32	96	1769	868	Convex, high angle

4. The functional micro-architectural rule is applied at sample points along the dendrites to determine the weights for every point in every available input channel.

The size and strength of the input connections for the functional micro-architectural patterns are summarized in Table B1.

B.2 V1 Model

The functional micro-architectures for V1 L2/3 units can be extremely simple (no pun intended) and still reproduce the complex shape tuning of V4 recordings: The functional micro-architecture

[4]For all functional micro-architectures discussed in this study the arrangement of the inputs in the direction of lower and higher cortical layers is not relevant to the operation of the micro-architecture. Only the layout of afferent inputs in the plane of the topographic map, or parallel to the layers of cortex, is relevant to the functioning described and the model only represents input layout in that plane.

is represented by a 6×9 convolution filter with a 3 pixel wide band of excitation in the center surrounded by inhibition, as illustrated in figure B4. Figure B4**A** also illustrates how this pattern could be created by having a few dendrites lying close to an axis in the topographic space make (primarily) excitatory connections to afferent inputs from L4, while dendrites away from the axis are (primarily) inhibitory. The model includes 8 different orientations of the same pattern, 22.5° degrees apart in the orientation of the optimal stimulus.

To reproduce the curvature and orientation tuning found in V4, the V1 L2/3 units were excitable enough to have broad tuning. The ratio of excitatory to inhibitory weight was 60% to 40%. This is illustrated in Figure B4, which shows weights in units of the neuron threshold: Defining the neuron's threshold as 1, the total excitatory weight in 6×9 convolution filter was 3.3 and the total inhibitory weight was 2.1. The excitatory weight was high enough that if only 40% of the inputs were active, it was enough to cause the unit to spike.

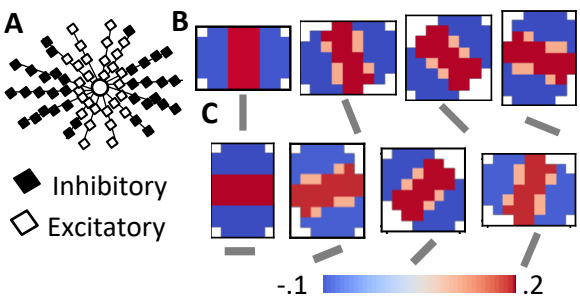

Fig. B4 Functional micro-architecture for V1 simple cell. **A** Sketch of functional architecture pattern for bar selective neurons with the bar oriented in vertical direction. Excitatory inputs are centered near the vertical axis, separating the inhibitory inputs. **B** 6×9 convolution filter resulting from the oriented bar detecting functional micro-architecture. The weights are shown in units of the neuron spiking threshold. The model includes 8 versions of the filter at orientations 22.5° apart. **C** Illustration of the preferred stimuli.

B.3 V2 Model

The V2 model is inspired by the findings in [34] and the V2 L2/3 units in the OC model implements both ultralong and complex-shaped selectivities described therein. The OC model further hypothesizes one underlying functional micro-architecture pattern may lead to both selectivities: It is assumed that these V2 units have a small number of dendrites receiving feedforward excitation, those dendrite extend radially outward in the topographically organized cortical plane, and prefer to connect to excitatory V1 afferents that have the same preferred stimulus orientation as the direction of the dendrite itself. If a unit has two such dendrites that extend in a direction from the soma that is substantially opposite to each other, they will form an ultralong receptive field; this arrangement is illustrated in Figure B5A. Inhibitory inputs are connected for orientations other than the ones preferred by the excitatory dendrites, and there are other dendrites whose role in the feedforward circuit is predominantly inhibitory.

Using the same approach, if a unit has two dendrites with the same connection preference, but in the case where those dendrites have some angle to each other the result will be a complex-shaped receptive field; this arrangement is illustrated for two dendrites at 90° to each other in Figure B5F. These units prefer stimuli that resembles two lines meeting in a corner, and the angle between the two excitatory dendrites sets the angle preference for the unit. The 9×9 convolution filters resulting from some instantiations of the V2 L2/3 functional micro-architecture are also illustrated in Figures B5: Excitatory connections are concentrated on a few orientations in simple straight or angular arrangements. Inhibition is diffuse and spread over the receptive field.

The model include 8 different orientation preferences for ultralong units, oriented 22.5° apart. The model also includes a range of angles defining the separation between the sub-units of the complex-shaped units, from 22.5° to 112.5° in 22.5° increments for a total of 5 complex-shaped sub-unit angles. Because the complex-shaped units are not symmetric to 180° rotations, 16 different orientations (22.5° apart) of each sub-unit separating angle are defined to respond to the full range of possible stimuli.

The best fit for V4 shape selectivity required the V2 neurons to have strong excitation, but in these cases the total inhibition was greater: The total excitatory weight in the 9×9 receptive fields for V2 ultralong units was 3.2× the threshold, while the total inhibition was 7.7× the threshold. For the complex-shaped units the excitation was

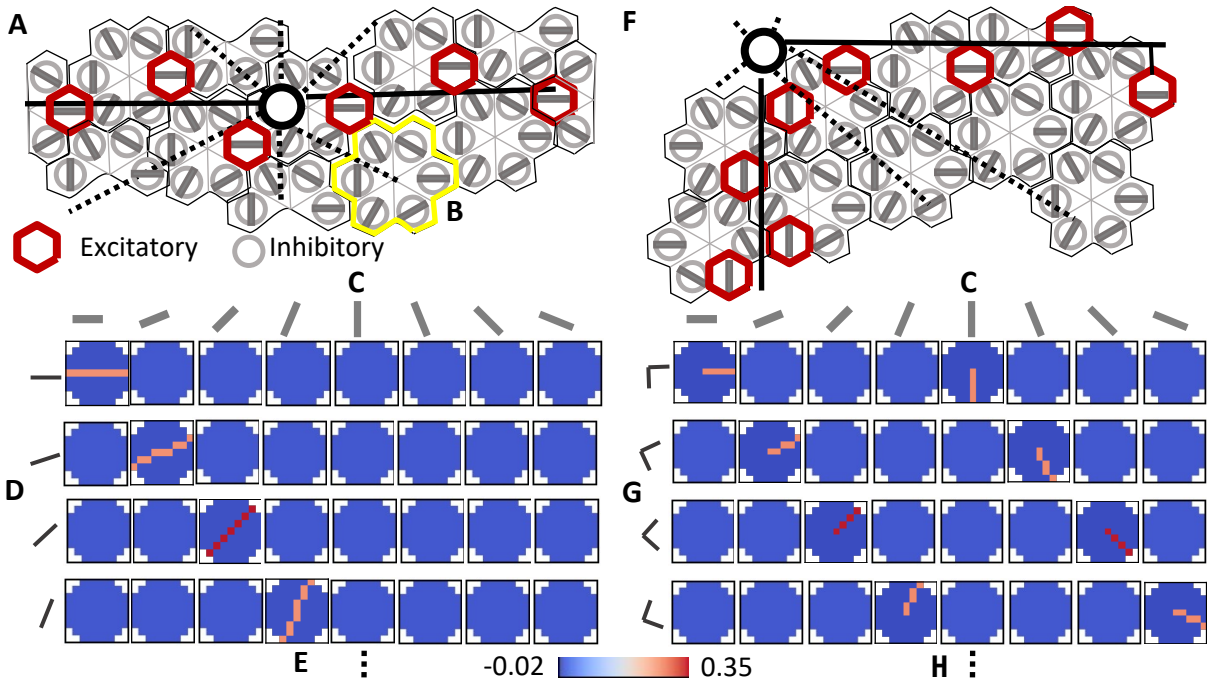

Fig. B5 Functional micro-architecture for V2 L2/3 units. **A** Illustration of a functional micro-architecture for an ultralong type V2 unit: Two dendrites having approximately opposite direction in the topographically organized visual field receive excitatory inputs from V1 afferents with orientation preference matching the dendrites' directions. Other dendrites form inhibitory connections to afferent inputs with all orientation preferences. **B** Pictorial representation of the afferent inputs from a pinwheel of orientation selective V1 units; 6 of 8 orientation preferences are shown. **C** Orientation preference of the afferent input channels from V1, as shown in Figure B4. **D** Illustration of the optimal stimulus for the ultralong units. **E** 9×9 convolution kernels for 4 out of the 8 ultralong units having a orientation preferences ranging from horizontal to 67.5° **F** Sketch of the functional architecture pattern for a complex-shaped V2 unit: The same as for the ultralong unit, but the dendrites are at approximately 90° to each other in the plane of the topographic map. **G** Illustration of the preferred stimulus of four complex-shaped units. **H** 9×9 convolution kernels for a complex V2 unit having a sub-unit angle of 90°; 4 out of 16 of the orientations in the model are shown.

2.3 times the threshold, and the total inhibitory weight is 5.2.

B.4 Layer 4 Pooling Models

The first layer of the model serves as a real valued input layer to the neural network and is used as a lumped model for the eye/retina, LGN, V1 L4α and V1 L4B.[5] [51] An unusual property of the input layer in this model in comparison to a standard DCNN is that it contains a redundant set of excitatory and inhibitory units - this is modeled after V1 L4B which makes both excitatory and inhibitory projections to V1 L2/3. The excitatory and inhibitory units are redundant in the input intensity they receive from the stimuli, but have independent random spike arrival times. With this redundant structure, the V1 L2/3 layer receives both excitatory and inhibitory input from every position in the input while still obeying Dale's Law. In the convolutional model, this translates to two separate channels. The model ignores that there should be different time constants for the excitatory and inhibitory stellate cells in L4 [21], leaving that as a subject for future study.

As in V1, V2 L4 plays the role of receiving excitatory input and replicating it so that it includes both excitatory and inhibitory afferents that obey Dale's Law.[5] The L4 model for V2 also performs a pooling function: The units have 2x2 receptive fields and are spaced 2 positions apart. This is known as a stride of 2 in the terminology of DCNN, and results in the dimension of the output being reduced by one half along each axis: The input to model V2 L4 is 112×112, and the output is 56×56.

Each of the 8 channels of V1 L2/3 functional micro-architectures has a corresponding unit type (channel) in V2 L4: each type accepts inputs from

one channel of the V1 L2/3 outputs, a total of 4 inputs for each V2 L4 unit. The inputs are weighted so that any one of the 4 inputs from V1 to an area V2 L4 unit will trigger an output spike. The result is a spiking version of the maximum pooling operation in a DCNN. As in the area V1 L4 model, there are excitatory and inhibitory channels representing separate populations of excitatory and inhibitory neurons.

The model for V4 includes a model of L4 that is identical to the V2 L4 model, adapted to the V2 afferents: V4 L4 units take afferents from a 2×2 pixel receptive field, and are staggered 2 pixels apart and the dimension of the visual maps are reduced from 56×56 to 28×28. Each unit in the V4 L4 model accepts afferent inputs from a single type (channel) of the V2 functional micro-architecture in terms of both sub-unit separation angle and the unit's overall orientation. The L4 units have sufficient excitation so that they are guaranteed to spike if any of their four inputs from V2 units spike.

B.5 Input Stimuli

Input images were created following the approach described in [39] and adapted to the neural network modeling framework. Line segments 12 pixels long were drawn in black on a white 112×112 pixel background. Each model stimuli consisted of 3 line segments joined at the end and forming 5 different degrees of curvature using 5 different angles: 0° for straight lines, 22.5° for low curvature, 45° for moderate curvature, 77.5° for high curvature, and 90° for "C" shaped. Each stimuli was used in 16 rotated versions separated by angles of 22.5°. The stimuli were placed in the image on a 3×3 grid, 38 pixels apart, beginning at pixel (18,18). This created a total of $5 \times 16 \times 9 = 720$ stimuli. Note that as in [39], the straight line stimuli are redundant when rotated by 180° but these duplicates were left in the stimuli set for simplicity.

After plotting the lines and rendering the image as a bitmap, gaussian noise was added during each simulation: Each pixel was adjusted by zero mean gaussian noise with a standard deviation set to 40% of the maximum image intensity. The resulting image was then re-normalized. Examples of the resulting input images from a single simulation are shown in Figure B6. The fairly high level of background noise was not used in the stimuli in [39], but is included in the model to help match the broad background firing seen in the recordings. As noted in the Discussion, other sources of noise are ignored for computational expediency.

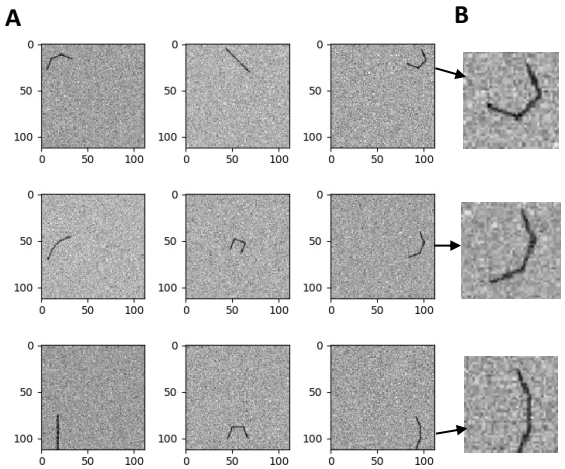

Fig. B6 **A** Sample input images in the 9 positions. Examples of all 5 curvatures and 9 out of 16 rotations are shown. **B** Zoom in on the right three curves.

B.6 Hardware and Runtime

Assuming atemporal binary inputs and outputs as in Listing 1, the tensor graph for 6 layers (3 cortical areas, 2 layers per area) has around 150 operations. This is more than suggested by Listing 1, as the actual model code must handle convolution filters of varied sizes and the input layer. The spike time simulation using the PSP equations given by Listing 2 and Listing 3 was run for 10 ms discretized at 0.2 ms intervals; the resulting tensor simulation is an approximately 50,000-operation graph representing the layers at each time, plus additional operations for the PSP calculations and logic relating to the temporal simulation.

These large tensor graphs run on standard hardware: a simulation of a 525,000-neuron network for 10 ms takes 45 minutes to run on a MacBook Pro without a GPU and 3.5 minutes on an AWS p3.2xlarge GPU instance having a Tesla V100 GPU with 16 GB of RAM. As previously

[5] Assuming single feed-forward layer in L4 is a simplification in the model of L4 for each cortical area: In animals there are sub-layers of V2 L4 that receives excitatory input from V1 and then make excitatory connections to other V2 L4 sub-layers, and those other V2 L4 sub-layers make both excitatory and inhibitory connections to V2 L2/3 [51].

mentioned, the GPU memory is the key constraining factor when it comes to running longer or more detailed simulations. The simulations shown in the Results section run in just under the memory available.

B.7 PSP Maximum

The constant Z in equation 1 is determined by noting that the difference of exponentials in equation 1 takes its maximum at a time after the input arrival given by:

$$t_{max} = \tau_r \tau_f \frac{\ln(\tau_r/\tau_f)}{\tau_r - \tau_f} \tag{B1}$$

and the constant Z is given by

$$\frac{1}{Z} = [e^{(\frac{t_{max}}{\tau_{fall}})} - e^{(\frac{t_{max}}{\tau_{rise}})}] \tag{B2}$$

References

[1] Ackman, J. B., Burbridge, T. J., and Crair, M. C. (2012). Retinal waves coordinate patterned activity throughout the developing visual system. *Nature*, 490(7419):219.

[2] Agarap, A. F. (2018). Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*.

[3] Bartunov, S., Santoro, A., Richards, B. A., Marris, L., Hinton, G. E., and Lillicrap, T. (2018). Assessing the scalability of biologically-motivated deep learning algorithms and architectures. *arXiv preprint arXiv:1807.04587*.

[4] Behrens, T. E. and Sporns, O. (2012). Human connectomics. *Current opinion in neurobiology*, 22(1):144–153.

[5] Bengio, Y., Lee, D.-H., Bornschein, J., Mesnard, T., and Lin, Z. (2015). Towards biologically plausible deep learning. *arXiv preprint arXiv:1502.04156*.

[6] Bengio, Y., Mesnard, T., Fischer, A., Zhang, S., and Wu, Y. (2017). Stdp-compatible approximation of backpropagation in an energy-based model. *Neural computation*, 29(3):555–577.

[7] Borst, A. and Helmstaedter, M. (2015). Common circuit design in fly and mammalian motion vision. *Nature neuroscience*, 18(8):1067–1076.

[8] Cadena, S. A., Denfield, G. H., Walker, E. Y., Gatys, L. A., Tolias, A. S., Bethge, M., and Ecker, A. S. (2019). Deep convolutional models improve predictions of macaque v1 responses to natural images. *PLoS computational biology*, 15(4):e1006897.

[9] Cadieu, C., Kouh, M., Pasupathy, A., Connor, C. E., Riesenhuber, M., and Poggio, T. (2007). A model of v4 shape selectivity and invariance. *Journal of neurophysiology*, 98(3):1733–1750.

[10] Caminiti, R., Carducci, F., Piervincenzi, C., Battaglia-Mayer, A., Confalone, G., Visco-Comandini, F., Pantano, P., and Innocenti, G. M. (2013). Diameter, length, speed, and conduction delay of callosal axons in macaque monkeys and humans. *Journal of Neuroscience*, 33(36):14501–14511.

[11] Carandini, M., Demb, J. B., Mante, V., Tolhurst, D. J., Dan, Y., Olshausen, B. A., Gallant, J. L., and Rust, N. C. (2005). Do we know what the early visual system does? *Journal of Neuroscience*, 25(46):10577–10597.

[12] Cornford, J., Kalajdzievski, D., Leite, M., Lamarquette, A., Kullmann, D. M., and Richards, B. A. (2020). Learning to live with dale's principle: Anns with separate excitatory and inhibitory units. In *International Conference on Learning Representations*.

[13] David, S. V., Hayden, B. Y., and Gallant, J. L. (2006). Spectral receptive field properties explain shape selectivity in area v4. *Journal of neurophysiology*, 96(6):3492–3505.

[14] Douglas, R. J. and Martin, K. A. (2004). Neuronal circuits of the neocortex. *Annu. Rev. Neurosci.*, 27:419–451.

[15] El-Shamayleh, Y. and Pasupathy, A. (2016). Contour curvature as an invariant code for objects in visual area v4. *Journal of Neuroscience*, 36(20):5532–5543.

[16] Ferster, D. and Miller, K. D. (2000). Neural mechanisms of orientation selectivity in the visual cortex. *Annual review of neuroscience*, 23(1):441–471.

[17] Gerstner, W. and Kistler, W. M. (2002). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press.

[18] Gray, C. M. and Singer, W. (1989). Stimulus-specific neuronal oscillations in orientation columns of cat visual cortex. *Proceedings of the National Academy of Sciences*, 86(5):1698–1702.

[19] Hu, J. M., Song, X. M., Wang, Q., and Roe, A. W. (2020). Curvature domains in v4 of macaque monkey. *Elife*, 9:e57261.

[20] Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154.

[21] Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S., Hudspeth, A. J., Mack, S., et al. (2000). *Principles of neural science*, volume 4. McGraw-hill New York.

[22] Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., and Masquelier, T. (2018). Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67.

[23] Kietzmann, T. C., McClure, P., and Kriegeskorte, N. (2019). Deep neural networks in computational neuroscience. In *Oxford research encyclopedia of neuroscience*.

[24] Kim, J. S., Greene, M. J., Zlateski, A., Lee, K., Richardson, M., Turaga, S. C., Purcaro, M., Balkam, M., Robinson, A., Behabadi, B. F., Campos, M., Denk, W., Seung, S. H., and the EyeWireres (2014). Space-time wiring specificity supports direction selectivity in the retina. *Nature*, 509(7500):331.

[25] Kim, T., Bair, W., and Pasupathy, A. (2019). Neural coding for shape and texture in macaque area v4. *Journal of Neuroscience*, 39(24):4760–4774.

[26] Kindel, W. F., Christensen, E. D., and Zylberberg, J. (2019). Using deep learning to probe the neural code for images in primary visual cortex. *Journal of vision*, 19(4):29–29.

[27] Kisvárday, Z. F., Toth, E., Rausch, M., and Eysel, U. T. (1997). Orientation-specific relationship between populations of excitatory and inhibitory lateral connections in the visual cortex of the cat. *Cerebral Cortex (New York, NY: 1991)*, 7(7):605–618.

[28] Ko, H., Cossell, L., Baragli, C., Antolik, J., Clopath, C., Hofer, S. B., and Mrsic-Flogel, T. D. (2013). The emergence of functional microcircuits in visual cortex. *Nature*, 496(7443):96–100.

[29] Kobatake, E. and Tanaka, K. (1994). Neuronal selectivities to complex object features in the ventral visual pathway of the macaque cerebral cortex. *Journal of neurophysiology*, 71(3):856–867.

[30] Koch, C. (2004). *Biophysics of computation: information processing in single neurons*. Oxford university press.

[31] LeCun, Y. and Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

[32] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

[33] Liang, H., Gong, X., Chen, M., Yan, Y., Li, W., and Gilbert, C. D. (2017). Interactions between feedback and lateral connections in the primary visual cortex. *Proceedings of the National Academy of Sciences*, 114(32):8637–8642.

[34] Liu, L., She, L., Chen, M., Liu, T., Lu, H. D., Dan, Y., and Poo, M.-m. (2016). Spatial structure of neuronal receptive field in awake monkey secondary visual cortex (v2). *Proceedings of the National Academy of Sciences*, 113(7):1913–1918.

[35] Marĉelja, S. (1980). Mathematical description of the responses of simple cortical cells. *JOSA*, 70(11):1297–1300.

[36] McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

[37] Miller, K. D. (1994). A model for the development of simple cell receptive fields and the ordered arrangement of orientation columns through activity-dependent competition between on-and off-center inputs. *Journal of Neuroscience*, 14(1):409–441.

[38] Mueller, B. K. (1999). Growth cone guidance: first steps towards a deeper understanding. *Annual review of neuroscience*, 22(1):351–388.

[39] Nandy, A. S., Sharpee, T. O., Reynolds, J. H., and Mitchell, J. F. (2013). The fine structure of shape tuning in area v4. *Neuron*, 78(6):1102–1115.

[40] Pasupathy, A. and Connor, C. E. (2001). Shape representation in area v4: position-specific tuning for boundary conformation. *Journal of neurophysiology*, 86(5):2505–2519.

[41] Plomp, G., Michel, C. M., and Quairiaux, C. (2017). Systematic population spike delays across cortical layers within and between primary sensory areas. *Scientific reports*, 7(1):1–14.

[42] Pospisil, D. A., Pasupathy, A., and Bair, W. (2018). 'artiphysiology'reveals v4-like shape tuning in a deep network trained for image classification. *Elife*, 7:e38242.

[43] Rapela, J., Felsen, G., Touryan, J., Mendel, J. M., and Grzywacz, N. M. (2010). eppr: a new strategy for the characterization of sensory cells from input/output data. *Network: Computation in Neural Systems*, 21(1-2):35–90.

[44] Riesenhuber, M. and Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025.

[45] Roe, A. W., Chelazzi, L., Connor, C. E., Conway, B. R., Fujita, I., Gallant, J. L., Lu, H., and Vanduffel, W. (2012). Toward a unified theory of visual area v4. *Neuron*, 74(1):12–29.

[46] Shirasaki, R. and Pfaff, S. L. (2002). Transcriptional codes and the control of neuronal identity. *Annual review of neuroscience*, 25(1):251–281.

[47] Singer, W. (1999). Neuronal synchrony: a versatile code for the definition of relations? *Neuron*, 24(1):49–65.

[48] Swadlow, H. A. and Waxman, S. G. (2012). Axonal conduction delays. *Scholarpedia*, 7(6):1451. revision #125736.

[49] Thorpe, S., Delorme, A., and Van Rullen, R. (2001). Spike-based strategies for rapid processing. *Neural networks*, 14(6-7):715–725.

[50] Tsao, D. Y., Freiwald, W. A., Tootell, R. B., and Livingstone, M. S. (2006). A cortical region consisting entirely of face-selective cells. *Science*, 311(5761):670–674.

[51] Vanni, S., Hokkanen, H., Werner, F., and Angelucci, A. (2020). Anatomy and physiology of macaque visual cortical areas v1, v2, and v5/mt: bases for biologically realistic models. *Cerebral Cortex*, 30(6):3483–3517.

[52] Yamins, D. L., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D., and DiCarlo, J. J. (2014). Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences*, 111(23):8619–8624.

[53] Zador, A. M. (2019). A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature communications*, 10(1):1–7.

[54] Zhang, Y., Lee, T. S., Li, M., Liu, F., and Tang, S. (2019). Convolutional neural network models of v1 responses to complex patterns. *Journal of computational neuroscience*, 46(1):33–54.